ID N. 16

UNIVERSITÀ CAMPUS BIO-MEDICO DI ROMA

DEPARTMENT OF ENGINEERING

UNIVERSITY OF ROME TOR VERGATA

DEPARTMENT OF ENTERPRISE ENGINEERING

Italian National Ph.D. in Artificial Intelligence
Health and Life Sciences
XXXVII Cycle

Situated Natural Language Understanding in HRI via LLMs

Supervisors

Prof. Roberto Basili
Dr. Danilo Croce

Candidate

Claudiu Daniel Hromei

Januray, 2025

Per aspera sic itur ad astra.

Acknowledgements

This PhD journey has been full of intellectual challenges, intense work, and constant stimulation.. but also of friendships, laughter, jokes, and unforgettable moments! I am deeply grateful to the research group where I had the chance to grow, not only as a researcher but also as a person.

I would like to express my heartfelt thanks to Professor Roberto Basili, who believed in me from the very beginning. He has been an extraordinary mentor, offering valuable guidance, sharing his vast knowledge, and never hesitating to challenge me when it was necessary for me to learn. I will never forget the late evenings spent revising papers and slides together, nor the lessons on how to present at a conference. His dedication and vision have been a constant source of inspiration. My deepest gratitude also goes to Dr Danilo Croce, who has been like an older brother throughout my PhD. He taught me the everyday essentials of being a researcher: from writing code and debugging late into the night, to brainstorming ideas for papers and experiments, to never mixing train and test sets. He helped me navigate all the responsibilities I've taken on during these years, with patience, humour, and wisdom. I owe a great deal to both of them: they have shaped who I am today and continue to be my role models.

I also want to thank my fellow travellers. friends who have made this experience joyful, sometimes stressful, but most of all, memorable. In particular: *Daniele, mito e legenda*, *Antonio, il maestro* and *Federico, dive and dipe*. These years have flown by, and I've been shaped by their presence too.

A special thank you goes to Flavia, my partner, who has always been by my side. She supported and endured me with incredible patience and kindness, even when I was fully absorbed by deadlines, papers, and experiments. She managed to lift me up even in the darkest moments. Her support has been vital.

Last but not least, I want to thank my parents, who have always stood by me, no matter what path I chose. Their unconditional support, love, and trust made all of this possible.

Abstract

This thesis focuses on advancing situated natural language understanding for human-robot interaction (HRI) through the use of large language models (LLMs), aiming to create robots that can understand, process, and respond to human commands in real-world environments. The work addresses key challenges in integrating multitasking capabilities, scaling models across multiple languages and modalities, including visual data, and adopting sustainable training techniques. The key contributions of this work include:

- **Integrating Multitask learning into LLMs**: Methods enabling LLMs to integrate various problem-solving approaches are explored, initially handling tasks individually and then within a unified structure. This is demonstrated through the **ExtremITA** model, which participated in the EVALITA challenge on a diverse set of Italian-specific linguistic tasks. Following the exploration of multitasking capabilities, the thesis further investigates the application of LLMs for syntactic analysis across languages.

- **Neural transcoding for grammatical parsing based on LLMs**: The thesis introduces **U-DepPLLaMA**, a model for universal dependency parsing using large autoregressive language models. This model achieves state-of-the-art results in dependency parsing across multiple languages and demonstrates the feasibility of scaling models using low-rank adapted parameters. It can handle multiple languages without task-specific architectural modifications.

- **Grounding Language Understanding in HRI**: Building on these capabilities, the thesis presents the **GrUT** approach, which interprets robotic commands in multiple languages, particularly English and Italian. This method combines frame semantics, a knowledge base, and lexical similarity to understand natural language commands.

- **Multimodal Interaction**: The study incorporates multimodal models, with a particular focus on visual question answering (VQA) in Italian, using the GQA-it dataset. The **MiniCPM-V** model was optimised to improve its performance on the GQA-it dataset.

- **Developing the MM-IGLU Resource**: The thesis introduces **MM-IGLU**, an interactive multimodal resource for grounded language understanding. It also presents **MM-IGLU-it**, an extension that supports the Italian language. These resources facilitate the training and evaluation of models in a multilingual context, grounded in the Minecraft-like world through environmental images.

- **Dialogue Systems in HRI**: Preliminary work is presented on creating a dialogue resource on **MM-IGLU**, where a robot can ask follow-up questions to clarify commands, and evaluating the abilities of Multimodal models in effectively planning the interaction and solving the ambiguities of input commands.

The thesis also provides a comprehensive evaluation of various models, using different, automatic or manual metrics. Error analyses are conducted to evaluate the strengths and limitations of the models. In summary, these contributions collectively pave the way for developing more efficient, versatile, and interactive robots capable of understanding complex commands, performing tasks in multilingual and multimodal contexts, and seamlessly integrating into real-world human environments.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Language is a uniquely human trait that distinguishes us from other species. It is not merely a means of communication but a fundamental aspect of our identity and culture. From early development, humans exhibit an extraordinary ability to learn and use language. This ability to understand and produce an infinite number of sentences from a finite set of elements is known as the generative nature of language [24]. The cognitive aspect of language involves various mental processes, including perception, memory, and reasoning, which allow us to construct complex grammatical structures and convey nuanced meanings [80, 124, 166].

More importantly, language enables humans to **exchange information** and achieve mutual understanding. From early development, humans exhibit an extraordinary ability to learn and use language, a skill deeply connected to imitation and tool use [176]. Natural pedagogy plays a crucial role in how humans exchange information and refine understanding through shared interactions [66]. As children, we start communicating with our parents, asking questions about the world and the meanings of words. This process helps us build our internal representations of concepts [194], which, although initially imperfect, can be refined over time. As we grow older, through language and symbols we study subjects such as grammar, mathematics, physics, and philosophy, acquiring knowledge and building more sophisticated internal representations of our world. When we encounter gaps in our understanding, we ask questions in natural language to seek clarification. Language also allows us to express our desires, plans, and viewpoints. We navigate nearly every aspect of life through language, making it an endlessly fascinating phenomenon. The diversity of languages plays a critical role in shaping cognitive processes, challenging the idea of universal linguistic principles [80]

Since the dawn of time, humans have been captivated by the idea of creating something and then **having a chat** with it. From the Golem of Jewish folklore [234], animated to

perform tasks and sometimes capable of understanding commands, to Hephaestus' golden maidens in Greek mythology [221], designed to speak and serve their creator, this fascination reflects a fundamental part of our nature. The development of a robot, a truly artificial being capable of seamless interaction, holds the potential to revolutionize our world. In the field of Human-Robot Interaction, researchers study how to facilitate clear communication between humans and robots [259], through language. Humans should be able to command robots using natural language, specifying their objectives and desired actions. In turn, robots must understand these utterances, create an internal representation [111, 113], determine the intent behind the commands, and identify any missing information [117, 6]. If necessary, robots should ask clarifying questions to gather the required information, check for inconsistencies, and decide if the requested action is feasible [118, 37, 5]. When a human requests information, the robot should respond in natural language to satisfy the information needs related to a query/request. If the request involves a physical action, the robot should be capable of executing it. Interactive task learning allows robots and agents to acquire new skills through natural interactions with humans [151]. Collaborative dialogue frameworks, such as those explored in Minecraft, demonstrate the complexities of human-agent communication [193]. Moreover, simulated environments like provide a testbed for teaching agents to execute instructions via dialogue [126].

For example, consider a dialogue between a human and a domestic robot in an automated house. The human needs help finding an item, and the robot assists by asking questions about the description and the possible location of the item. After gathering the necessary information, the robot successfully locates the item and delivers it to the owner.

H_1: Hey, Robo, I need your help.

R_2: Of course! What do you need assistance with today?

H_3: I can't seem to find my glasses. Can you help me locate them?

R_4: Understood. Can you please tell me where you last remember using your glasses?

H_5: I think I had them in the living room last night while I was watching TV.

R_6: The living room, got it. Do you remember if you placed them on any specific surface?

H_7: Maybe on the coffee table or the side table next to the couch.

R_8: Ok. I will check both the coffee table and the side table. Would you like me to check any other areas in the living room? Maybe the sofa?

H_9: No, I think those are the most likely spots.

R_{10}: I'm going to the living room now... I see something on the side table next to the couch. Confirming... Yes, these are your glasses. I'll bring them to you.

H_{11}: Thanks, Robo!

R_{12}: You're welcome! Is there anything else you need help with?

H_{13}: Actually, yes. Could you remind me to water the plants in an hour?

R_{14}: Sure, I will set a reminder for you to water the plants in one hour.

H_{15}: Perfect. Thanks again!

R_{16}: Happy to help! Don't hesitate to ask if you need anything else.

1.1 Natural Language and Human Communication

At its core, **communication** involves the exchange of information between individuals through various mediums, with natural language and vision being paramount. Human communication is not merely about the transmission of words; it encompasses a rich tapestry

of verbal and non-verbal cues. These include body language, facial expressions, gestures, and intonation, all of which contribute to the effectiveness of the message being conveyed. The interplay between these elements allows for nuanced and layered interactions. **Natural language** is the primary tool humans use to convey thoughts, emotions, and information. It is a structured system of symbols, sounds, words, and sentences, governed by grammatical rules that enable us to produce and understand an infinite variety of expressions.

The interplay of utterances of different speakers makes a dialogue. In **conversation**, dialogue is driven by underlying intents and goals, which shape both the flow and purpose of the exchange. Intents refer to the speaker's immediate purpose or motivation, such as seeking information, providing clarification, persuading, or building rapport. Goals represent the broader outcomes the speaker aims to achieve, such as reaching an agreement, resolving a problem, or planning a trip. These elements influence not only what is said but also how it is communicated, affecting tone, word choice, and conversational strategies.

Dialogue is also characterized by **turn-taking**, where speakers alternate roles to maintain a coherent and dynamic exchange, as exemplified in the conversation above. This process is guided by social and cultural norms that dictate when and how individuals should speak, listen, and respond. Effective communication requires participants to recognize cues that signal the end of one turn and the beginning of another. These cues can be verbal, like pausing at the end of a sentence, or non-verbal, such as maintaining eye contact or using hand gestures. At its core, the primary purpose of dialogue is the **exchange of information**, which can take many forms, from sharing facts and opinions to expressing emotions and intentions. Successful communication depends on the clarity and relevance of the information shared, as well as the ability of the participants to understand and interpret it accurately.

For communication to be effective, the interlocutors must establish **common ground**: a shared understanding of the topic at hand. This involves the mutual recognition of knowledge, beliefs, and assumptions. For instance, in the dialogue above, the human understands that they can ask the robot to perform tasks like finding objects (H_3) or setting alarms (H_{13}). Similarly, the robot isn't surprised by these requests (R_4), nor is the human by the robot using its vision (H_{11}). Both are familiar with each other's capabilities and interaction styles, leading to a seamless communication dynamic. Moreover, this common ground is built and maintained through **confirmations and questions**. Confirmations signal agreement or understanding, while questions seek clarification or additional information, helping to ensure that both parties are on the same page. Confirmations can be explicit, such as saying "*I understand*" (R_4), "*Ok*" (R_8) or "*I agree*", or implicit, such as providing appropriate follow-up comments. For example, an implicit confirmation involves repeating part of the turn of the interlocutor: in R_{14} the robot repeats that he will set a reminder to wa-

ter the plants in one hour, to assure the human what it understood. These signals help to validate the information shared and reinforce mutual understanding. Questions play a crucial role in dialogues. They can be open-ended (R_4), encouraging expansive responses, or closed-ended (R_8), seeking specific information. Questions help to clarify ambiguities, probe for deeper insights, and guide the direction of the conversation. They are essential tools for exploring common ground and ensuring that the exchange of information is accurate and comprehensive. Seminal works laid the foundation for using procedural representations to understand natural language [277]. More recent advances focus on problem-solving agents that can communicate effectively while learning from interactions [192, 1, 288, 131].

Vision complements natural language by providing additional context and meaning to verbal exchanges. Facial expressions, eye movements, and body language are visual cues that convey emotions and intentions, often reinforcing or contradicting spoken words [91]. Facial expressions are powerful indicators of emotional states. A smile can indicate happiness or agreement, while a frown might suggest confusion or disagreement. These visual signals can enhance the listener's understanding of the speaker's message and emotional tone [79]. Eye movements and contact play a critical role in regulating turn-taking and signalling attention and interest. Sustained eye contact can indicate attentiveness and engagement, while frequent shifts in gaze might suggest discomfort or distraction. Body language encompasses gestures, posture, and movements. Open body language, such as uncrossed arms and leaning forward, can indicate openness and interest, while closed body language, such as crossed arms and leaning back, might suggest defensiveness or disengagement [134].

It is worth noting that **common sense** [210] plays an intriguing role in our interactions as well. It involves the ability to make sound judgments based on everyday knowledge and experiences. It underpins much of our decision-making and problem-solving processes, contributing to effective communication by allowing individuals to infer meanings and intentions beyond the literal words spoken [129]. For example, in the above conversation, when the human asks the robot to find their glasses, the robot infers that the human likely wants them. So, it first locates the glasses and then brings them to the human, even though this wasn't explicitly requested. This behavior is guided by common sense knowledge: when people ask to find some of their belongings, they usually need or intend to use them. Notice that this is particularly true with glasses that allow to improve the ability to see and recognize visually the environment, something that humans sometimes need. The actions of the robot should align with this understanding. However, a detailed exploration of common sense and non-verbal communication is beyond the scope of this thesis, as human communication is a dynamic and intricate process that relies heavily on natural language and visual cues. Through dialogue, turn-taking, and the exchange of information, individuals build

common ground, ensuring effective and meaningful interactions. While common sense and body language are captivating elements of human cognition, their complexities are reserved for further study. Understanding the fundamental aspects of communication enhances our appreciation of the rich tapestry of human interaction.

1.2 Situated Natural Language Understanding in Human Robot Interaction: problems and objectives

The interaction between humans and robots is rapidly evolving, driven by advancements in Natural Language Processing (NLP) and **Human-Robot Interaction** (HRI). A taxonomy to structure and analyze human-robot interaction provides a framework for understanding diverse interaction scenarios [198, 89]. The rise of Large Language Models (LLMs) has brought significant improvements in the conversational abilities of robots. Among these, ChatGPT [199] represents a key milestone due to its powerful natural language processing abilities. This model excels in engaging users in fluid, human-like conversations while efficiently handling a wide range of linguistic tasks, from paraphrasing and summarizing to text interpretation. However, these models still face challenges in fully understanding and interpreting human language [43], especially in the context of real-world references. This Thesis highlights the complexities of situated natural language understanding in HRI, focusing on the problems and objectives that define this field. Human-robot interaction should be intuitive, efficient, and effective. Robots are expected to understand and respond to human language in a way that feels natural [89], as expressed in the previous section, while also being capable of interpreting visual cues and situational context. Ideal interaction involves a seamless integration of language and vision, allowing robots to engage in meaningful and contextually appropriate dialogues with humans about the surrounding world. Recent reviews highlight the integration of machine learning in human-robot collaboration [236, 249]. The advent of **Large Language Models** has revolutionized NLP and HRI. These models, such as OpenAI's GPT series [199], are highly proficient in generating human-like text and engaging in coherent conversations. However, they have limitations:

- **Understanding and Interpretation**: LLMs excel in generating text but struggle with deeper understanding and interpretation, particularly when it involves making references to the real world.
- **Contextual Awareness**: Current models lack the ability to perceive and interpret physical and situational context effectively, which is essential for grounded interactions.

> Most of the time, they simply integrate an image with text to answer questions related to it. However, true contextual awareness involves more than this. It requires the model to discern whether the visual context is relevant: maintaining consistent answers when the visual input is irrelevant, and appropriately altering responses when the visual input is indeed pertinent. Achieving this level of awareness is critical for robust and meaningful multi-modal understanding.

One of the critical problems in HRI is enabling robots to **interpret spoken language** accurately to make informed decisions. Robots must understand the intent behind human instructions and consider the contextual implications to execute tasks effectively. Misinterpretations can lead to errors or inappropriate actions, highlighting the need for robust understanding mechanisms. Robots often serve as passive responders in interactions, providing answers to human queries. However, for effective communication and task execution, robots need to ask questions as well [117, 118, 37]. They need to implement the mechanics and behaviors from the **human-human interaction**, such as turn-taking, confirmations and asking questions when information is missing or something is not clear. Situations where robots should ask questions include clarifications, to ensure they understood the instructions correctly, additional information gathering about the task or environment, or disambiguation in human utterances. Developing strategies for when and how robots should ask questions is essential for improving their interaction capabilities.

For robots to function as real embodied agents, **grounded** or **situated interaction** is crucial. This involves (a) referencing the real world, where robots must relate language to physical objects and environments and (b) situational awareness, where understanding and interpreting the context in which interactions occur is fundamental. This is called Grounded Interaction, which is challenging but essential for robots to perform tasks effectively in the real world. In [250], the authors emphasize the importance of context in human-computer interactions, arguing that effective communication requires systems to be aware of and responsive to the situational context of their use. In [98] the symbol grounding problem is introduced, discussing how abstract symbols acquire meaning through their connection to physical entities and experiences, laying the foundation for grounded cognition in AI. A behavior-based approach to robotics is presented in [38], emphasizing the importance of grounding robotic actions in real-world perceptions. In [103], the authors explore how language learning agents can ground language in perception and action, contributing to the development of AI systems capable of understanding and generating contextually appropriate language. Finally, building interactive agents that learn to solve tasks through grounded natural language instructions in collaborative environments is showed in [133]. In this scenarios, vision is vital for situated interaction as it provides contextual information

that complements language. **Visual perception** allows robots to: identify objects and environments; interpret gestures and facial expressions; and understand spatial relationships and situational context. Combining language and vision enables robots to engage in more meaningful and contextually appropriate interactions, resembling how humans interact with each other.

Figure 1.1: A 3D simulation of a house.

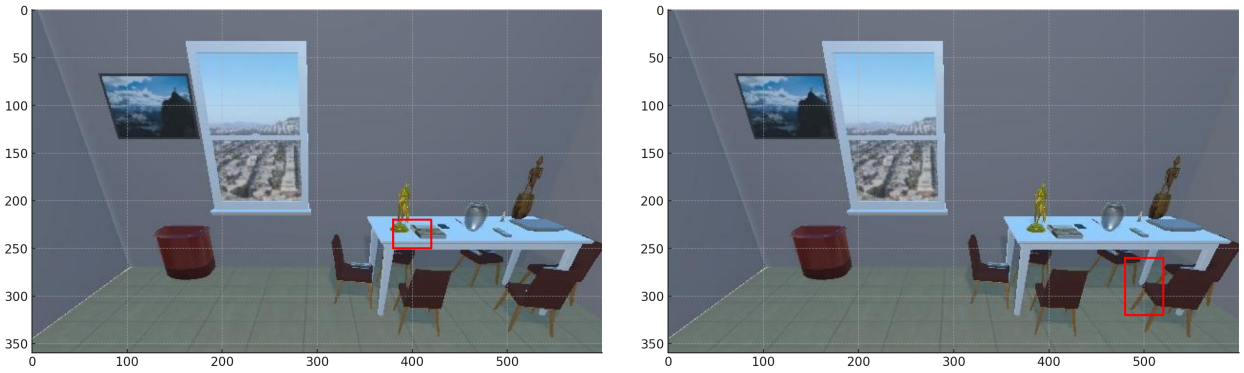
In a **typical interaction** with an Intelligent Agent, consider the scenario depicted in Figure 1.1, which illustrates a graphical 3D capture of part of a living room. In this scene, there is a table surrounded by chairs, some furniture, books scattered on the table, a painting hanging on the wall, a bin, and a window situated next to the painting. If a user wants the robot to perform a task, they would typically convey their request in natural language, as discussed earlier, relying heavily on the vision and cognitive abilities of the robot to interpret the environment. For example, the user might issue a command like, "*Take the book from the table and throw it away please, it is ruined now!*". To carry out this task, the Intelligent Agent must first identify the relevant objects, the book and the table. In this context, "identifying" an object involves two possible approaches: *(1)* using the vision system of the robot to draw bounding boxes on the image, visually highlighting the objects to dynamically compute their position and distance, or *(2)* retrieving the objects and their positions from the agent's internal knowledge base, which may already store spatial and object information. In many cases, one of these methods is sufficient to solve the task, but an ideal Intelligent Agent would employ both strategies for increased accuracy and robustness.

Home automation robots, like the iRoomba and similar devices, typically rely on laser sensors to create a map of their operating environment [77, 191]. These sensors allow the robot to define the boundaries of its navigable space, a strategy that proves highly effective for tasks like cleaning [167, 148]. However, when people imagine an intelligent robot, they often envision a humanoid android capable not only of moving but also engaging in conversations about the world around it. The most interesting and complex approach is the one that relies on vision, as it mimics human-like perception and requires the agent to dynamically process visual input from the environment. Once the objects are identified, whether through vision, internal knowledge, or a combination of both, the robot must make a series of decisions. These include planning its movements to approach the table, safely grasp the book, and then navigate toward the bin. The agent must also be aware of obstacles, such as chairs around the table or other objects on the floor, to avoid collisions. Furthermore, context-aware reasoning is essential: for instance, understanding that "throw it away" means placing the book in the bin, as opposed to simply throwing it on the floor. The task also involves a degree of physical manipulation, such as adjusting grip strength to avoid damaging the book, especially if it is damaged and fragile. When reaching the bin, the robot must release the book with precision, ensuring it lands inside the bin, not outside of it. Additional considerations include the agent's ability to handle dynamic environments, different lighting conditions, multiple books on the table (requiring the robot to identify the correct one), or if the bin has been moved from its expected position. In such cases, the Intelligent Agent should update its understanding of the scene in real time to adapt to the changing conditions. Thus, executing this seemingly simple command involves an intricate combination of natural language understanding, real-time visual perception, spatial reasoning, and motor control. The challenge lies in seamlessly integrating these components to create robust, real-world Intelligent Agents capable of adapting to complex environments.

While **ChatGPT**, considered one of the most advanced dialogue frameworks at the time of writing, excels in generating coherent text and demonstrates an impressive breadth of (albeit shallow) knowledge, its class of models often lack the ability to integrate visual information with contextual awareness. As previously discussed, complex requests involving real-world objects or situational context can easily confuse language-only models. Introducing visual input, such as an image, adds a layer of complexity that requires the agent to reason across both language and vision. This challenge exposes the limitations of current models when it comes to handling grounded, multimodal interactions. For instance, in the interaction involving Figure 1.1, ChatGPT[1] exhibited an excellent understanding of the task through its language-processing capabilities. However, it consistently struggled to identify

[1]Version 4o, accessed at https://chatgpt.com/ in September 2024.

objects or infer the necessary knowledge to execute actions like throwing something away. Interestingly, while the model did produce bounding boxes for the book and the bin, they were often incorrect or misaligned with the actual objects, as shown in Figure 1.2. This highlights the gap in current AI models when it comes to combining language comprehension with visual reasoning, emphasizing the need for more advanced systems capable of fully grounded interactions.



(a) (b)

Figure 1.2: The Bounding Boxes ((a) book, (b) bin) produced by ChatGPT for the environment in Figure 1.1.

In the context of advancing model capabilities, one promising direction is **multi-task training**, where a model is trained to handle a variety of tasks simultaneously. Multi-tasking enhances the inherent flexibility of Large Language Models by enabling them to process and solve multiple distinct problems within a unified architecture. This approach brings several key advantages: first, it encourages knowledge transfer between tasks, allowing the model to leverage shared patterns and representations across different domains. For example, understanding syntax in one task may assist the model in parsing complex instructions in another, which improves its overall performance and efficiency across tasks. In this Thesis, the beneficial training was first explored and discussed in the ExtremITA model [109, 110], where I participated in the Italian NLP challenge, EVALITA [147], an evaluation campaign on a diverse set of linguistic tasks specific to the Italian language. ExtremITA was the first LLM applied to all 22 tasks in this challenge using a multi-tasking approach, where the model was trained to handle tasks such as text classification, sentiment analysis, fake news detection, and named entity recognition, all within a unified architecture.

The results of ExtremITA (the model won 9 out of the 22 tasks and achieved top 3 performance in 16 out of the 22) highlight the significant advantages of a multi-tasking scenario, as the model was able to transfer knowledge across different domains. The ability to generalize from one task to another led to overall stronger performance, demonstrating that

a multi-tasking approach can outperform traditional task-specific models, which may lack the ability to leverage cross-task insights. The **benefits of multi-tasking** are particularly apparent in real-world applications like HRI, where agents must integrate several modalities and respond to a variety of requests. In these cases, multi-tasking capabilities enable models to be more adaptive, moving beyond language-only interactions to incorporate visual reasoning, spatial awareness, and environmental context, all within a single framework. This unification reduces the need for specialized models for each task, simplifying model engineering and deployment, and reducing computational overhead.

Another challenge in this domain is representing **contextual information** that is not always strictly textual. Robots and intelligent agents often receive non-textual inputs from their sensors, such as data about the environment, movement, or proximity to objects. This data must somehow be processed and integrated into the decision-making process. One method for handling this is Textification, the process of converting **non-linguistic** data into textual format so that it can be processed by language models. By converting sensor data or environmental context into a textual description, the robot can reason about it in the same way it processes traditional language inputs. For example, a robot could convert its sensor readings about room temperature, object proximity, or movement into textual statements like "*the room is warm*" or "*the object is 2 meters away*". This enables the model to respond to real-time sensor data and make decisions that involve not only the task at hand but also the broader environmental context.

Another significant complexity in developing and training Large Language Models arises from the increasing **hardware requirements**. Over the past few years, there has been an exponential growth in the number of parameters these models utilize, which directly translates to higher demands for GPU memory and cutting-edge technologies. To put this into perspective, LLMs have evolved from containing a few hundred million parameters to nearly a hundred billion in just a few short years, representing an increase of three orders of magnitude. This surge in model size brings several challenges, as traditional hardware setups struggle to keep up with the ever-increasing demand for memory, compute power, and processing time. Training models of this scale typically requires high-end GPUs with substantial memory capacity. For instance, state-of-the-art models with hundreds of billions of parameters, such as GPT-3 [216], GPT-4 [199, 200], PALM540B [52], Qwen32B [283], LLaMA3.2 405B [92], Claude3.5 [120], Mistral [186], Falcon [9], often need GPUs with 48GB or more memory to handle full training processes efficiently. However, such GPUs are both expensive and inaccessible to many researchers and smaller organizations, creating a bottleneck for innovation and equitable development in the AI space. Additionally, the computational costs are further exacerbated by the sheer number of iterations needed to

fine-tune these models, which not only requires more time but also significantly drives up energy consumption [229, 122, 16], contributing to environmental concerns in AI research.

One promising avenue to mitigate these challenges is the adoption of techniques aimed at **reducing memory and computational overhead** without compromising model performance. Approaches such as Low-Rank Adaptation (LoRA) [119] and lower-bit approximations are becoming increasingly popular. LoRA, for example, reduces the number of parameters that need to be updated during training by decomposing large-weight matrices into smaller, low-rank matrices. This method drastically lowers memory requirements, enabling the training of models with billions of parameters on standard GPUs with much less memory. Recent studies, such as [109], have demonstrated that with these advanced techniques, a 7-billion-parameter model can be trained using a standard NVIDIA T4 GPU with just 16GB of memory, as opposed to the 48GB typically required for full training. This represents a significant reduction in hardware requirements and democratizes access to cutting-edge AI research. In addition to LoRA, lower-bit approximation methods further reduce memory usage by representing model weights in fewer bits (e.g., 8-bit or even 4-bit precision) instead of the traditional 16 or 32 bits. This technique has been shown to retain much of the accuracy while significantly cutting down on memory and computational requirements, making it a compelling solution for training larger models on more affordable hardware.

1.3 Thesis Contribution

Given the problems highlighted before, the following topics have been studied and explored in this thesis.

- **Integration of Multi-tasking in LLMs**: A key contribution of this thesis is the exploration of multi-tasking capabilities in LLMs. While LLMs excel at processing natural language, they often struggle to manage multiple tasks concurrently or solve complex, multi-dimensional problems. This thesis examines methods that enable LLMs to integrate diverse problem-solving approaches, firstly one at a time, and then within a unified framework. By employing a multi-tasking architecture, LLMs can process different types of inputs and contextual information, enhancing their performance in situated interactions. This approach marks a step toward creating more versatile and adaptive AI systems capable of solving heterogeneous tasks, ultimately improving overall performance through synergy and diversification.

- **Integration of Vision and Language for Situated Interaction**: Integrating vi-

sion and language is crucial for enabling LLMs to handle grounded, situated dialogue. This thesis explores methods for combining visual perception with natural language processing to enhance an agent's ability to interact meaningfully with its environment. By incorporating both modalities, the agent can reason not only based on textual input but also by interpreting visual scenes. This integration is essential for achieving contextually grounded understanding, which is necessary for tasks such as object manipulation, navigation, and situational decision-making in real-world scenarios.

- **Analysis of Situated Interpretation and Dialogue Capacity in Large Language Models (LLMs)**: This thesis provides an in-depth analysis of situated interaction in LLMs, focusing on their ability to interpret and respond to commands in real-world contexts. Through use cases such as interactions in an automated smart home environment and collaborative tasks within a simulated Minecraft-like world, this work examines how LLMs perform when tasked with command comprehension and command replying, involving physical objects and contextual information. The results reveal significant limitations of language-only models, which often struggle to interpret user commands involving spatial relationships, object identification, or situational awareness. These use cases emphasize the gap between current LLM capabilities and the complex demands of real-world human-robot interactions.

Together, these contributions aim to push the boundaries of Human-Robot Interaction by enabling systems to perform grounded, multimodal tasks in real-world contexts while remaining efficient and accessible.

1.4 Thesis Organization

The organization of this thesis follows a progressive structure, designed to build a deep understanding of situated natural language understanding (NLU) in Human-Robot Interaction (HRI) via Large Language Models (LLMs). Each chapter introduces key concepts, presents methodologies, and analyses results that can be built upon for the development of robust, interactive robots.

- **Chapter 1: Introduction**: This chapter introduced the fundamental motivations and challenges of applying NLU to HRI. It outlined the problem of enabling robots to understand, interpret, and act upon human instructions in real-world environments. By framing language as the primary medium of interaction, it sets the stage for the subsequent exploration of semantic interpretation and multimodal understanding.

- **Chapter 2: Modern Approaches to Situated Natural Language Understanding**: This chapter provides a comprehensive literature review on modern approaches to NLU and HRI, starting with foundational concepts and leading to advanced multimodal models. Section 2.1 explores the intersection of NLP and HRI, highlighting the necessity of natural language as a tool for human-robot collaboration. It discusses the challenges of grounding language in a physical context and provides an overview of existing frameworks. Then, SRL is presented in Section 2.2 as a crucial component for enabling robots to understand the meaning of sentences by identifying the roles played by different entities (e.g., agent, object, location). This section explains how SRL supports command interpretation and execution in robotic systems. Section 2.3 describes the evolution of machine learning models for NLP, culminating in the introduction of the Transformer architecture. In Section 2.4 an overview of LLMs, focusing on their scalability and generalization capabilities, is presented by emphasizing their potential in zero-shot and few-shot learning, which are key for developing adaptable robots that can handle new tasks with minimal retraining. As interaction is made of not only language, but more than one modality, Section 2.5 discusses the integration of vision and language, a critical aspect for robots operating in dynamic environments. Finally, given the high computational costs of training LLMs, Section 2.6 introduces techniques like Low-Rank Adaptation (LoRA) and Q-LoRA. These methods enable efficient fine-tuning on constrained hardware, making them practical for real-world robotic applications.

- **Chapter 3: Improving the Training in LLMs for Complex Linguistic Tasks**: This chapter presents the methodologies developed to train LLMs on complex linguistic tasks under hardware limitations. Each section focuses on a specific approach, to lay down the fundamentals of adapting LLMs on specific task. Section 3.1 describes the fine-tuning of LLMs on producing semi-structured text from natural language. The relevance to HRI lies in providing robots with the ability to understand syntactic structures, aiding in the correct interpretation of commands. As we would like robots to solve more than one specific linguistic task and be able to operate in a multi-task scenario, Section 3.2 presents ExtremITA as a multi-task learning approach, where a single model handles diverse NLP tasks. Finally, Section 3.3 introduces the integration of multimodal inputs (text and images) into LLMs. It highlights how combining visual and linguistic information improves the ability of a robot to interpret its surroundings and respond accordingly.

- **Chapter 4: Natural Language Understanding in HRI**: This chapter shows the

application of the developed models in real-world HRI scenarios. It is divided into three main sections: Section 4.1 focuses on enabling robots to interpret and act upon human commands using SRL and grounding techniques. Examples of real-world scenarios are provided to illustrate the practical impact of such an architecture if installed onboard of robots. Section 4.2 discusses the use of interactive dialogue systems in collaborative tasks, where robots must clarify ambiguous commands and request additional information when needed. It explores the Situated Interaction through a Minecraft-like game made of blocks. Finally, Section 4.3 presents the MM-IGLU resource, developed for training and evaluating multimodal models. The resource is designed to support robots in tasks that require both linguistic and visual understanding.

- **Chapter 5: Conclusions and Discussion**: The thesis concludes by summarising the main contributions and discussing their implications for future research in Natural Language Understanding (NLU) and Human-Robot Interaction (HRI). In 5.2, a preliminary exploration of recent language and multimodal models is presented, by applying one of them to a targeted task to assess its suitability and test the impact of more recent progress. This is followed by a discussion of open challenges and potential research directions in 5.3, with an emphasis on the overarching goal of developing more adaptive and intelligent robotic agents. The chapter culminates in 5.3.1 with the proposal of a Cognitive Architecture that integrates the capabilities presented throughout the thesis, compact and efficient enough to be deployed onboard a physical robot, and designed to support Situated Natural Language Interaction with humans.

Chapter 2

Modern Approaches to Situated Natural Language Understanding

This chapter surveys modern approaches to Situated Natural Language Understanding (Situated NLU), a critical field for enabling intelligent agents, such as robots, to interpret and act upon human language in real-world environments. The ability to ground language in a physical or visual context is essential for tasks where language alone is insufficient to convey meaning, such as interacting with a dynamic environment or executing commands that require spatial or temporal reasoning. Situated NLU is at the core of this thesis, as the primary goal is to extend the capabilities of multimodal large language models (LLMs) to handle complex dialogues that require both linguistic interpretation and environmental awareness.

To set the foundation for this research, the chapter begins by presenting key concepts in Human-Robot Interaction (HRI) and Natural Language Processing (NLP), which are pivotal in shaping modern approaches to language understanding in situated contexts. We then explore the Semantic level of Natural Language Interpretation by using the Semantic Role Labeling (SRL) task as an example. This task is central to structured semantic interpretation, as it provides a framework for understanding the roles played by different entities in a given instruction or sentence. Understanding these roles is fundamental for disambiguating commands and ensuring that robots interpret language correctly within their environment.

The chapter continues by exploring state-of-the-art Transformer architectures and Large Language Models (LLMs), which have revolutionized NLP and are key components of modern NLU systems. It will discuss their evolution, highlighting models that are particularly relevant for this work, including instruction-tuned LLMs that improve task generalization by learning from explicit human-provided instructions. Since the thesis aims to extend such

models to multimodal and dialogic scenarios, it will review how instruction tuning, when combined with visual grounding, enhances model performance in real-world tasks.

Following this, the multimodal architectures are examined, which integrate language and vision and play a crucial role in situated dialogues, where the context is provided not only by the text but also by the visual scene. Multimodal learning is a cornerstone of this thesis, as it forms the basis for extending LLMs to handle situated interactions effectively. The key works in multimodal NLU will be highlighted and their relevance to this approach is discussed.

Finally, the chapter is concluded with a discussion on sustainable training techniques. Given the resource-intensive nature of large-scale models, efficient training methodologies are essential for ensuring that future models remain accessible and environmentally responsible. Techniques such as parameter-efficient fine-tuning (e.g., LoRA and Q-LoRA) and quantization strategies are particularly relevant, as they allow the adaptation and extention of LLMs for specific tasks, languages, and environments without requiring excessive computational resources.

By the end of this chapter, the reader will have a comprehensive understanding of the current state of the art in Situated NLU, which serves as the foundation for the contributions presented in this thesis. Each of the surveyed areas, such as HRI, SRL, LLMs, multimodal architectures, and sustainable training, addresses a critical aspect of the challenges this thesis aims to overcome and provides the necessary context for the novel approaches developed in the subsequent chapters.

2.1 HRI and Natural Language Processing

Human-Robot Interaction (HRI) is a multidisciplinary field that seeks to design, implement, and study communication between humans and robots. Among the many modalities of interaction, natural language processing (NLP) is one of the most intuitive and effective, as it allows humans to interact with robots in the same way they communicate with each other. The goal of NLP in HRI is to enable robots to understand, interpret, and generate human language in a way that is meaningful and grounded in the physical and social context of the interaction [89, 7].

One foundational challenge in HRI is recognizing the intents and objectives behind human communication. As Austin (1962) famously argued, language is inherently tied to the speaker's intentions and actions [19]. For example, when a human tells a robot "*I can't seem to find my glasses, can you help me locate them?*", the intent is to have the robot locate the correct object (*the glasses*) in the environment and perform the action (Bring Them).

Understanding these intents requires dialogue systems to process dialogue acts, which represent the purpose of a speaker's utterance (e.g., a command, request, or question) [8, 61]. Dialogue act annotation schemes like DAMSL (Dialogue Act Markup in Several Layers) [8] provide a structured way to identify these intentions in human communication.

In recent years, dialogue systems for HRI have evolved to handle situated dialogues, where the robot must ground language in its immediate environment. Situated dialogue systems go beyond processing raw text: they integrate multimodal data, such as visual and spatial cues, to interpret utterances in a real-world context. For instance, in [208] systems that combine NLP with visual understanding, enabling robots to follow complex instructions like "*Take my glasses from the table near the sofa*", are introduced. Such systems use knowledge representation frameworks to link natural language with real-world semantics, as demonstrated by efforts in multi-domain dialogue systems [78].

Moreover, robots often need to learn conceptual structures for performing tasks through dialogues. Interactive dialogues can teach robots conceptual knowledge specific to a domain, such as teaching a robot the difference between tools and furniture for a cleaning task [207]. These efforts highlight the importance of using dialogue as a tool for not only interaction but also for learning.

Advancements in multimodal dialogue systems have further enhanced the capabilities to interpret human language. In [7], the authors propose that by combining speech, gesture, and vision, robots can more effectively interpret ambiguous utterances or resolve conflicts in instructions. This is especially critical in tasks where context significantly affects language interpretation, such as identifying which object to interact with when there are multiple candidates in the environment.

Lastly, ethical and social considerations in HRI cannot be overlooked. The deployment of Large Language Models (LLMs) in dialogue systems raises issues of privacy, fairness, and bias: LLMs can perpetuate gender or cultural biases, which must be addressed when designing systems for human-robot communication [195]. These challenges emphasize the need for responsible and fair AI development in HRI.

Despite significant advancements in Natural Language Processing (NLP) and Human-Robot Interaction (HRI), current systems face several limitations that hinder their application in real-world scenarios. Traditional NLP models, while effective in specific tasks, often lack the flexibility required to handle diverse linguistic inputs in dynamic environments. Moreover, many existing systems rely heavily on predefined rules or static datasets, limiting their ability to adapt to new contexts or user-specific needs. Even modern transformer-based models, although capable of generalizing across tasks, struggle with grounding language in physical environments and interpreting ambiguous or incomplete commands. These short-

comings underscore the importance of developing more adaptive models that can seamlessly integrate linguistic and multimodal information, which is a key focus of this thesis. By introducing methods for multi-task learning, multimodal integration, and efficient fine-tuning, this work aims to bridge the gap between theoretical models and practical applications in HRI. While dialogue systems provide the foundation for understanding and generating natural language, the role of semantic structures such as those captured by Semantic Role Labeling is crucial for more fine-grained interpretation of meaning. The next section explores SRL and its relevance to Situated NLU.

2.2 Natural Language Understanding: the Semantic Level

In this section, we introduce a theoretical framework for automatic language interpretation, a fundamental step toward enabling robots to comprehend human language and act accordingly in real-world environments. This involves moving beyond syntactic analysis to a semantic level, where understanding requires extracting the meaning of utterances and linking them to real-world entities, actions, and properties. Such an approach is critical for robots operating in dynamic and complex contexts, where they must infer intent, resolve ambiguities, and ground language in their perceptual and physical environment. To demonstrate this concept, we employ the task of Semantic Role Labeling (SRL), which provides a concrete example of how structured semantic understanding can facilitate effective human-robot interaction.

SRL is a critical task in natural language understanding (NLU) that involves identifying the semantic roles played by words or phrases in a sentence. It provides a structured representation of meaning by answering "who did what to whom, when, where, and why" in a given sentence. For example, in the sentence "*John gave Mary the book*", SRL identifies *John* as the agent (GIVER), *Mary* as the recipient (RECEIVER), and *the book* as the thing given (THEME). This level of semantic analysis is crucial for enabling machines to understand natural language in a way that aligns with human interpretation [88]. This aspect will be explored more in-depth in Chapter 4, particularly in Section 4.1.1.

SRL for Situated NLU. In the context of Situated NLU and Human-Robot Interaction (HRI), SRL plays a central role by linking linguistic expressions to their corresponding actions, entities, and properties in the context. For instance, consider the instruction: "*Find me the sunglasses, they should be on the table near the sofa*". Through SRL an automatic system is able to:

- Recognize *the sunglasses* as the THEME of the action `finding`;
- Identify *the table* as the SOURCE of the action;
- In case of multiple tables in the environment, disambiguate the correct one: determine the spatial relationship between *the table* and *the sofa*, consider what *near* actually means and, if these two entities satisfy the property of being near, consider it the correct place from where to take *the sunglasses*.

Such structured understanding allows robots to ground their actions in the physical world and execute tasks effectively [100, 240].

Foundational Resources for SRL. The development of SRL has been heavily influenced by annotated linguistic resources. These resources provide the semantic frames, roles, and syntactic patterns that form the backbone of SRL systems. By foundational resources, we refer to large-scale, systematically annotated datasets that serve as the basis for training and evaluating SRL systems. These resources encode essential information about how meaning is structured in language, including semantic frames, roles, and their syntactic realizations. They are indispensable for developing SRL models, as they provide the necessary data for learning how to associate linguistic expressions with their underlying meaning. Without such foundational resources, it would be impossible to build robust systems capable of performing accurate semantic analysis across a wide range of contexts. The following are some of the most prominent foundational resources that have significantly contributed to the development and advancement of SRL systems:

- **FrameNet** [23]: FrameNet organizes meaning around semantic frames, which describe prototypical situations (e.g., a GIVING frame includes roles such as GIVER, RECIPIENT, and THEME). This resource emphasizes frame-based semantics and provides a rich dataset for SRL research.
- **VerbNet** [235]: VerbNet categorizes verbs into classes based on their semantic and syntactic behaviors. It provides role labels and selectional restrictions, making it a valuable resource for linking lexical semantics with syntactic patterns.
- **PropBank** [204]: PropBank focuses on verb-centric annotations, assigning semantic roles (e.g., ARG0 for agent, ARG1 for patient) to verb arguments in specific syntactic contexts. PropBank is widely used in SRL tasks due to its broad coverage and practical applicability.

These resources collectively provide the theoretical foundation for modern SRL systems, enabling the modeling of rich semantic structures.

The Semantic Role Labeling Cascade. SRL can be viewed as a cascading process with multiple subtasks, where each subtask builds on the output of the previous one to progressively refine the understanding of the meaning in a sentence. This structured decomposition helps in isolating specific challenges within the broader task, making it easier to address individual errors and improve system performance. Below are the key subtasks involved in this cascading process:

- **Predicate Identification**: The system identifies the main predicates (typically verbs) in the sentence. For example, in "*Find me my sunglasses, they should be on the table near the sofa.*", the predicates are *find* and *be*. These predicates usually evoke the Frame that gives the meaning to the whole sentence. In this case, the predicate *to find* evokes the Frame LOCATING in FrameNet [23], while *to be* evokes, among others, the Frame BEING_LOCATED.

- **Argument Identification**: The system determines the boundaries of the arguments associated with each predicate. In the same sentence, the arguments are *me*, *my sunglasses*, *the table* and *the sofa*[1]. Usually, this step is made through a notation called BIO, where each word is annotate with a class that reflects the position of the words in a relevant span, the argument: B, for words in the beginning; I for words in the middle or at the end of a span; O for words outside of any Argument. These categories will be later used by the next step in the cascade.

- **Argument Classification**: Each contiguous series of B and I words are merged together to form an argument and each argument is assigned to a semantic role (e.g., AGENT, RECIPIENT, THEME) based on its relationship with the predicate.

This multi-step approach allows SRL systems to decompose the problem into manageable components, facilitating better error analysis and optimization [162]. At the end of the cascade, the union of all the indentifications and categorization give rise to the interpretation of the sentence in terms of structured objects, such as the Frames and Arguments. From these, automatic systems can understand natural language and infer new knowledge.

Techniques for Semantic Role Labeling. Traditional SRL systems relied on rule-based or feature-engineered methods, but the advent of deep learning has transformed the field.

[1]It could be argued that, depending on the status of the world, i.e., how the entities are disposed, *the table near the sofa* could be a single Argument. This is true and will be further explored in Section 4.1 with the GrUT approach.

In the past they were mainly based on complex semantic lexicons emphasizing via rule matching linguistic modeling and unification [177, 273], e.g., data driven methods for SRL have been also proposed either based on HMMs [47] or SVM [213], or hybrid version based on HMMs-SVMs [190, 266]. Modern SRL approaches can be broadly categorized as follows:

- **Pipeline-Based Approaches**: Early SRL systems were often divided into separate stages: syntactic parsing followed by semantic role assignment. These systems relied onto pre-trained parsers to extract syntactic structures (e.g., dependency trees) before labeling semantic roles [88]. An important limitation arises: errors in syntactic parsing propagated to the semantic role assignment stage, limiting the overall accuracy.
- **End-to-End SRL**: Recent advances have enabled SRL systems to bypass explicit syntactic parsing by using neural models. For example:
  - In [100] a deep learning model that uses BiLSTMs (Bidirectional Long Short-Term Memory networks) to perform SRL directly from raw text is introduced.
  - In [42] the SRL accuracy is improved by incorporating self-attention mechanisms, which allow the model to capture long-range dependencies in the input.
- **Pre-Trained Language Models for SRL**: Transformer-based architectures like BERT [70] and RoBERTa [163] have further advanced SRL. These models leverage contextual embeddings to improve role labeling without relying on task-specific feature engineering. Pre-trained models are often fine-tuned on donwstream SRL tasks, achieving state-of-the-art performance with minimal supervision.

Applications of SRL. SRL has wide-ranging applications in both traditional NLP tasks and emerging domains:

- **Question Answering (QA)**: SRL enhances QA systems by providing semantic representations that help match questions with their answers. For instance, in "*Where are my sunglasses?*" SRL can identify *on the table* as the answer by analyzing the Location Argument of the predicate Being_Located from the previous example [240].
- **Machine Translation**: SRL can improve the fidelity and coherence of automatic translation by ensuring compatibility between Frames and Arguments in the different languages. The Frame Semantics [83] exemplified before is language independent, and so can be used in any language.

- **Human-Robot Interaction (HRI)**: In Situated NLU, SRL helps robots ground language in physical actions, enabling them to interpret, disambiguate and understand commands and further execute tasks more effectively [208].

2.3 Adopting Machine Learning in Natural Language Processing: Transformer-based Architectures

Natural Language Processing (NLP) has undergone a profound transformation over the last few decades, evolving from rule-based systems [177, 273] and statistical methods to neural architectures [42] that define the modern era. Early approaches, such as Hidden Markov Models (HMMs) [47] and Support Vector Machines (SVMs) [213], leveraged probabilistic and optimization techniques to tackle tasks like part-of-speech tagging and text classification. However, these methods were limited by their reliance on handcrafted features and their inability to model complex linguistic patterns and long-range dependencies.

The introduction of distributed representations, such as Word2Vec [184], marked a significant milestone by embedding words in a continuous vector space, capturing semantic relationships. This innovation enabled the use of neural network architectures, particularly Recurrent Neural Networks (RNNs) and their variants like Long Short-Term Memory networks (LSTMs) [104], which modeled sequences and context dynamically. Despite their success, RNNs suffered from computational inefficiencies due to their sequential nature, making them difficult to scale to large datasets or long input sequences.

An important breakthrough came with the development of attention mechanisms, which allowed models to dynamically focus on the most relevant parts of an input sequence. First introduced in [21] for neural machine translation, attention addressed the limitations of recurrence by enabling the parallel processing of input tokens. This innovation culminated in the transformer architecture [267], which entirely replaced recurrence with self-attention mechanisms, fundamentally changing how machines process language.

This section examines the progression of machine learning approaches for NLP, culminating in the transformer architecture. The core principles of the transformer, including self-attention and multi-head attention, will be discussed, followed by an exploration of its variants and their applications in modern NLP tasks.

From Statistical Methods to Neural Architectures. Historically, statistical methods dominated NLP, with models like Hidden Markov Models (HMMs) and Support Vector Machines (SVMs) leading advancements in tasks such as part-of-speech tagging and text classification. HMMs, for instance, excelled in sequential tasks by modeling probabilistic

transitions between states. However, their inability to capture long-range dependencies in sequences constrained their effectiveness. Similarly, while SVMs were robust for classification tasks, their reliance on handcrafted features limited scalability and generalization to complex linguistic phenomena [128].

The advent of neural networks introduced distributed word representations, often referred to as word embeddings. Models such as Word2Vec [184] provided a breakthrough by mapping words into continuous vector spaces, capturing semantic relationships based on their co-occurrence in large corpora. Building on this foundation, Recurrent Neural Networks (RNNs) and their variants, such as Long Short-Term Memory networks (LSTMs) [104] and Gated Recurrent Units (GRUs) [51], offered a solution to modeling sequential dependencies. These architectures allowed NLP systems to process input in an order-sensitive manner, enabling advancements in machine translation, sentiment analysis, and more. However, their sequential nature introduced computational inefficiencies, particularly when processing long sequences.

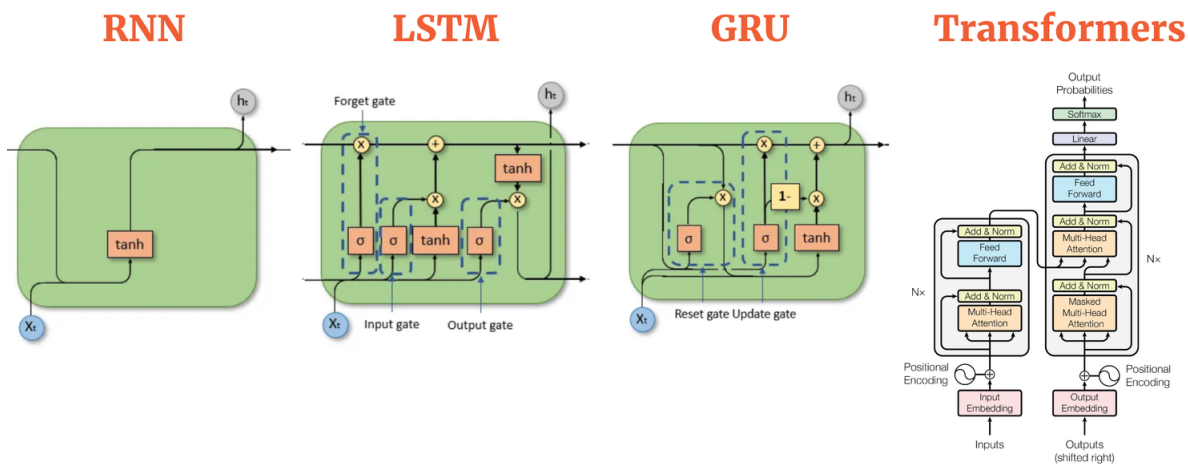


Figure 2.1: Comparing different Sequence models: RNN, LSTM, GRU, and Transformers.

The introduction of attention mechanisms addressed these limitations. First proposed in [21] for neural machine translation, attention enabled models to dynamically focus on relevant parts of an input sequence, allowing for better handling of long-range dependencies. This innovation set the stage for the development of the transformer architecture. In Figure 2.1, a conceptual diagram comparing RNNs/LSTMs and transformers, highlighting how RNNs process input sequentially while transformers process input in parallel using self-attention.

Moreover, in the RNNs architectures the word embeddings were not context-sensitive, i.e. the same word in different context had the same representation, failing in capturing different

meanings and nuances. As an example, the word *bank* had the same representation in both these sentences: "*The bank of the river*" and "*I asked the bank a loan*". It is evident that this drawback fails in disambiguating the meaning of the *bank* in such different contexts. On the other hand, Transformers introduced learnable and context-dependent word embeddings, to overcome this limitation.

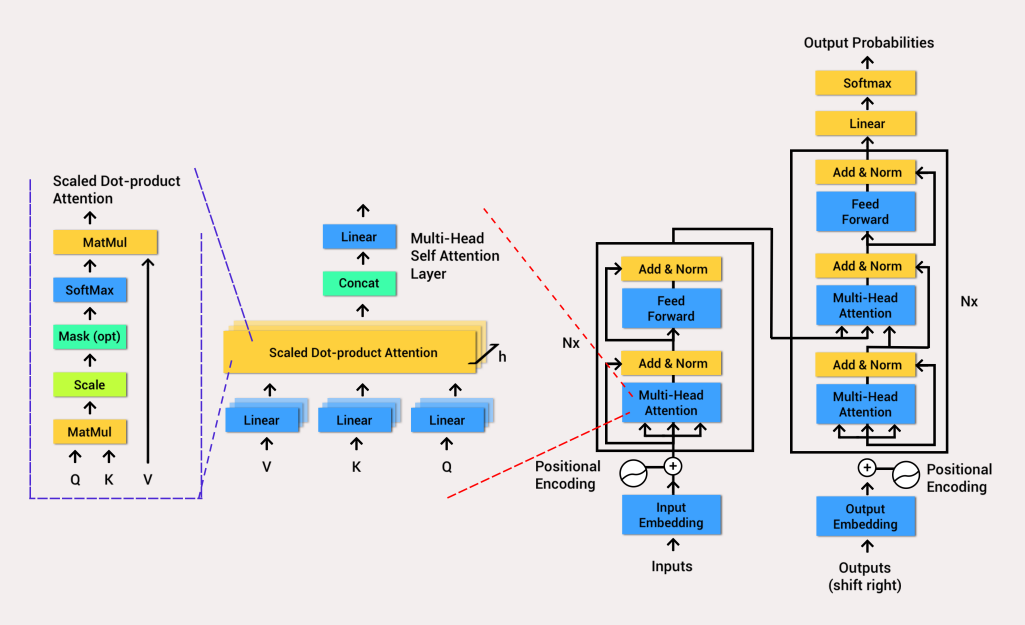


Figure 2.2: The Transformer architecture, as presented in [267], where the MultiHead Attention module is decomposed until the basic operations of multiplication and addition of the Q, K and V matrices.

The Transformer Architecture. The transformer architecture [267] (Figure 2.2) marked a turning point in NLP by replacing recurrence with a fully attention-based architecture. Unlike RNNs, which process tokens sequentially, the transformer operates on entire input sequences simultaneously. This parallelism, coupled with its ability to model dependencies across an entire sequence, significantly improved both computational efficiency and task performance. At the heart of the transformer is the **self-attention mechanism**, which computes the relationship between every pair of tokens in a sequence. This mechanism dynamically adjusts the focus of the model based on contextual relevance. For a given input, the self-attention mechanism computes attention scores using the formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

where Q, K and V represent the query, key, and value matrices, and d_k is the dimensionality of the keys. This formulation ensures that the attention scores are normalized,

allowing the model to weigh the contributions of different tokens effectively. Here, the dot product between Q and K measures relevance, scaled by $\sqrt{d_k}$ to stabilize training. The output is a weighted combination of the value vectors (V), where the weights represent the importance of each token for the given query. Unlike RNNs, which process tokens sequentially, self-attention captures long-range dependencies across an entire sequence in parallel, significantly improving efficiency and scalability.

The transformer architecture consists of an encoder and a decoder, both of which are composed of identical stacked layers. Each layer includes a **multi-head self-attention mechanism**, which enables the model to attend to multiple aspects of the input simultaneously, followed by a **position-wise feedforward network**. Residual connections and layer normalization are applied throughout the architecture to stabilize training.

In the encoder, self-attention mechanisms focus exclusively on the input sequence, generating contextualized representations for each token. In the decoder, an additional **cross-attention mechanism** allows the model to incorporate information from the encoder outputs, enabling tasks such as translation and summarization. The reliance on attention, rather than recurrence, allows this architecture to scale effectively, making it suitable for training on massive datasets. In fact, its strength and popularity comes not only from the architectural design, but from training on massive datasets as well.

Key Innovations of the Transformer. The transformer extends self-attention with *multi-head attention*, which allows the model to learn multiple perspectives of the input. Instead of computing a single attention score, the input is projected into multiple subspaces, with independent attention computations for each head. The results are concatenated and transformed via a learned weight matrix. This design enables the model to simultaneously capture different types of relationships, such as syntactic and semantic structures, enriching its representational power.

Unlike RNNs, which inherently process tokens in order, the transformer processes **all tokens in parallel**, requiring an explicit representation of their positions in the sequence. This is achieved through *positional encodings*, which are added to the token embeddings. These encodings are computed using sinusoidal functions:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right), \quad PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right) \tag{2.2}$$

where pos is the position index, i is the dimension index, and d_{model} is the embedding dimension. This approach allows the model to generalize to sequences of varying lengths while preserving order information.

A major advantage of the transformer is its ability to process **sequences in parallel**,

unlike the sequential nature of RNNs. The attention mechanism operates on all tokens simultaneously, enabling efficient computation and better hardware utilization. This parallelism reduces training time significantly and allows transformers to scale effectively to larger datasets and models. It has been key to the development of large language models, such as GPT and BERT, which are trained on massive corpora.

Each sublayer of the transformer incorporates **residual connections** and **layer normalization**. Residual connections enable the model to retain information from earlier layers, facilitating gradient flow, while layer normalization ensures stable activations. These mechanisms are critical for scaling the transformer to deep architectures without degradation in performance.

Moreover, in transformer-based architectures, **word embeddings** are **learnable** and **context-dependent**, meaning they are dynamically updated based on the surrounding context of a word within a given input sequence. Unlike static embeddings, these embeddings are generated through a self-attention mechanism, where each token attends to all other tokens in the sequence. This allows the model to produce unique embeddings for the same word depending on its context, effectively capturing polysemy (multiple meanings) and nuanced word relationships. These embeddings are learned during pre-training on large corpora and fine-tuned on downstream tasks. In contrast, classical models like RNNs and LSTMs often used static embeddings such as Word2Vec, where each word is represented by a fixed vector regardless of context. Word2Vec embeddings are pre-trained using techniques like skip-gram or CBOW (Continuous Bag of Words) and remain unchanged during downstream task training. While these embeddings can capture global word similarities based on co-occurrence statistics, they fail to differentiate between different meanings of a word in varying contexts. Transformers overcome this limitation by dynamically updating word embeddings through attention mechanisms, providing richer and more precise contextual representations. This distinction makes transformer-based models significantly more powerful for tasks requiring deep semantic understanding.

These innovations collectively transformed the field of NLP, making the transformer the architecture of choice for tasks like machine translation, text summarization, and text generation. By replacing recurrence with attention, the transformer achieves state-of-the-art results across a wide range of applications, cementing its role as the backbone of modern NLP systems.

Architectural variants of the Transformer. The transformer architecture, as originally proposed [267], consists of an encoder-decoder framework designed for sequence-to-sequence tasks, such as machine translation. Over time, this architecture has been adapted into three distinct configurations: encoder-only, decoder-only, and encoder-decoder models. Each

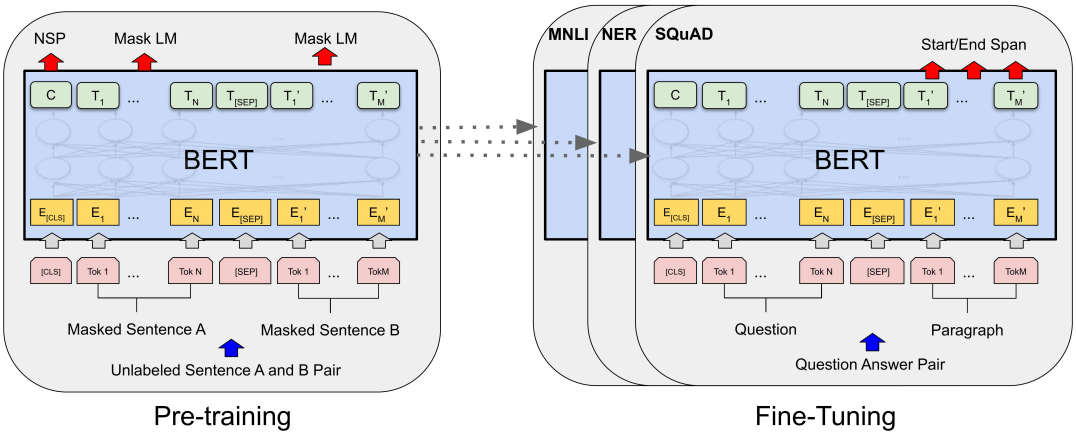


Figure 2.3: The BERT [70] architecture, with a specific focus on the pre-training (left) and fine-tuning (right) process and tasks.

variant is optimized for specific types of NLP tasks, with unique design features tailored to their respective objectives.

Encoder-only models are designed to process and understand input sequences. These architectures consist solely of a stack of encoder layers, where each layer applies self-attention and feedforward transformations to generate contextualized representations of the input. Encoder-only models are particularly effective for tasks that require comprehensive understanding of the input, such as text classification, named entity recognition, and sentiment analysis. The key innovation of encoder-only models is their use of bidirectional attention, which allows the model to attend to all tokens in the input sequence simultaneously. For example, BERT (Bidirectional Encoder Representations from Transformers) [70] processes input left-to-right and right-to-left by considering both preceding and succeeding tokens, capturing richer contextual information compared to unidirectional approaches. This bidirectionality makes encoder-only models highly effective for tasks where understanding the full context of the input is critical. In Figure 2.3, the BERT model is first pre-trained on a large amount of data to learn what language is, then, on the right, a specific focus on linguistic downstream tasks, where the model has been adapted to solve them.

While the original BERT model [70] set the standard for encoder-only transformer architectures, subsequent advancements such as RoBERTa (Robustly Optimized BERT Approach) [163] introduced refinements that significantly enhanced performance. RoBERTa improves upon BERT by optimizing its pretraining methodology: it removes the next-sentence prediction (NSP) task, trains on larger datasets, increases the number of training steps, and uses larger batch sizes. These changes allow RoBERTa to learn more robust representations, making it one of the most widely adopted models for understanding tasks like classification, named entity recognition (NER), and sentiment analysis.

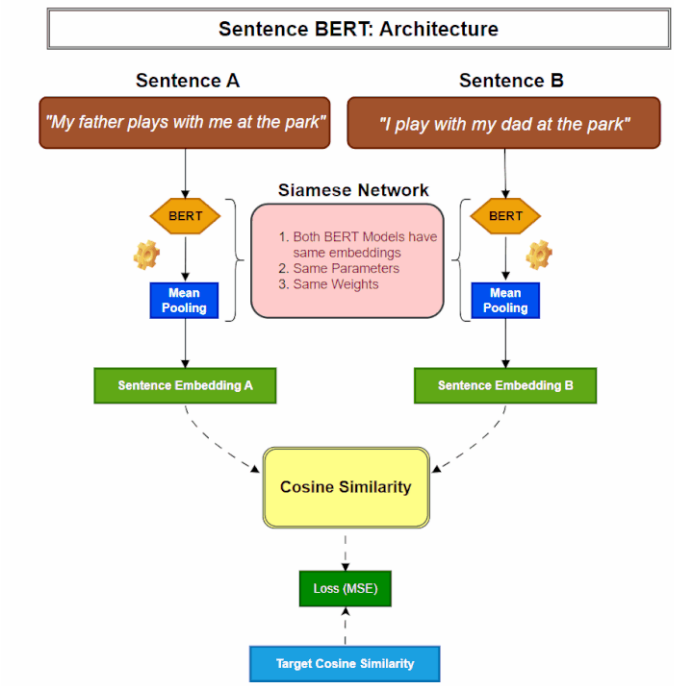


Figure 2.4: The sBERT [220] architecture, where two similar sentences are encoded through the Siamese networks and compared using the Cosine Similarity measure.

In the context of **Encoder-only** model for **the Italian language**, ALBERTo [211] represents a significant adaptation of encoder-only transformer architectures. ALBERTo is a language model specifically trained on Italian tweets, making it optimized for social media text, which is typically informal and often unstructured. This model is pre-trained on large-scale Italian Twitter datasets, capturing nuances such as slang, abbreviations, and hashtags that are common in social media communication. ALBERTo has been particularly useful for tasks such as sentiment analysis and user profiling in Italian-language datasets. Its specialization in informal text complements more general models like multilingual BERT or XLM-R [58], which are trained on more diverse, but often formal, corpora.

While standard BERT and its variants like RoBERTa and ALBERTo are effective at generating contextualized representations for tokens, they are not inherently optimized for sentence-level tasks such as **semantic similarity** or clustering. To address this limitation, Sentence-BERT (sBERT) [220] was introduced as a modification of BERT specifically designed for sentence embeddings. sBERT uses a Siamese network structure to derive fixed-length representations of entire sentences, which can then be compared using standard similarity measures, such as cosine similarity, as depicted in Figure 2.4. sBERT is particularly useful for tasks such as paraphrase detection, semantic textual similarity (STS), and information retrieval. For example, in multilingual and Italian-specific contexts, sBERT has

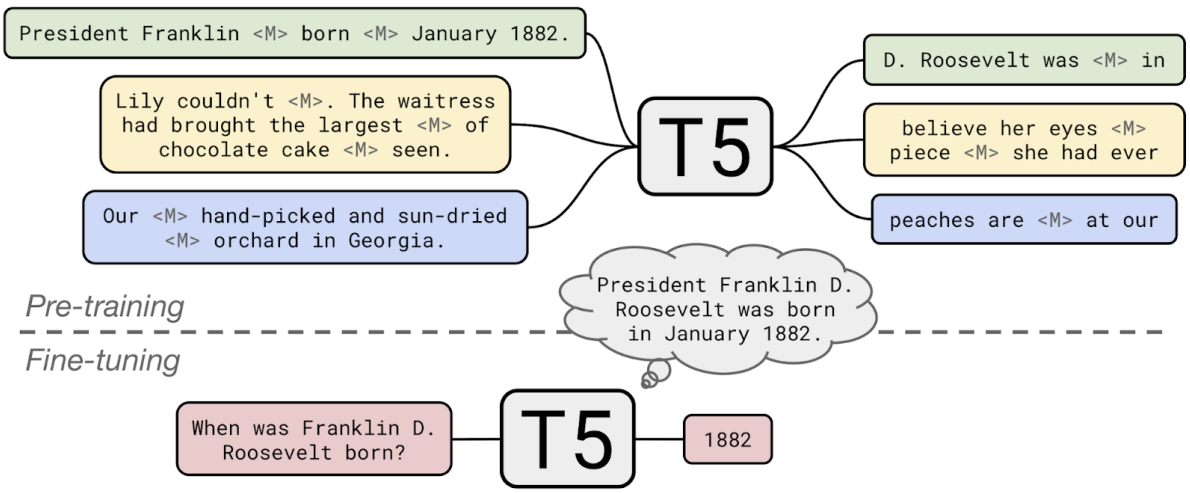


Figure 2.5: *Top*: The T5 [217] pre-training strategy, where each task is transformed into a text-to-text. *Bottom*: At inference time, the T5 model answers directly by using its internal knowledge.

been fine-tuned on datasets such as STS-B and multilingual parallel corpora to create representations that capture cross-lingual semantic equivalence. Its ability to encode sentences efficiently has made it a foundational model for downstream applications, including question answering, search ranking, and clustering.

The **Encoder-Decoder** configuration consists of the union of the two components, and these models are designed for sequence-to-sequence tasks, such as machine translation, summarization, and question answering. For example, T5 (Text-to-Text Transfer Transformer) [217] and BART (Bidirectional and Auto-Regressive Transformers) [152] utilize this architecture to achieve state-of-the-art results in diverse NLP applications. The combination of bidirectional attention in the encoder and autoregressive decoding allows encoder-decoder models to leverage both global context and sequential dependencies.

One of the key strengths of T5 lies in its innovative approach of recasting all NLP tasks into a unified *text-to-text* format, as shown in Figure 2.5. Traditionally, tasks such as classification or regression required designing task-specific architectures or adapting model outputs. In contrast, T5 reformulates these tasks as text generation problems. For example, a classification task such as sentiment analysis can be reformulated by providing input in the format "`sentiment: The movie was excellent`" and expecting output such as "`positive`". Similarly, a regression task like rating prediction is treated as text generation, with the model outputting numerical ratings as strings. This unified framework not only simplifies training pipelines but also enables seamless multi-task learning, allowing T5 to excel in diverse NLP applications ranging from translation to summarization, question answering, and classification. BART, on the other hand, distinguishes itself with a focus on text reconstruction.

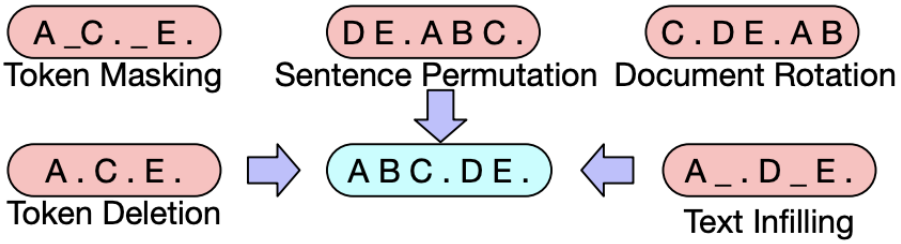


Figure 2.6: The BART [152] pre-training strategy, each task serves as a denoising strategy.

During pretraining, various noising functions are applied to the input text (e.g., token masking, sentence shuffling, as in Figure 2.6) and then BART is trained to reconstruct the original text. This denoising objective enables BART to become highly effective at rewriting tasks such as paraphrasing, sentence fusion, and style transfer. Additionally, its ability to recover syntactically and semantically coherent text from noisy inputs makes it an excellent choice for text generation tasks like summarization. The autoregressive nature of its decoder further enhances its capabilities in generating fluent and coherent output.

Focusing on **Italian specific Encoder-Decoder models**, two architectures stand out: IT5 [231] and BART-IT [142]. IT5 is an Italian adaptation of T5, pretrained on large-scale Italian corpora including datasets from legal, medical, and literary domains. It retains the text-to-text approach of the original T5 model, while specializing in Italian-specific tasks such as named entity recognition (NER), sentiment analysis, and machine translation. For instance, IT5 has demonstrated state-of-the-art performance in tasks like paraphrasing Italian sentences or answering questions about Italian literature [231]. BART-IT [142], an Italian adaptation of BART, follows a similar philosophy by pretraining the model with noising and text reconstruction objectives on diverse Italian corpora. This makes it particularly adept at rewriting and summarization tasks in Italian. For example, it can effectively summarize long Italian legislative texts into concise, readable formats or paraphrase Italian tweets for sentiment analysis [142]. Together, IT5 and BART-IT highlight the adaptability of encoder-decoder models to specific languages and their potential to drive advancements in language-specific NLP applications.

In the **multilingual** space, models like mT5 [282] and flan-T5 [54] extend the capabilities of encoder-decoder architectures to handle multiple languages simultaneously. mT5 is a multilingual variant of T5 pretrained on a diverse set of 101 languages, including Italian. This extensive multilingual pretraining allows mT5 to perform exceptionally well in cross-lingual tasks such as translation, multilingual question answering, and zero-shot learning. For instance, mT5 can answer questions in Italian based on context provided in English or translate between low-resource languages without explicit supervision.

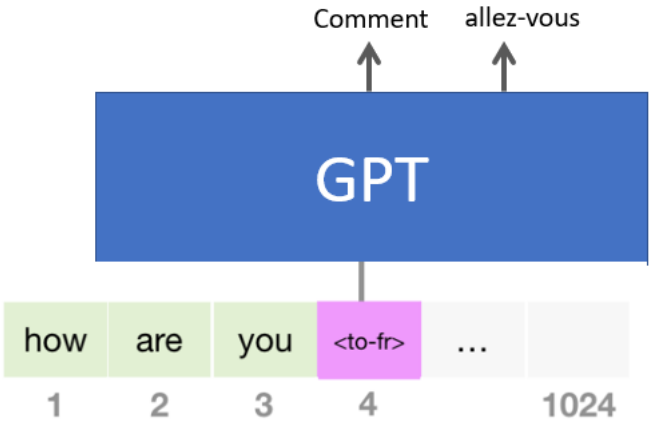


Figure 2.7: The Autoregressive nature of GPT [216] that generates one "token" at a time, taking in input the original input and the generated tokens until the previous step.

flan-T5 [54], on the other hand, pushes the **encoder-decoder** paradigm to its limits by **incorporating instruction** tuning during pretraining. Unlike standard T5, which relies primarily on task reformulation, flan-T5 is explicitly trained on a broad set of instruction-following datasets. This makes it exceptionally proficient in zero-shot and few-shot learning settings. For example, flan-T5 can be prompted to summarize complex multilingual documents, perform reasoning tasks, or follow user-specific instructions in multiple languages, including Italian [54]. Its instruction-tuned nature allows it to adapt to a wide range of user queries with minimal additional training.

Decoder-only models, on the other hand, are specialized for generative tasks, where the goal is to produce a sequence of outputs. These architectures consist of a stack of decoder layers, which include self-attention and feedforward layers, as well as an additional causal masking mechanism. The causal mask ensures that each token in the output sequence can only attend to previous tokens, preserving the left-to-right dependency required for generation. A notable example of a decoder-only model are GPT (Generative Pre-trained Transformer) [216] and LLaMA [261], which predict the next token in a sequence based on the preceding context, as in Figure 2.7. This autoregressive behavior enables decoder-only models to excel in tasks such as text generation, dialogue systems, and code generation. However, the unidirectional nature of decoder-only models limits their ability to fully model bidirectional relationships, making them less suitable for tasks requiring deep understanding of the input. The recent advent of Large Language Models has been entirely based on this variation, i.e., the Decoder-only models, with a proliferation in the last month of models released almost daily.

2.4 Large Language Models

Large Language Models (LLMs) represent a significant leap forward in Natural Language Processing (NLP), characterized by their massive scale and ability to perform diverse tasks. LLMs are typically defined as models containing **billions of parameters**, which enable them to capture complex linguistic patterns and relationships in text. These parameters, learned during extensive pretraining on large corpora, provide LLMs with a generalized understanding of language that can be adapted to numerous applications without requiring extensive task-specific fine-tuning.

The transformative potential of LLMs became evident with the introduction of GPT (Generative Pre-trained Transformer) [216], which demonstrated the **generative power** of decoder-only transformer architectures. Its autoregressive design enabled it to generate coherent and contextually appropriate text by predicting the next token in a sequence. This breakthrough established LLMs as a foundational tool for text generation tasks.

Subsequent advancements in LLMs, such as GPT-3 [39], highlighted the role of scale in enhancing model capabilities. GPT-3, with its 175 billion parameters, demonstrated that **increasing the size of LLMs** significantly improves their performance across a wide range of linguistic tasks, even in few-shot or zero-shot learning scenarios. Without fine-tuning, GPT-3 achieved state-of-the-art results on tasks such as translation, summarization, and question answering, illustrating the emergent abilities of LLMs as they grow in size [39].

With the release of ChatGPT [199, 200], LLMs gained widespread attention from the general public. ChatGPT showcased impressive conversational abilities, combining fluency, coherence, and context-awareness in dialogue. Its success demonstrated how instruction-tuned LLMs could be fine-tuned to specialize in interactive tasks, making them accessible and useful for everyday applications.

While GPT models demonstrated the capabilities of proprietary systems, the LLaMA (Large Language Model from Meta AI) series [261, 262] marked a shift towards **openness and accessibility**. Released as open-source architectures, LLaMA and LLaMA 2 enabled researchers to study, fine-tune, and adapt LLMs without the constraints of proprietary licensing. These models became widely used in academic and industrial research, fostering innovation and experimentation in the NLP community.

Despite their remarkable achievements, LLMs are not without limitations. They are prone to issues such as biases, ethical concerns, hallucinations, and a lack of interpretability. Moreover, certain tasks, such as mathematical reasoning and counting, reveal fundamental weaknesses in their design. For instance, LLMs struggle with solving arithmetic problems, as shown in benchmarks like MultiArith [223]. Recent efforts, such as MathPrompter [123],

have sought to address these shortcomings by augmenting LLMs with external tools like Python code execution to ground their reasoning and improve accuracy on mathematical tasks.

In the following subsections, we will explore the evolution of LLMs in greater detail, focusing on how increasing model size and extensive pretraining have contributed to their capabilities. Additionally, we will examine key concepts such as prompt engineering, which allows users to exploit LLMs without task-specific fine-tuning, and instruction-tuning, which has paved the way for conversational models like ChatGPT. By tracing the progression of LLMs, we aim to highlight the factors that have driven their success and identify the challenges that remain in advancing these transformative technologies.

2.4.1 Foundational Models: the race to more and more parameters

The rapid advancement of Large Language Models (LLMs) has been driven by an ongoing pursuit of scale, where increasing the number of model parameters has consistently resulted in enhanced performance across a variety of linguistic tasks. Foundational models such as GPT [216], GPT-3 [39], and the LLaMA series [261, 262] exemplify this trend, highlighting how architectural improvements combined with greater scale have propelled NLP to new heights.

The release of GPT (Generative Pre-trained Transformer) [216] introduced a decoder-only transformer architecture that demonstrated the generative potential of LLMs. It was the first architecture that deviated from the original Transformers, then BERT [70] was released as a contrapposition. In any case, by employing an autoregressive approach, GPT generates text token by token, predicting each word based on the preceding context. This autoregressive nature enables the model to produce coherent and contextually appropriate text, making it suitable for a wide range of tasks, such as text generation, dialogue, and story writing, as shown in Figure 2.7. Its success stems from two key factors:

- **Pretraining on Large Text Corpora**: GPT was trained on vast amounts of unstructured text, allowing it to learn linguistic patterns, grammar, and semantic relationships across a variety of contexts.
- **Transfer Learning for Downstream Tasks**: Once pretrained, GPT could be fine-tuned[2] on specific tasks, leveraging its general language understanding to achieve high performance with relatively small amounts of task-specific data.

[2]Assuming some hardware requirements, such as access to enough powerful GPUs for enough time.

The release of GPT-3 [39] marked a turning point in the development of LLMs, demonstrating that scaling the number of parameters leads to emergent capabilities. With 175 billion parameters, GPT-3 is over 100 times larger than its predecessor, GPT-2. This massive increase in scale enabled the model to achieve state-of-the-art performance on a diverse set of linguistic tasks, even in zero-shot and few-shot settings. Without requiring explicit task-specific fine-tuning, GPT-3 could generalize to new tasks by conditioning on examples provided in the input prompt. The effectiveness of scaling in GPT-3 can be attributed to:

- **Capacity to Memorize and Generalize**: Larger models have a greater capacity to store complex patterns from training data, enabling them to generalize more effectively to unseen contexts.
- **Emergent Abilities**: Scaling the number of parameters led to qualitative improvements, such as coherent text completion, reasoning over multiple paragraphs, and performing arithmetic, tasks that smaller models struggled with.
- **Zero-Shot and Few-Shot Learning**: By providing examples in the prompt, GPT-3 could perform tasks it was not explicitly trained on, such as translating text or summarizing documents, showcasing its flexibility.

While GPT and GPT-3 demonstrated the power of proprietary LLMs, the LLaMA (Large Language Model Meta AI) series [261, 262] introduced open-source alternatives that democratized access to foundational models. LLaMA models were designed to be efficient, providing competitive performance with significantly fewer parameters compared to proprietary models. For instance, LLaMA-2 achieved results comparable to GPT-3, despite having fewer parameters, thanks to architectural refinements and highly optimized pretraining. The LLaMA series emphasized accessibility and reproducibility:

- **Smaller Scale with Comparable Performance**: By focusing on efficient training and tokenization strategies, LLaMA demonstrated that high-quality language models do not necessarily require billions of parameters to perform well.
- **Open-Source Availability**: LLaMA and LLaMA-2 enabled researchers to experiment with large-scale architectures without the restrictions of proprietary licensing, fostering innovation across academic and industrial domains.

The consistent improvement of LLMs with increasing parameters can be explained by several factors:

- **Representation Power**: Larger models can capture more intricate patterns in the training data, enabling them to better understand complex linguistic relationships and generate more nuanced text.
- **Smoother Optimization Landscapes**: As the parameter count increases, the optimization process becomes more stable, reducing the likelihood of local minima and enabling better convergence during training.
- **Emergent Abilities**: Scaling unlocks qualitative improvements that smaller models cannot achieve. For example, GPT-3 demonstrated advanced reasoning, multi-hop question answering, and logical inferences that smaller models failed to replicate.
- **Task Generalization**: With more parameters, LLMs can generalize across a wider range of tasks, often requiring only minimal prompts or fine-tuning to adapt to new challenges.

Despite their remarkable benefits, LLMs are particularly prone to challenges such as bias and hallucination, as they often amplify patterns and imperfections present in their training data. These issues raise significant ethical concerns, emphasizing the importance of carefully balancing trade-offs between model size, computational efficiency, and fairness. Ethical considerations become even more pressing as LLMs are deployed in real-world applications, where unintended biases or hallucinated content can lead to misinformation or harmful outcomes. Furthermore, the exponential growth in the number of parameters introduces significant technical challenges, including dramatically higher computational costs, increased memory requirements, and extended training times. For example, while training traditional machine learning models, such as Support Vector Machines (SVMs), could often be completed in seconds or minutes on a standard laptop, and LSTMs required only modest computational resources, the advent of LLMs has drastically shifted these demands. Transformer-based architectures rely heavily on parallelization, necessitating access to powerful GPU clusters. Models like GPT-3 reached a scale that placed their training far beyond the reach of most academic researchers and smaller institutions, further highlighting the increasing divide in access to cutting-edge AI technologies. Interestingly, GPT-3 [39] also introduced the ability to use LLMs in few-shot settings, a technique often referred to as In-Context Learning. This approach enables LLMs to adapt to new tasks without requiring explicit task-specific fine-tuning, showcasing the versatility of these models despite their immense computational demands.

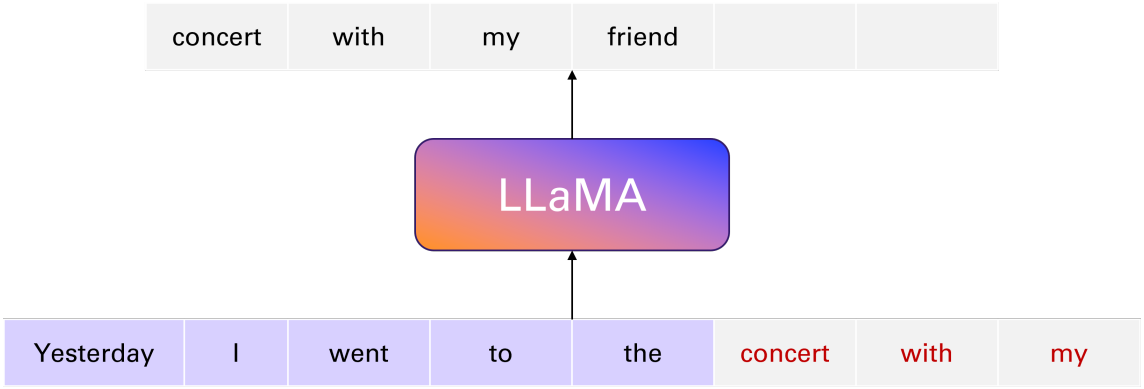


Figure 2.8: The LLaMA [261] Autoregressive decoding process, where each output token at previous steps are input for the next steps.

2.4.2 In-Context Learning and Prompt Engineering

In-Context Learning (ICL) represents a groundbreaking capability of large language models (LLMs), first demonstrated at scale by GPT-3 [39]. Unlike traditional machine learning methods, where models are explicitly fine-tuned on downstream tasks, ICL allows models to generalize to unseen tasks by conditioning on input prompts that include both task instructions and a small number of task-specific examples [154]. It is referred to as "in-context" because the model performs the task based on examples provided within the input context, without any updates to its weights. GPT-3 showcased state-of-the-art performance across numerous benchmarks, including translation, question answering, and summarization, solely by leveraging this novel capability [39].

To use in-context learning, the user constructs a prompt consisting of examples of input-output pairs (shots) and a query that the model is required to complete. For instance, to classify the sentiment of a sentence, the input might look like:

1. *"I love this movie!"* → *Positive*
2. *"The weather is terrible today."* → *Negative*
3. *"This book is fantastic!"* → *Positive*
4. *"I didn't enjoy the meal."* → *Negative*
5. *"The service at the hotel was great!"* →

The model then generates the output "`Positive`", leveraging the patterns observed in the preceding examples. The input in ICL is thus a carefully crafted combination of examples, while the output is the prediction based solely on this context [74, 275], as a completion method, as shown in Figure 2.8.

In-context learning fundamentally differs from traditional fine-tuning, which involves updating its weights by training the model on task-specific labeled data, requiring computa-

tional resources and additional training steps. In contrast, ICL operates entirely at inference time [185], with the weights remaining unchanged. This makes ICL particularly useful for scenarios where training data is scarce [185] or where flexibility across multiple tasks is required [280]. However, because ICL relies solely on the input prompt, its effectiveness is contingent on the quality and structure of the provided examples [161], often requiring careful prompt engineering [276].

A critical factor in ICL is the number of examples (K) included in the prompt, referred to as the number of "shots" [185]. In a *few-shot* setting, the model is provided with several examples of input-output pairs to infer the task. For instance, in the example above, $K = 4$. When only one example is given ($K = 1$), the setting is termed *one-shot*, while providing no examples and relying solely on the internal knowledge of the model defines the *zero-shot* setting. The choice of K depends on the capacity of the model and the length constraints of the input. As K increases, the model generally achieves better results by leveraging more contextual examples [39, 276]. However, excessively high values of K can lead to diminishing returns [39] due to input length limitations and the inability of the model to effectively parse and utilize overly long prompts. Determining an optimal K is thus task-specific, often requiring experimentation to balance the trade-off between context length and performance, as shown in [39].

Even with a fixed number of shots (K), results can be improved through the careful design of the input prompt, a practice known as *prompt engineering* [185]. It involves rephrasing task instructions, selecting high-quality examples, or reorganizing the structure of the prompt to maximize the performance [39, 185, 276]. For instance, ordering examples from the most to the least representative of the task or from the simplest to the hardest, or framing the query in natural language, can significantly enhance the output quality. An example of rephrasing a classification task prompt is:

Task: Classify the sentiment of the following sentences by labeling them as Positive or Negative.

Such changes help in guiding the model to better understand the task, often yielding more accurate results. Prompt engineering is an active area of research [161, 228, 268], as small changes to the prompt can lead to disproportionately large improvements in model performance [165].

When all examples (K) are eliminated from the prompt and only the task description remains, in-context learning transitions to what is known as *instruction-based learning* [230, 254]. Here, the prompt serves solely to describe the task, relying on the ability to follow explicit instructions. For example:

Classify the sentiment of this sentence: "I love this movie!"

This approach underpins the transition to instruction-tuned models [255, 254, 230, 20, 28, 212], where models are explicitly asked to follow task-specific instructions across a wide range of applications, even though sometimes it still necessitates fine-tuning [255, 119, 69]. Instruction tuning is a critical development that has led to conversational models like ChatGPT [199], which specialize in understanding and responding to human-provided instructions. The next chapter will explore instruction tuning and how it has shaped the capabilities of modern LLMs.

2.4.3 Instruction tuning for LLMs

Instruction tuning represents a pivotal step in the evolution of Large Language Models (LLMs), building on the foundational capabilities introduced by In-Context Learning (ICL). While ICL enables models to perform tasks by conditioning on prompts that include instructions and examples, it relies entirely on the pre-trained knowledge and does not involve altering the parameters of models. Instruction tuning, by contrast, involves fine-tuning LLMs on datasets explicitly designed to teach them how to follow human-provided instructions across a diverse range of tasks. This process creates models that are inherently better at understanding and executing user instructions without requiring elaborate prompts or examples. The input construction shifts from completion-based prompts to detailed instructions: as an example, for Sentiment Analysis tasks, with instruction tuning we describe to the model what we want it to perform, so the shift goes from "*The sentiment of* <INPUT> *is:* " to "*Please tell me what sentiment the following sentence evokes, choose between Positive, Negative and Neutral.* <INPUT>".

Moreover, instruction-tuned models shift the paradigm from inference-time adaptation to task-specific generalization through fine-tuning, ensuring a deeper alignment with human intentions. Instruction-tuning generally yields more reliable and consistent performance compared to zero-shot or few-shot in-context learning because it explicitly aligns the internal representations of the model with the desired task behaviour during fine-tuning. While in-context learning leverages large prompt examples to elicit task-specific responses, it can be less predictable, especially when generalising to diverse tasks or inputs. Instruction-tuned models, by contrast, benefit from learning task-specific patterns across a broad set of instruction-response pairs, making them more sample-efficient, better generalised, and less sensitive to prompt design. This can be critical when deploying models for specialised tasks or under resource constraints. As an example of an instruction-response pair, consider the task of Sentiment Analysis: the instruction could be "*Please determine the sentiment*

of the following sentence and classify it as Positive, Negative, or Neutral:" followed by the specific input sentence. The corresponding response would then be one of the three sentiment labels, i.e. the response is expected to be one of *Positive, Negative, or Neutral*. This example highlights how instruction tuning shifts the interaction paradigm from simple text completion to explicit task-oriented communication, ensuring that the model comprehends and executes the task based on clear human-provided instructions. This distinction is particularly evident in the development of open-source instruction-tuned models like Alpaca [254], which marked a milestone as one of the first publicly available instruction-following models based on LLaMA [261].

LLaMA (Large Language Model from Meta AI) [261] introduced a series of highly efficient and performant LLMs, trained with a focus on accessibility and reproducibility. By scaling parameters while maintaining computational efficiency, LLaMA models achieved results comparable to proprietary models like GPT-3, despite having fewer parameters. However, the original LLaMA models were not explicitly instruction-tuned and thus lacked the fine-tuning necessary for robust instruction-following capabilities. In essence, they were still Large Decoder-only models based on completion. Alpaca [254] extended LLaMA by fine-tuning it on a dataset of task instructions and responses derived from the `text-davinci-003` model from OpenAI. Using approximately $52,000$ synthetic instruction-response pairs, Alpaca demonstrated impressive performance in instruction-following tasks, becoming the first open-source instruction-tuned model. The significance of Alpaca lies not only in its capabilities but also in its accessibility, as it enabled researchers and practitioners to explore instruction-tuning without reliance on proprietary systems. This advancement paved the way for further experimentation and adaptation of LLMs to specific languages and domains.

In addition to general-purpose instruction-tuned models like Alpaca, recent efforts have focused on developing instruction-tuned LLMs tailored to specific languages. For the Italian language, notable contributions include Camoscio [230], Fauno [20], LLaMAntino [28], and its advanced version, LLaMAntino-3-ANITA [212]. These models address the unique linguistic and cultural nuances of Italian while leveraging the benefits of instruction tuning to achieve high performance across various NLP tasks. Camoscio [230] is one of the most prominent instruction-tuned LLMs designed specifically for the Italian language. It is based on the LLaMA architecture and fine-tuned using the Alpaca dataset translated in Italian, including task-specific examples, enabling it to perform a wide range of tasks, from translation to summarization and question answering. In addition, Camoscio employs a refined prompt engineering strategy during its instruction tuning phase, ensuring that the model generalizes effectively across various instructions while minimizing bias and inaccuracies. The performance of Camoscio on Italian benchmarks has demonstrated its robustness in both general

and domain-specific NLP tasks, setting a new standard for Italian LLMs.

While instruction tuning focuses on aligning LLMs to follow instructions effectively, language adaptation emphasizes adapting a model to a specific language by pretraining or fine-tuning on language-specific datasets. For instance, Camoscio [230] and Fauno [20] integrate instruction tuning with language adaptation, creating models that not only excel at following instructions but also address the unique challenges posed by Italian morphology, syntax, and semantics. The distinction between instruction tuning and language adaptation [27] explore the impact of language-specific pretraining on the performance of instruction-tuned models. While instruction tuning enhances a ability of a model to generalize across tasks, language adaptation ensures that the model is deeply attuned to the linguistic nuances of the target language. Combining these approaches results in models like Camoscio and LLaMAntino-3-ANITA, which set benchmarks for Italian LLMs by addressing both generalization and language-specific requirements.

Although LLMs excel in text-only tasks, they face significant limitations when applied to Situated Human-Robot Interaction, where understanding language requires grounding it in a physical environment. In HRI scenarios, robots must interpret linguistic instructions in conjunction with visual information, such as recognizing objects, understanding spatial relations, and responding appropriately to changes in the environment. Traditional LLMs, trained solely on textual data, lack the capability to perceive and reason about the external world, making them insufficient for tasks that demand situated understanding. This is where Multi-Modal LLMs become highly relevant to this work. By integrating both visual and textual inputs, these models enable grounded interactions, allowing robots to link language to their perceptual context. Such multi-modal capabilities are essential for bridging the gap between abstract language representations and real-world execution, a core challenge in this thesis. The following section explores multi-modal neural learning architectures, which form the basis for extending LLMs to handle complex, situated dialogues in dynamic environments.

2.5 Multi-Modal Neural Learning Architectures

Multimodal input signals enable virtual or physical agents to perceive and interact with their environments in more meaningful ways. A cornerstone of this field lies in the development of advanced computer vision techniques. Early approaches relied heavily on handcrafted features and traditional machine learning, but the advent of deep learning revolutionised the field with architectures such as Convolutional Neural Networks (CNNs) [150]. These models, exemplified by AlexNet [139], VGG [241], and ResNet [99], excel at extracting hierarchical

features from images through convolutional layers. They apply learnable filters that capture local features through convolutional operations. Stacking multiple layers enables hierarchical feature extraction, from edges to complex patterns, while pooling layers reduce spatial dimensions, improving efficiency and invariance to transformations. Optimised via backpropagation, CNNs excel in tasks like image classification and segmentation by effectively modelling spatial hierarchies. However, CNNs struggle with capturing long-range dependencies in visual data: their convolutional layers focus on local regions, relying on deeper layers to aggregate broader context. This hierarchical approach makes it harder to model direct relationships between distant parts of an image. To address these limitations, Visual Transformers were introduced. Inspired by the success of Transformers in natural language processing, Vision Transformers (ViTs) [75] use self-attention mechanisms to model global dependencies in images. By splitting an image into patches and treating each patch as a token in a sequence, ViTs allow for a more comprehensive understanding of both local and global image features. This approach is particularly effective for tasks requiring long-range dependencies and holistic context, such as image classification and segmentation. These models marked a significant shift in computer vision, enabling more sophisticated representations of visual data.

One increasingly explored area in this domain is the conjunction of text and images. A seminal contribution to this field is CLIP [215], an architecture that takes a text and an image as input and originally produces a similarity score between the two. The architecture is optimised using a contrastive learning schema, which defines a specific contrastive loss: this is minimised if the text accurately describes the image and maximised if the two inputs represent entirely different topics. The strength of CLIP lies in its ability to bring similar images and texts closer in a shared latent space while pushing dissimilar pairs far apart. This is achieved in a supervised manner, meaning that the training dataset must be annotated. Building upon CLIP, BLIP [155] was introduced to bootstrap the architecture from a pre-trained model in an unsupervised fashion, leveraging web image data along with their captions. This approach eliminates the need for large amounts of annotated data. Both CLIP and BLIP, however, depend on a global comparison of image-text similarities across a batch, even if the pairs are not directly related. As a solution, Sigmoid loss for Language-Image Pre-training (SigLIP) [286] was developed. Unlike the contrastive learning employed by CLIP, which uses Softmax normalisation and relies on global pairwise similarity comparisons, SigLIP introduces a sigmoid loss that operates only on individual image-text pairs. This approach significantly enhances performance while reducing the dependence on large batch sizes for normalisation. A major limitation of contrastive loss in CLIP is its assumption of a single correct image-text pairing for each example. However, in real-world

scenarios, images and captions often have multiple plausible associations. Sigmoid loss addresses this by computing binary cross-entropy for each potential image-text pair, framing the problem as multi-label classification. This allows the model to score multiple relevant image-text pairs, rather than forcing a single correct match. As a result, Sigmoid loss enables smoother and more flexible alignment between images and texts, effectively handling cases where an image may correspond to several valid captions, or a caption may describe multiple images.

These models are specifically designed to learn optimal representations of both modalities: vision and text. Recently, architectures such as LLaVA [160], CogVLM [274], and CogAgent [106] have demonstrated effective approaches for integrating CLIP and/or BLIP with a Large Language Model, often based on variants of LLaMA [262], to tackle complex inference tasks. These tasks range from generating detailed image descriptions using ad hoc prompts, to explaining visual memes encountered online, and performing visual grounding by identifying entities in images through bounding boxes. Such models are of particular interest to us, as they offer the potential to equip Intelligent Agents with the capability to seamlessly integrate vision and language, resulting in more informed decision-making processes based on both visual and textual inputs. Specifically, I envision employing a similar architecture for the Intelligent Robot, as outlined in the Introduction.

2.5.1 Recent Vision and Language Models

LLaVA [160] is an end-to-end multimodal model that connects a vision encoder (CLIP [215]) with a Large Language Model (Vicuna [50]) for general-purpose visual and language understanding. At this point, one challenge arises: how do we reconcile the encoding of images with the encoding of text? To address this, a third component is needed: specifically, a single-layer MLP, known as the Projector, that maps the output of the visual encoder into the input space of the LLM. This alignment is achieved using a subset of the CC3M dataset, focusing on image-text pairs to map visual inputs into the embedding space of the language model, thereby enabling the model to comprehend and generate contextually relevant textual responses based on visual data. This alignment allows the model to effectively fuse visual features with textual prompts, enabling seamless integration of information from both modalities. This pre-training stage is crucial as it establishes a shared embedding space between the visual and textual modalities, facilitating effective multimodal understanding. By keeping the vision encoder and language model weights frozen and updating only the projection matrix, LLaVA efficiently learns to integrate visual information into the language model, setting the foundation for subsequent fine-tuning stages that enhance its performance

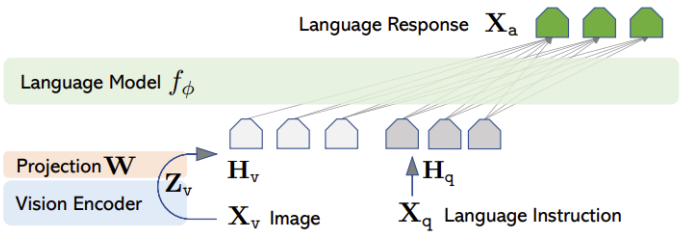


Figure 2.9: The LLaVA architecture.

on specific tasks LLaVA excels in solving tasks such as image captioning, where it provides detailed textual descriptions of images, and Visual Question Answering (VQA), where it answers contextually relevant questions about an image. Additionally, it is highly effective for instruction-following tasks, such as explaining visual memes or grounding textual references to specific entities in images using bounding boxes. Its success lies in its ability to combine the strengths of pre-trained vision and language models while ensuring they work harmoniously through alignment techniques like the Projector. By leveraging powerful pre-trained components, LLaVA achieves state-of-the-art performance with relatively low computational overhead, making it a versatile tool for applications in human-computer interaction, automated content generation, and multimodal research.

Figure 2.9 illustrates the architecture of LLaVA, specifically highlighting how visual and textual inputs are processed to generate a language response. The process begins with a vision encoder, which extracts image features $\mathbf{X}_v$ and encodes them into a latent representation $\mathbf{Z}_v$. These visual features are then projected into the embedding space of the language model via a trainable projection matrix $\mathbf{W}$, producing $\mathbf{H}_v$. This ensures compatibility between the visual and textual modalities. Simultaneously, the language model receives textual instructions $\mathbf{X}_q$, such as prompts or queries, which are encoded into the textual embedding $\mathbf{H}_q$. Both $\mathbf{H}_v$ and $\mathbf{H}_q$ are fed into the language model f_ϕ, which integrates the visual and textual embeddings to generate the final language response $\mathbf{X}_a$. This response reflects the understanding of the image and the provided instruction, enabling tasks like image captioning, visual question answering, or multi-modal reasoning.

On the other hand, CogVLM [274] takes a different approach. It bridges the gap between a frozen pre-trained language model and a frozen image encoder by inserting a trainable visual expert module into the attention and feed-forward network (FFN) layers. This modification allows the model to more deeply understand both the text and image inputs. Its architecture allows it to process images, supporting resolutions up to 490×490 pixels, facilitating detailed visual understanding. The model has demonstrated state-of-the-art performance across multiple cross-modal benchmarks. Pre-training in CogVLM involves a

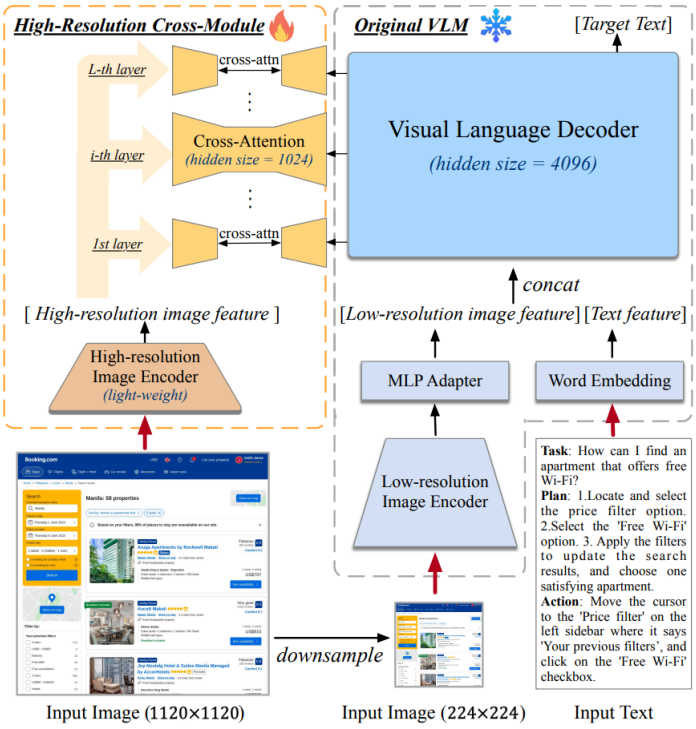


Figure 2.10: The CogAgent architecture, which extends the CogVLM architecture (right) with the cross-attention module (left).

two-stage process designed to integrate visual and textual information effectively. In the first stage, the model is trained on an image captioning task, predicting the next token in the text sequence based on the visual input. This stage utilizes a large dataset of 1.5 billion image-text pairs and runs for $120,000$ iterations with a batch size of $8,192$. The second stage combines image captioning with referring expression comprehension, enabling the model to understand and generate textual descriptions that refer to specific regions within an image. However, a limitation of CogVLM is that it processes images at a lower resolution, which makes it less capable of capturing smaller details in the image. As a follow-up to CogVLM, CogAgent [106] is built on the same architecture but introduces a visual cross-attention module. This module operates between the image-text input pair and a high-resolution version of the input image. By incorporating this high-definition image, the model can boost performance, enabling it to capture finer details and provide more accurate visual understanding. Both CogVLM and CogAgent excel in understanding and describing images, following instructions, and solving image-dependent tasks such as Visual Question-Answering (VQA). They also have the ability to reference specific entities within an image using bounding boxes, a crucial feature that helps these models focus on particular objects and allows intelligent agents to better understand their surroundings.

Figure 2.10 presents the architecture of CogAgent, which builds upon the CogVLM framework while incorporating a high-resolution cross-attention module for enhanced image understanding. On the right, the architecture follows the original CogVLM design. Input images are first processed by a low-resolution image encoder, which downscales them to 224×224 dimensions, extracting low-resolution image features. Simultaneously, input text is embedded through a word embedding layer. Both the text features and low-resolution image features are concatenated and passed through an MLP adapter to align them before being input to the visual language decoder, which has a hidden size of 4096. This decoder integrates the multimodal features and generates the target text response. On the left, the new high-resolution cross-attention module extends the architecture to improve visual grounding and reasoning. High-resolution images, preserved at 1120×1120, are processed by a lightweight high-resolution image encoder to extract detailed image features. These features are passed through multiple layers of cross-attention modules, where they interact with the intermediate representations of the visual language decoder. This cross-attention mechanism, with a hidden size of 1024, allows the decoder to leverage high-resolution details, capturing finer visual information.

Finally, a smaller multimodal model called MiniCPM-V 2.6 [284] has been released, which aims to reduce the number of parameters while maintaining strong OCR and multimodal capabilities. Unlike larger models such as CogVLM and LLaVA, which require considerable computational resources, MiniCPM-V 2.6 is tailored for scenarios where computational efficiency and speed are crucial, such as mobile phones, thanks to its carefully designed training methodology that enables scaling of both model size and data horizons. The architecture of MiniCPM-V 2.6 consists of three main components. The visual encoder, based on the SigLIP model, processes high-resolution images to extract visual tokens. These tokens are then passed through a compression layer, which includes a perceiver resampler (as in Flamingo [4]) that uses cross-attention to reduce the dimensionality of the data while retaining key features. Finally, the compressed visual tokens and text inputs are fed into the LLM, based on the recent Qwen2-7B [283] architecture. The pre-training of MiniCPM-V involves a multi-phase approach. Initially, the model undergoes pre-training on large-scale datasets to learn general language and vision-language representations. This is followed by supervised fine-tuning (SFT) on specific tasks to enhance its performance in targeted applications. Remarkably, it shows competitive performance even when compared to much larger models. This makes it an appealing choice for real-time or resource-constrained environments, as it can process complex multimodal inputs without sacrificing the depth of its understanding or generation quality.

Figure 2.11 illustrates the architecture of MiniCPM, which integrates adaptive visual

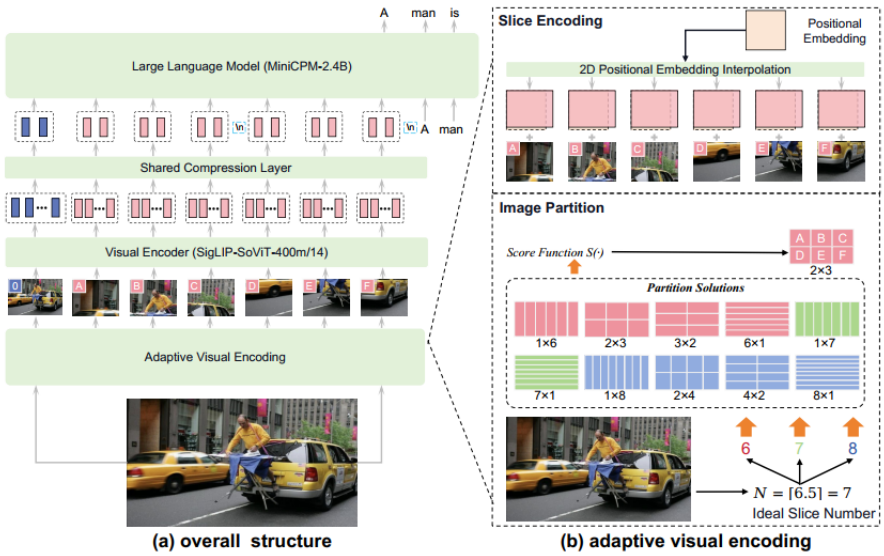


Figure 2.11: The MiniCPM architecture.

encoding with a large language model (LLM) for multimodal processing. In the overall structure *(a)*, the input image is processed through an adaptive visual encoding layer, which partitions the image into slices to extract visual features. These features are passed to a visual encoder to generate compact visual tokens. A shared compression layer then reduces the dimensionality of these tokens to align them with the language embedding space of the MiniCPM model. The LLM receives both the visual tokens and text input, integrating them to produce a multimodal language response. The adaptive visual encoding module, detailed in *(b)*, partitions the input image into slices based on a scoring function $S(f)$, which determines the ideal slice number N for optimal feature extraction. Various partition solutions are considered, such as 1×6, 2×3, 4×2, etc., and the scoring function selects the configuration that best represents the image. Once partitioned, 2D positional embedding interpolation is applied to the slices to preserve spatial relationships, generating slice encodings. These encodings are then passed through the rest of the pipeline, allowing the model to handle high-resolution images efficiently while maintaining strong multimodal understanding capabilities.

The development of recent vision-language models like LLaVA, CogVLM and MiniCPM has showcased the power of integrating visual and textual modalities to solve complex multimodal tasks. However, these models require huge amounts of data, particularly images and text, which is challenging to collect. This issue will be further explored in Chapter 4, in particular in Section 4.3.1. Moreover, as these models grow in scale and sophistication, their training and adaptation demand increasingly significant computational and data resources. This challenge underscores the need for sustainable training methodologies that

balance performance and efficiency. The next focus is on techniques like fine-tuning, LoRA, and quantization, which aim to optimise training pipelines for resource constraints while maintaining state-of-the-art capabilities.

2.6 Sustainable Training Methods

The development of large language models (LLMs) has revolutionised natural language understanding (NLU), transitioning from task-specific architectures to expansive, multi-task, and multi-modal frameworks capable of addressing a wide array of linguistic and visual challenges. Traditional training methods for such models required task-specific annotated datasets and significant computational resources. These approaches often relied on fine-tuning pre-trained models for specific objectives, which, while effective, became increasingly infeasible as the scale of LLMs grew. In contrast, zero-shot, few-shot, and instruction-based learning approaches leverage the inherent generalisation capabilities of pre-trained models, significantly reducing the need for task-specific data and enabling the application of LLMs across diverse domains with minimal additional training. Despite these advancements, the ever-growing size of LLMs and multimodal large language models (MLLMs) introduces severe computational and memory constraints, necessitating novel strategies for sustainable training.

Modern LLMs frequently surpass 100 billion parameters, and while smaller models, such as those with one billion parameters, can be fine-tuned within the memory limits of a single GPU, larger models require considerable resources. Training these models efficiently involves overcoming the limitations of memory bandwidth, computational power, and data parallelism. To address these challenges, adaptation methods such as Low-Rank Adaptation (LoRA) [119] and Quantized LoRA (Q-LoRA) [69] have emerged as key techniques. LoRA introduces trainable low-rank matrices (W_{down} and W_{up}) into specific layers of the model, freezing the majority of the pre-trained weights (W_q, W_k, W_v, and W_o). Here, W_{down} is a projection matrix that reduces the dimensionality of the input by mapping it to a lower-dimensional space. This reduction captures the most salient features necessary for the specific task, minimizing computational complexity. W_{up} is a projection matrix that maps the lower-dimensional representation back to the original higher-dimensional space. This expansion integrates the task-specific adaptations into the existing feature space of the model. This projection and expansion effectively works as an autoencoder. By only updating the low-rank matrices, LoRA significantly reduces the number of trainable parameters and approximates the necessary weight updates for fine-tuning, enabling task-specific fine-tuning even on limited hardware. For instance, LoRA facilitates the fine-tuning of 7-billion-parameter models

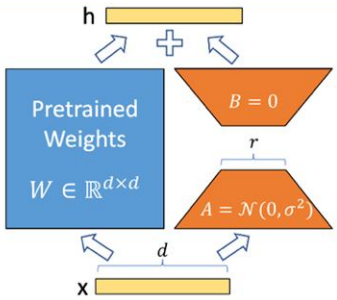


Figure 2.12: The LoRA flow. On the left a pre-trained model, that is kept frozen. On the right, the two matrices (A and B) which function as an approximation of the training.

like LLaMA on a single 16GB GPU, making it feasible to explore sophisticated linguistic objectives without excessive infrastructure demands [255, 230, 109].

In Figure 2.12 the LoRA mechanism, used to fine-tune pre-trained models efficiently is represented. The original weight matrix $W \in \mathbb{R}^{d \times d}$, shown in blue, represents the pre-trained weights of a model. These weights remain *frozen* during the fine-tuning process, preserving the knowledge already learned during pre-training. The input vector x of dimension d is processed through the pre-trained weights W to generate the primary output. To adapt the model for specific tasks, LoRA introduces two additional *trainable matrices*, A and B. The matrix $A \in \mathbb{R}^{d \times r}$ projects x into a lower-dimensional space of size r (where $r \ll d$), and $B \in \mathbb{R}^{r \times d}$ maps the resulting representation back to the original dimension. Importantly, B is often initialized to zero, ensuring that the outputs of the pre-trained model are not perturbed at the start of fine-tuning. The final output h is computed as the sum of the output from the frozen pre-trained weights ($W \cdot x$) and the task-specific adaptation introduced by the low-rank decomposition ($B \cdot A \cdot x$).

Q-LoRA [69] extends this efficiency by integrating quantization into the fine-tuning process. Quantization reduces the precision of model parameters, such as converting 16-bit floating-point values to 4-bit integers, thereby drastically reducing memory usage while preserving downstream task performance. Moreover, Q-LoRA permits the introduction of LoRA matrices into all the modules of the Transformer:

- W_q (Query weight matrix): Transforms the input embeddings into query vectors, which are used to assess the relevance of other tokens in the sequence.
- W_k (Key weight matrix): Transforms input embeddings into key vectors, facilitating the computation of attention scores that determine the influence of each token on others.
- W_v (Value weight matrix): Transforms input embeddings into value vectors, which are

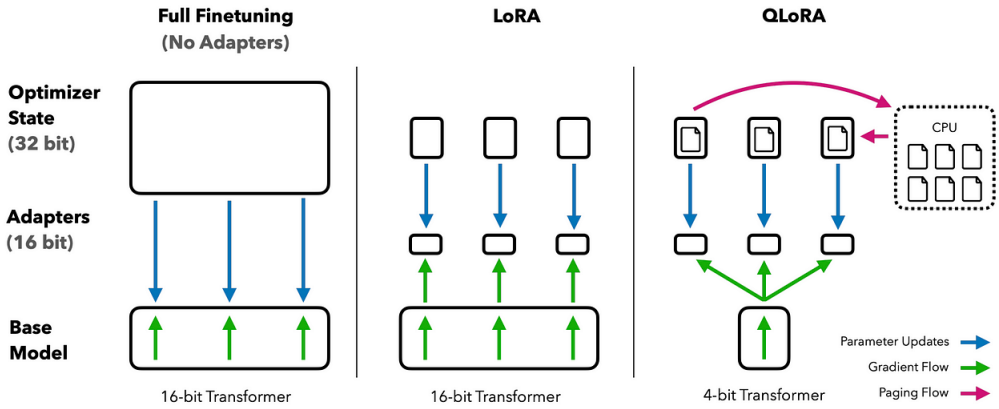


Figure 2.13: The QLoRA flow in comparison with the traditional full fine-tuning and the LoRA adaptation.

weighted and aggregated based on attention scores to produce the attention output.

- W_o (Output weight matrix): Applies a linear transformation to the concatenated outputs from multiple attention heads, integrating the attention information back into the representation space of the model.

This technique not only compresses the model to fit within available hardware but also ensures that fine-tuning maintains high precision by dynamically dequantizing parameters during computations, balancing efficiency with accuracy.

Figure 2.13 illustrates the differences between three approaches for fine-tuning transformer models: Full Fine-Tuning, LoRA, and Q-LoRA, focusing on their resource usage and efficiency. In Full Fine-Tuning, all parameters of the transformer model are updated during training. This requires a 16-bit transformer, a large optimizer state (32-bit), and significant memory resources since no parameter-efficient techniques are employed. Consequently, this approach is computationally expensive and often impractical for large-scale models on limited hardware. LoRA addresses these limitations by freezing the pre-trained model weights and introducing lightweight adapters into specific layers of the transformer. The model itself remains a 16-bit transformer, but only the adapters are updated during training, drastically reducing the optimizer state. Q-LoRA builds on LoRA by incorporating quantization, further improving resource efficiency. It reduces the precision of the weights of the base model to 4 bits, which significantly decreases memory requirements. Additionally, Q-LoRA offloads some operations, such as paging, to the CPU.

Quantization and low-bit precision methods are increasingly recognised as fundamental for sustainable training of LLMs and MLLMs. These approaches optimise resource usage, enabling efficient training and fine-tuning of large models in low-resource environ-

ments. In recent months, such methods have been successfully applied to Visual Language Models (VLMs), demonstrating their adaptability to multimodal tasks. For instance, by integrating LoRA and Q-LoRA with vision encoders, researchers have extended the benefits of parameter-efficient adaptation to models like LLaVA and MiniCPM, enabling effective visual-textual reasoning and multi-modal understanding with reduced computational requirements. This flexibility positions these methods as critical tools for advancing LLM and VLM research while addressing the pressing challenges of resource constraints in diverse domains. These parameter-efficient techniques have proven effective across a wide range of applications, including text classification, generation, visual question answering, and dependency parsing. Their ability to adapt large-scale models to specific tasks without incurring prohibitive computational costs ensures their utility in both high-resource and low-resource settings. This versatility not only optimises the training pipeline but also makes LLMs and MLLMs accessible to broader domains and research communities, paving the way for sustainable advancements in NLU and multimodal learning.

Model	**Traditional Fine-Tuning**	**LoRA Additional**	**Q-LoRA Additional**
LLaMA-7B	7B	0.42B	0.21B
LLaMA-13B	13B	0.78B	0.39B
LLaMA-65B	65B	3.90B	1.95B

Table 2.1: Comparison of model parameters during inference and training for different tuning methods, all numbers are to be intended as billions.

The tables 2.1 and 2.2 provide a detailed comparison of model parameters and memory usage during training for different tuning methods. In Table 2.1, we observe that while traditional fine-tuning requires updating all parameters of the model, LoRA and Q-LoRA introduce only a small fraction of additional parameters, approximately 6% and 3%[3] of the total, respectively. This results in a minimal increase in the overall size of the models, making these methods highly efficient in terms of parameter overhead.

Model	**Traditional Fine-Tuning**	**LoRA Tuning**	**Q-LoRA Tuning**
LLaMA-7B	28GB	7+4GB	7+2GB
LLaMA-13B	52GB	13+8GB	13+4GB
LLaMA-65B	260GB	65+40GB	65+20GB

Table 2.2: Memory requirements during training for different tuning methods, all numbers are to be intended as GBs.

[3]This percentage is computed assuming an r hyper-parameter equal to 8

Table 2.2 highlights the significant reduction in memory requirements when using LoRA and Q-LoRA compared to traditional fine-tuning. For instance, training LLaMA-7B with traditional fine-tuning requires approximately 28 GB of memory, whereas LoRA reduces this to 4 GB and Q-LoRA further reduces it to just 2 GB, summing of course the memory requirements for the forwards pass of the original model, i.e. 7 GB. This reduction becomes even more pronounced for larger models such as LLaMA-65B, where Q-LoRA requires only 20 GB of memory compared to the 260 GB needed for traditional fine-tuning. These results demonstrate that LoRA and Q-LoRA are not only parameter-efficient but also memory-efficient, making them practical choices for large-scale model tuning in resource-constrained environments.

This chapter has provided an extensive survey of modern approaches to Situated Natural Language Understanding (Situated NLU), laying the groundwork for the contributions presented in the remainder of this thesis. The surveyed topics are crucial for enabling intelligent systems to interpret language in real-world environments, a core challenge of this work. Starting from foundational aspects of Human-Robot Interaction and Natural Language Processing and Interpretation, the importance of grounding language in physical contexts has been explored for effective human-robot communication. Next, Semantic Role Labeling was presented as a structured method for semantic interpretation, essential for task-oriented systems operating in dynamic settings. The review continued with state-of-the-art transformer-based architectures and Large Language Models, focusing on instruction-tuned models that improve task generalization. Since this thesis aims to extend such models to multimodal and dialogic scenarios, the chapter also examined multimodal architectures that integrate language and vision, laying the foundation for grounded interaction. Lastly, sustainable training techniques like LoRA and Q-LoRA were discussed, providing methods for efficient fine-tuning under resource constraints. These insights collectively set the stage for the novel methodologies and contributions presented in the following chapters.

Chapter 3

Improving the Training in LLMs for Complex Linguistic Tasks

In the previous chapter, we analysed the Transformer architecture and the foundational principles of Large Language Models (LLMs), examining their core components, capabilities, and practical applications. Among these, we emphasised the significance of in-context learning, which enables models to execute tasks without further parameter updates, and fine-tuning, the traditional method for optimising model performance on specific datasets or objectives. Additionally, we highlighted the growing importance of LLMs tailored for the Italian language, given the scarcity of resources for non-English languages, and explored multimodal models that integrate diverse data modalities to enhance their utility.

As LLMs grow increasingly complex, their size and computational demands pose significant challenges. Training and deploying these models require substantial memory and processing power, often limiting accessibility to well-resourced institutions. To address these issues, researchers have developed parameter-efficient fine-tuning techniques, such as LoRA [119, 69]. By isolating task-specific updates and incorporating them into pre-trained models, LoRA significantly reduces computational costs, enabling scalable fine-tuning without compromising model performance.

This chapter explores three distinct methodologies for adapting LLMs to specific applications, each addressing unique challenges and demonstrating the versatility of such models across linguistic and multimodal contexts. These approaches not only showcase the flexibility of LLM architectures but also illustrate how targeted adaptations can unlock their potential across a wide array of tasks. Organised into three methodological frameworks, this chapter introduces an incremental complexity for fine-tuning LLM-based architectures, ranging from simple rewriting tasks to multitask learning and multimodal integration:

- **Training a Large Language Model for Rewriting Natural Language into Semi-Structured Syntactic Representations**: A foundational challenge in natural language processing is the conversion of free-form text into structured representations suitable for downstream tasks. Models like `DepPLLaMA` [112] and `U-DepPLLaMA` [114] are examples of this capability by fine-tuning LLMs for syntactic parsing. These models represent a critical step in transitioning from natural language to semi-structured, syntax-based text. They generate dependency parsing trees (structured, interpretable representations of linguistic syntax) enabling applications in computational linguistics and multilingual analysis. Importantly, `DepPLLaMA` and `U-DepPLLaMA` achieve state-of-the-art results without requiring specialised architectures, underscoring the adaptability of Transformer-based models. This section demonstrates how to employ the LoRA approach to fine-tune large models (7B or 13B parameters) on a downstream task.

- **Multitask Scaling to the Extreme**: Extending the functionality of LLMs to handle a variety of tasks is essential for broad applicability. `ExtremITA` [109] highlights the potential of multi-instruction tuning, where tasks are framed as natural language prompts, allowing a single model to process diverse objectives within a unified framework. This multitask paradigm streamlines the integration of numerous tasks into a single monolithic system, reducing the need for bespoke models while maintaining high performance. `ExtremITA` demonstrates how carefully curated instruction tuning enhances the generalisability and efficiency of LLMs. This capability is a cornerstone of what we expect from intelligent robots: the ability to solve multiple, diverse problems while maintaining high performance across all tasks, as humans do.

- **Incorporating Multimodal Evidence into LLMs**: Expanding the scope of LLMs to integrate multiple data modalities, such as text and images, opens new frontiers in AI research. Multi-Modal Visual Question Answering in Italian (`MM-VQA-it`) [233] represents a significant advancement in this domain by fine-tuning models to process and reason across visual and textual inputs. By addressing tasks that require synthesising multimodal information, `MM-VQA-it` introduces a straightforward methodology to enhance the capability of LLMs to mirror human-like reasoning, such as answering questions about images. This methodological approach differs radically from the others, as it involves fine-tuning all components of the model, i.e., no LoRA is employed and all model parameters are trained.

The three methodologies explored in this chapter directly address core challenges in extending Large Language Models for Situated Human-Robot Interaction, a central theme of this thesis. Specifically, each method contributes to the overarching goal of creating a robust,

adaptable multimodal system capable of interpreting language and context in real-world environments. Fine-tuning models for syntactic parsing supports structured understanding of commands, crucial for grounding language in physical tasks. Multitask scaling equips models with the flexibility to handle a diverse range of tasks, a fundamental requirement for intelligent agents operating in complex settings. Lastly, integrating multimodal evidence lays the foundation for perception-driven interaction, enabling models to bridge textual and visual information. These methodologies collectively demonstrate how targeted adaptations of LLMs can overcome key limitations in HRI and provide a stepping stone towards building a comprehensive multimodal dialogue system.

Moreover, this chapter provides a comprehensive overview of these training methodologies, illustrating how they enhance the adaptability, efficiency, and scope of LLMs. The foundational methods discussed here have been instrumental in training and evaluating other models and systems presented throughout this thesis. The ultimate goal is to explore various training modalities that, when combined, could one day enable the creation of a truly intelligent robot.

3.1 Training a Large Language Model for Rewriting Natural Language into Semi-Structured Syntactic Representations

Training large language models (LLMs) under constrained computational resources presents a significant challenge, especially as the demand for more sophisticated natural language understanding (NLU) capabilities grows. This section demonstrates the application of Q-LoRA (Quantised Low-Rank Adaptation) [69] as an efficient method for fine-tuning a large LLM on a single 16GB GPU, using the task of dependency parsing as a testbed for the approach. Dependency parsing serves as an interesting test case for evaluating LLM fine-tuning due to its relevance in two critical aspects of situated NLU.

Firstly, formal syntactic representations, such as parse trees, are valuable as knowledge representation formalisms for integrating linguistic input into dynamic, context-aware systems, particularly in situated environments like Human-Robot Interaction (HRI) [128, 170]. Parse trees provide structured insights into relationships between words, enabling systems to capture how words interconnect within sentence structures. Secondly, dependency parsing exemplifies complex linguistic inference, a core requirement in HRI and similar domains where syntactic understanding contributes to tasks such as instruction execution and object grounding [32]. By leveraging dependency parsing as a mere task, this section illustrates

how Q-LoRA facilitates efficient training of LLMs on computationally intensive tasks under constrained resources.

Dependency parsing, a critical component of NLU, involves analysing the grammatical structure of sentences to delineate relationships between words [140]. Its primary goal is to establish dependencies among words, enabling systems to capture how words interconnect and rely on one another within sentence structures [258]. This process serves as the backbone for a myriad of downstream applications, including machine translation [172, 72], question answering [31, 96], and information retrieval [182, 127], as it allows systems to derive meaning and context from unstructured text [67, 219]. Moreover, high-performance neural parsing systems support the study of emergent grammatical competence, as demonstrated through probing techniques [59]. These could be used to test the internal representation and understanding of grammatical rules of the language in a model, as it is thought to be fundamental for comprehending and executing commands in robotic platforms. Despite its centrality to language understanding, developing dependency parsers that are both robust and efficient remains a complex challenge [158]. This complexity is further magnified in multilingual contexts and low-resource languages, where the scarcity of annotated data hinders progress.

The integration of LLMs into dependency parsing workflows addresses key limitations of traditional approaches. Conventional parsers often depend on manually engineered features or bespoke architectures tailored to specific tasks or languages, resulting in limited scalability and flexibility. In contrast, LLMs provide a unified, data-driven framework that can adapt to diverse linguistic phenomena with minimal task-specific adjustments. This versatility enhances their applicability across various languages, particularly in multilingual and low-resource scenarios where task-specific solutions are often impractical. A prominent parsing technique is the shift-reduce method [196, 48]. This parser processes sentences from left to right while maintaining a buffer for unprocessed words. Machine learning-based approaches, such as biaffine neural networks [76], have also proven effective in capturing complex word dependencies. An alternative approach is UDPipe [247], which focuses on parsing within the Universal Dependency Framework [68]. UDPipe handles dependency parsing alongside tasks such as tokenization, morphological analysis, part-of-speech tagging, and lemmatization across multiple languages without relying on external data. It employs a Bi-LSTM architecture fed with end-to-end, character-level, pre-trained, and contextualised embeddings. Trained on a vast multilingual dataset, UDPipe captures cross-lingual relationships effectively. The system was later extended as UDPipe+ [248], integrating multilingual BERT [70] into its token representations.

Despite their state-of-the-art results, these methods are tailored for structured prediction

problems and rely on task-specific techniques, specialised neural architectures, and complex optimisation strategies. The emergence of sequence-to-sequence (seq2seq) modelling for dependency parsing [157] introduced a simpler, end-to-end paradigm. This approach allows models to predict the relative position of word heads directly within a sentence. It also incorporates a beam search decoder with tree constraints and sub-root decomposition to improve accuracy. Further advancements include [137], where a multi-task, multilingual BERT [70] pre-trained on 104 languages demonstrated the ability to predict dependency parsing trees, lemmas, part-of-speech tags, and more within a unified framework.

The goal of this section is to evaluate the ability of LLMs to perform end-to-end dependency parsing without requiring task-specific architectural designs. We present a straightforward methodology to fine-tune LLaMA-based models for this downstream task. Specifically, we focus on generating tree-shaped representations that transform sentence syntax into bracketed structures. These bracketed structures, directly derived from sentence dependency graphs, represent the syntactic structure of a sentence in tree format [64]. The final result is a system that processes raw sentences as input and outputs these bracketed forms. Once the bracketed representation is obtained, constructing the corresponding dependency graph becomes straightforward. The Dependency Parsing task serves as an example of how LLMs can be effectively and easily fine-tuned for syntactic downstream tasks, even in limited-resource scenarios.

3.1.1 Large Language Models for Dependency Parsing

This section presents two dependency parsing models developed using LLMs: DepPLLaMA (Dependency Parsing LLaMA) [112] and its evolved version, U-DepPLLaMA (Universal Dependency Parsing LLaMA) [114]. Both models are LLaMA-based architectures fine-tuned for dependency parsing but the former is trained and optimized for the Italian language, while the latter is extended to support a broader spectrum of languages, in a multi-lingual scenario. Modelling tasks using LLMs poses a unique challenge due to their inherent nature of taking sequences as input and generating sequences as output. To illustrate this, let's consider the following sentence:

This section presents two dependency parsing models developed using LLMs: DepPLLaMA (Dependency Parsing LLaMA) [112] and its evolved version, U-DepPLLaMA (Universal Dependency Parsing LLaMA) [114]. Both models are based on LLaMA architectures fine-tuned for dependency parsing; however, the former is specifically trained and optimised for the Italian language, while the latter extends its support to a broader spectrum of languages in a multilingual scenario. Modelling dependency parsing tasks using LLMs intro-

duces unique challenges due to their inherent sequence-to-sequence nature, where inputs and outputs are treated as linear sequences. To illustrate this concept, consider the following sentence:

"*He was most widely recognized for some of his books.*" (3.1)

The Dependency Graph of this sentence is depicted in Figure 3.1, where nodes represent words and arcs signify syntactic relationships, each labelled with a specific dependency type. A notable node in such graphs is the `ROOT`, which typically corresponds to the main verb. The same syntactic information can be represented as a Dependency Tree, shown in Figure 3.2. In this tree, the main verb acts as the actual tree `ROOT` and non-terminal nodes represent dependency labels, while terminal nodes correspond to the actual words of the sentence. For instance: the `NSUBJ` arc identifies *He* as the subject of the verb; the `OBL` arc highlights the nominal phrase *some*; the verb *recognized* is modified by the adverb *widely*, which itself is further modified by *most* through the `ADVMOD` relation. Both representations, the Dependency Graph and the Dependency Tree, convey the same syntactic information. The transition between these representations preserves all structural details.

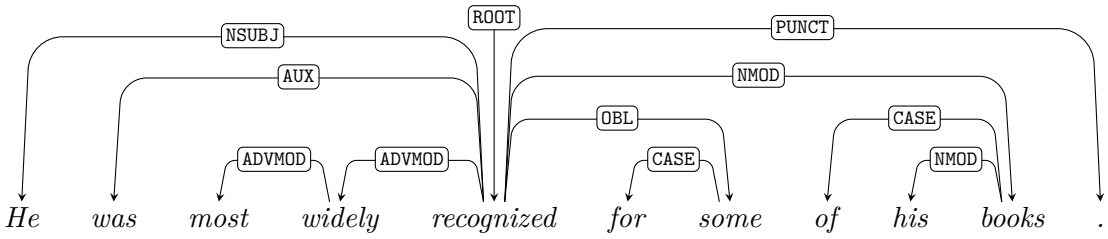


Figure 3.1: Example of a dependency graph associated with the sentence (3.1).

Finally, the plain parenthetic representation of the dependency tree shown in Figure 3.2 corresponds to the following form:

$$\begin{aligned}&[\texttt{ROOT}[\texttt{NSUBJ}[\textit{He}]][\texttt{AUX}[\textit{was}]][\texttt{ADVMOD}[\texttt{ADVMOD}[\textit{most}]][\textit{widely}]][\textit{recognized}]\\&[\texttt{OBL}[\texttt{CASE}[\textit{for}]][\textit{some}][\texttt{NMOD}[\texttt{CASE}[\textit{of}]][\texttt{NMOD}[\textit{his}]][\textit{book}]]][\texttt{PUNCT}[.]]]\end{aligned} \tag{3.2}$$

The idea is thus straightforward: to employ an LLM to transform the example sentence (3.1) into its parenthetical format (3.2) using solely raw text, without relying on intermediate information such as morphological data, part-of-speech tags, or lemmatization. In particular, this sequence-to-sequence model is obtained by fine-tuning the LLaMA2 $7B$[1] and $13B$[2]

[1] https://huggingface.co/meta-llama/Llama-2-7b-hf
[2] https://huggingface.co/meta-llama/Llama-2-13b-hf

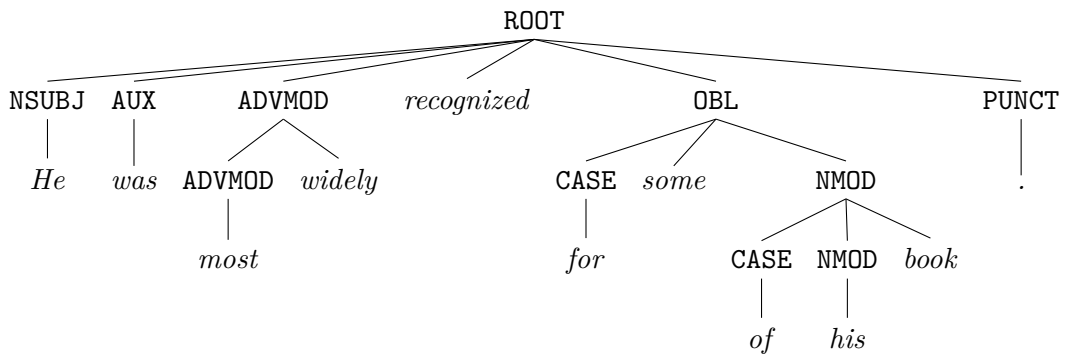


Figure 3.2: The syntactic parse tree associated with the dependency graph from Figure (3.1).

models [262]. As of this writing, these models strike an optimal balance between accuracy and size across a broad range of tasks, as detailed in [262].

LLaMA2 operates as an auto-regressive sequence-to-sequence model, processing input text and generating output text. The model begins by analysing the input sentence to identify and generate the `ROOT` node, which typically corresponds to the primary verb or central idea of the sentence. Once the root is established, the attention mechanism delineates the primary nodes directly connected to the root, capturing the main syntactic constituents or "branches" sprouting from it. At this point, the model employs a recursive strategy: for each primary node, it drills down to identify and generate the corresponding subtrees. As it traverses deeper into the syntactic branches of the sentence, the phenomena encountered become more localized and, thus, more straightforward. The model processes each segment sequentially: first "*He*", then "*was*", followed by "*most widely*", and so forth. This step-by-step approach ensures that each subtree (i.e., syntactic chunk) is unpacked systematically, while the attention mechanism dynamically handles both short-range and long-range dependencies. The recursive nature of this method ensures a thorough and hierarchical examination of each segment, from broad constituents to specific syntactic details. This first idea was explored in `DepPLLaMA` [112], specifically tailored for Italian. Given the inherent multilingualism of syntactic structures, as formalised in the Universal Dependency Annotation scheme [174], and the capability of LLMs to process multiple languages, we extend this framework to a multilingual model. By aggregating training data from as many languages as possible, we aim to achieve a singular, language-independent architecture.

It is important to note that LLaMA2 was not pre-trained on data from every existing language. Its pre-training corpus comprises texts from 27 languages, with a substantial bias towards English, which accounts for approximately 90% of the sentences. The remaining languages each contribute less than 1% of the training data (the full distribution is detailed in Table A.1 in the Appendix). To evaluate the efficacy of this model fairly, experiments

are restricted to the languages present in the pre-training data. While this approach does not address challenges associated with low-resource languages, which were not considered during LLaMA2 pre-training, this work aims to avoid attributing potential shortcomings of the initial experiments to a lack of pre-training data. Nonetheless, there are no intrinsic constraints on further fine-tuning LLaMA2 with data from additional languages or applying it to cross-lingual parsing scenarios, as demonstrated by [239]. To implement this methodology, we utilise training data from the Universal Dependency Treebank[3] and refer to this enhanced model as `U-DepPLLaMA`: Universal Dependency Parsing via Auto-regressive LLMs based on LLaMA [114].

The only potential complexity arises from the non-projectivity of the dependency graph, which may alter the word order within the tree. A dependency graph is said to be non-projective when arcs cross over one another, meaning the word order in the sentence does not align neatly with a tree structure [141]. Non-projectivity often arises in languages with flexible word order or constructions that involve long-range dependencies. While this phenomenon adds a layer of complexity to parsing, it remains manageable within the framework. However, this does not hinder syntactic analysis, as realigning the words to match the original sentence is straightforward. The recursive pseudo-code used to derive the parenthetical form from a dependency graph is provided in Table B.1 in the Appendix. This algorithm assumes that dependency graphs are loaded using the conllu[4] Python library. While Algorithm B.1 handles projective graphs effectively, it may encounter challenges when dealing with non-projective graphs. An example of such a case is shown in Figure 3.3, which depicts the non-projective dependency graph for the sentence:

"*I guess you get what you pay for.*"

In this example, the phrase "*what you pay for*" exhibits non-projectivity. However, as shown by the corresponding tree in Figure 3.4, it can still be generated straightforwardly. The phrase was reordered to "*what for you pay*". During fine-tuning, such derived non-projective trees are directly provided to the model. It is important to note that non-projective cases constitute a minor fraction of the overall Universal Dependency dataset. During tagging, realigning the sentence to its original word order while retaining the dependency relations is sufficient to ensure accurate analysis.

[3] https://universaldependencies.org/
[4] https://pypi.org/project/conllu/

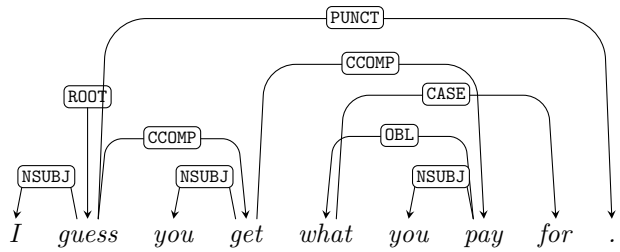


Figure 3.3: Example of a non-projective dependency graph associated with the sentence "*I guess you get what you pay for.*".

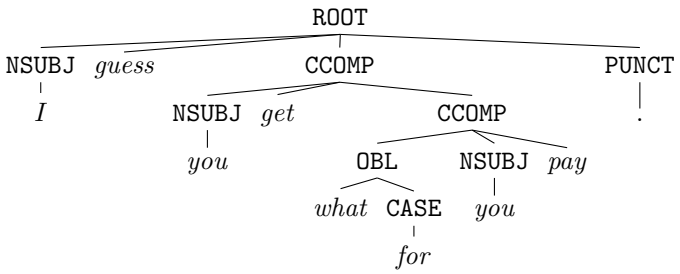


Figure 3.4: The syntactic parse tree associated with the dependency graph from Figure (3.3).

3.1.2 Experimental Setup and Evaluation

Training and fine-tuning large-scale models typically require vast computational resources, often necessitating multiple GPUs. As discussed in the previous chapter, an efficient solution is LoRA [119]. The Quantized-LoRA (Q-LoRA) technique, detailed in [69], further improves efficiency by propagating gradients through a 4-bit quantized frozen pre-trained model within the LoRA structure. This enables fine-tuning of a 7-billion-parameter model (7B) on a conventional 16GB GPU while preserving full 16-bit task performance. For the experiments, we adopted the Q-LoRA method to produce adapters[5] specialised for the dependency parsing task. The focus lies on assessing the potential of the proposed model for multilingual dependency parsing. Although this model is not inherently task-specific, its uniqueness stems from the distinct input and output representations. Key questions include its robustness across languages, particularly those it was minimally exposed to during pre-training (i.e., all except English), and the impact of sentence complexity on parsing quality. We also examine the benefits and limitations of employing architectures with 7 billion and 13 billion parameters. Lastly, we evaluate the end-to-end capability of the model, wherein it autonomously derives tokenization from raw sentences and generates the corresponding

[5]The model, along with all necessary software for training and utilizing U-DepPLLaMA, has been publicly released on a dedicated GitHub page at https://github.com/crux82/u-deppllama, under the LLaMA https://ai.meta.com/llama/license/ and UD https://github.com/UniversalDependencies/LICENSE licenses.

dependency graph.

In these experiments, we focused on the Universal Dependency Parsing dataset v2.3[6]. This approach aligns with the state-of-the-art architecture proposed in [248]. From this dataset, we concentrated on the subset of 27 languages supported by LLaMA2, as indicated in their report [262]. This subset spans languages from English and Finnish to Korean (listed in Table 3.1). However, texts written in Vietnamese were excluded due to encoding issues in the LLaMA2 tokenizer. The final dataset thus comprises examples written in 26 languages. For high-resource languages like Czech, Norwegian, and Russian, which contain over 30,000 examples, the training datasets were capped at 30,000 sentences per language to maintain balance across all languages. After this reduction, the aggregated dataset includes 392,088 training examples, 59,084 development sentences, and 62,069 testing examples. To ensure consistency with the evaluation frameworks presented in [247, 248], two pivotal metrics were employed: Unlabeled Attachment Score (UAS), which evaluates the precision of the dependency tree structure by verifying the correct generation of head and dependency arcs; Labeled Attachment Score (LAS), which offers a stricter evaluation by assessing the accuracy of the assigned dependency labels for each arc. The training process was implemented using PyTorch and the HuggingFace library, combined with the Peft package for Q-LoRA. The LLaMA2 models were subjected to 3 training epochs, utilizing a learning rate of $3 \cdot 10^{-4}$ and a batch size of 16. To optimize the performance of the model, a linear scheduler with warmup was employed, incorporating a warmup ratio of 0.1. Throughout the training process, Q-LoRA with 4-bit precision was utilized to enhance the W_q, W_v, W_k, W_o, W_{gate}, W_{down}, W_{up} and W_{lm_head} modules of the transformer, as detailed in the previous Chapter 2.6. The LoRA matrices featured a matrix rank (R) of 8 and a parameter α of 16.

Training was conducted on a single NVIDIA H100 GPU with 80GB total memory. Using Nvidia MIG, only 20GB was utilised during both training and inference. Training required approximately 160 hours for the U-DepPLLaMA 7B model and 190 hours for the U-DepPLLaMA 13B model. This result demonstrates the feasibility of Q-LoRA for fine-tuning even on standard hardware with the two smallest LLaMA2 models (7B and 13B) without requiring extensive computational resources[7]. All hyperparameters were selected using a 10% subset of the training and development data. In the final generation process, no sampling method was applied, as the objective was to produce the best sequence that explains the given sentence, not to increase sequence variability. Consequently, a greedy decoding policy was adopted. Preliminary experiments using beam search revealed a modest

[6] http://hdl.handle.net/11234/1-2895

[7] Reducing the batch size to 4, these models can also be trained on an NVIDIA T4 GPU with 16GB memory.

0.2% increase in UAS and/or LAS, but at a significant computational cost: decoding time increased by nearly an order of magnitude. To prioritise efficiency, the greedy approach was preferred as we are interested in exploring the adaptability of an LLM to syntactic-based tasks, rather than optimising performance. The final goal is to show how such models can be fine-tuned with limited resources.

The experimental results focus on the comparison between `U-DepPLLaMA` and the state-of-the-art linguistic analysis system, UDPipe2.0 [248]. The primary distinction between the two lies in their architectures. UDPipe2.0 is a task-specific system, combining recurrent networks, attention-based biaffine networks, and diverse word embedding techniques. It also handles multiple tasks beyond parsing, including POS tagging and lemmatization, and has been trained on 89 datasets across 54 languages. In contrast, `U-DepPLLaMA` is based on LLaMA2 and does not rely on architectural modifications. Its uniqueness stems from the prompting method used for parsing. `U-DepPLLaMA` is trained on a narrower set of 50 datasets for LLaMA-supported languages. Although its architecture possesses significantly more parameters, it remains a task-agnostic model, leaving open possibilities for adaptation to other tasks. Further discussion on multi-task scenarios and how problems can mutually benefit each other is provided in the next section, ExtremITA [109].

Different output formats have been explored, as in [157, 136], where sentences like "*The dog runs*" are rewritten into pseudo-sentences such as "*`R1-det R1-nsubj ROOT`*". These approaches explicitly count the relative distance between head and modifier tokens to label dependency relations. However, LLMs are not inherently designed for numerical precision tasks [223, 123], which makes measuring token distances challenging. This limitation particularly affects decisions involving long-distance dependencies, such as correctly attaching a punctuation mark to a distant ROOT.

Table 3.1 presents *UAS* and *LAS* performance metrics for both UDPipe2.0 and its enhanced variant, UDPipe2.0++. The key improvement in UDPipe2.0++ stems from the integration of BERT and additional word/character embeddings for input representation, leading to a noticeable boost in performance across all treebanks. Both UDPipe models rely on gold-standard (GS) tokenization for input text. For consistency, the initial `U-DepPLLaMA` experiments also adopt this simplification.

The `U-DepPLLaMA` 7B and 13B architectures are evaluated using GS tokenization: the 7B version shows improvements for specific treebanks, such as Czech-CLTT, Dutch-LassySmall, German-GSD, and Italian-PoSTWITA, with on-par results for other Romance languages. However, it suffers significant performance drops for languages that are structurally different from English. The 13B version, with its larger parameter size, achieves better results across several treebanks, including Bulgarian, Croatian, Czech, French-GSD, Italian, Nor-

Language	UDPipe 2.0		UDPipe 2.0++		U-DepPLLaMA 7B GS Tokenization		U-DepPLLaMA 13B GS Tokenization		U-DepPLLaMA 7B End-to-End	
	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS	UAS	LAS
Bulgarian-BTB	93,38%	90,35%	95,34%	92,62%	96,21%	93,55%	**96,37%**	**93,77%**	*95,82%*	*93,27%*
Catalan-AnCora	93,22%	91,06%	94,49%	92,74%	94,28%	92,60%	**94,53%**	**92,86%**	*94,22%*	*92,47%*
Chinese-GSD	84,64%	80,50%	**90,13%**	**86,74%**	85,35%	81,09%	86,20%	82,21%	*83,40%*	*78,69%*
Croatian-SET	91,10%	86,78%	93,20%	89,35%	93,09%	88,47%	**93,71%**	**89,51%**	*93,07%*	*88,60%*
Czech-CAC	92,99%	90,71%	93,59%	91,50%	93,39%	91,19%	**94,48%**	**92,71%**	*93,50%*	*91,35%*
Czech-CLTT	86,90%	84,03%	89,59%	87,01%	93,14%	90,84%	91,82%	89,55%	***94,38%***	***92,28%***
Czech-FicTree	92,91%	89,75%	94,34%	91,87%	93,58%	90,94%	**95,30%**	**93,44%**	*93,74%*	*91,05%*
Czech-PDT	93,33%	91,31%	94,43%	92,56%	93,51%	91,44%	**94,85%**	**93,22%**	*93,45%*	*91,41%*
Danish-DDT	86,88%	84,31%	**89,32%**	**87,24%**	87,29%	84,63%	88,04%	85,55%	*87,77%*	*84,96%*
Dutch-Alpino	91,37%	88,38%	94,12%	91,78%	94,07%	91,78%	**94,32%**	**91,93%**	*92,94%*	*90,24%*
Dutch-LassySmall	90,20%	86,39%	93,07%	89,88%	**94,33%**	91,13%	94,19%	91,07%	*94,19%*	***91,29%***
English-EWT	89,63%	86,97%	**92,50%**	**90,40%**	92,19%	89,91%	92,32%	90,05%	*91,85%*	*89,44%*
English-GUM	87,27%	84,12%	**91,47%**	**88,80%**	90,84%	87,92%	90,42%	87,47%	*90,71%*	*87,88%*
English-LinES	84,15%	79,71%	87,28%	83,48%	**88,25%**	84,94%	88,23%	**84,97%**	*88,21%*	*84,71%*
English-ParTUT	90,29%	87,27%	**93,75%**	**91,12%**	93,05%	90,41%	92,96%	90,29%	*91,43%*	*88,85%*
Finnish-FTB	90,68%	87,89%	**91,68%**	**89,02%**	85,51%	80,60%	86,74%	81,95%	*91,31%*	*88,53%*
Finnish-TDT	89,88%	87,46%	**91,66%**	**89,49%**	85,96%	81,69%	86,90%	83,01%	*91,33%*	*88,74%*
French-GSD	90,65%	88,06%	92,55%	90,31%	93,51%	91,21%	**94,00%**	**91,90%**	*93,80%*	*91,51%*
French-ParTUT	92,17%	89,63%	**94,51%**	**92,47%**	92,03%	89,42%	91,36%	88,75%	*91,32%*	*88,34%*
French-Sequoia	92,37%	90,73%	**94,88%**	**93,81%**	92,02%	89,89%	91,92%	89,47%	*92,12%*	*89,54%*
French-Spoken	82,90%	77,53%	**86,27%**	**81,40%**	85,66%	80,24%	84,77%	78,92%	*85,30%*	*80,29%*
German-GSD	85,53%	81,07%	88,11%	84,06%	**88,34%**	**84,16%**	88,22%	84,14%	*87,82%*	*83,65%*
Hungarian-Szeged	84,04%	79,73%	**88,76%**	**85,12%**	85,42%	80,59%	86,87%	82,03%	*86,01%*	*81,24%*
Indonesian-GSD	85,31%	78,99%	86,47%	**80,40%**	86,43%	80,05%	86,09%	80,08%	***86,49%***	*80,32%*
Italian-ISDT	93,49%	91,54%	94,97%	93,38%	95,20%	93,61%	**95,65%**	**94,09%**	*95,02%*	*93,39%*
Italian-ParTUT	92,64%	90,47%	95,36%	93,38%	**96,65%**	**94,59%**	96,62%	94,42%	*96,20%*	*93,92%*
Italian-PoSTWITA	86,03%	81,78%	87,25%	83,07%	87,24%	83,40%	**87,65%**	**83,87%**	*87,33%*	*83,35%*
Japanese-GSD	95,06%	93,73%	**95,55%**	**94,27%**	93,32%	90,83%	92,53%	89,85%	*87,18%*	*84,14%*
Korean-GSD	87,70%	84,24%	**89,38%**	**86,05%**	79,81%	70,33%	80,21%	70,56%	*85,66%*	*80,88%*
Korean-Kaist	88,42%	86,48%	**89,35%**	**87,54%**	86,00%	82,63%	87,15%	83,79%	*87,17%*	*84,82%*
Norw.-Bokmaal	92,39%	90,49%	93,78%	92,19%	93,81%	92,20%	94,28%	**92,78%**	***94,34%***	*92,70%*
Norw.-Nynorsk	80,09%	75,04%	82,64%	78,08%	86,55%	83,16%	**87,02%**	**83,68%**	*86,57%*	*83,14%*
Norw.-NynorskLIA	68,08%	60,07%	**71,42%**	64,12%	70,87%	**64,41%**	70,75%	64,23%	*70,95%*	*64,24%*
Polish-LFG	96,58%	94,76%	**97,44%**	**96,03%**	96,22%	94,11%	96,38%	94,47%	*96,28%*	*94,05%*
Polish-SZ	93,39%	91,24%	**95,73%**	**94,25%**	94,11%	89,67%	94,18%	89,63%	*93,61%*	*89,18%*
Portuguese-Bosque	91,36%	89,04%	**92,69%**	**90,70%**	91,22%	87,66%	92,12%	88,64%	*89,77%*	*84,83%*
Portuguese-GSD	93,01%	91,63%	**94,74%**	**93,71%**	94,44%	92,92%	94,54%	93,13%	*94,18%*	*92,35%*
Romanian-Nonst.	89,12%	84,20%	89,61%	84,78%	88,71%	83,73%	**89,88%**	**85,32%**	*89,21%*	*83,28%*
Romanian-RRT	91,31%	86,74%	92,41%	88,05%	92,47%	88,18%	**92,72%**	**88,46%**	*92,68%*	*88,27%*
Russian-GSD	88,15%	84,37%	90,74%	**87,51%**	90,10%	85,56%	**91,16%**	87,03%	*90,76%*	*86,38%*
Russian-SynTagRus	93,80%	92,32%	94,92%	**93,68%**	94,45%	92,68%	**95,21%**	93,62%	*94,41%*	*92,65%*
Russian-Taiga	75,45%	69,11%	80,74%	75,65%	82,94%	77,08%	**84,91%**	**79,27%**	*83,42%*	*77,34%*
Serbian-SET	92,70%	89,27%	94,57%	91,65%	95,24%	91,85%	**96,14%**	**92,52%**	*95,87%*	*92,49%*
Slovenian-SSJ	92,96%	91,16%	**94,81%**	**93,49%**	94,14%	92,55%	94,43%	93,14%	*94,04%*	*92,51%*
Slovenian-SST	73,51%	67,51%	**77,23%**	**71,79%**	77,12%	71,06%	76,65%	71,06%	*76,78%*	*70,67%*
Spanish-AnCora	92,34%	90,26%	**93,75%**	**92,03%**	93,31%	91,21%	93,41%	91,38%	*93,51%*	*91,44%*
Spanish-GSD	90,71%	88,03%	**92,32%**	**90,11%**	91,26%	87,76%	90,96%	87,65%	*89,37%*	*85,17%*
Swedish-LinES	86,07%	81,86%	88,16%	84,55%	89,43%	85,98%	**90,05%**	**86,91%**	*89,63%*	*86,21%*
Swedish-Talbanken	89,63%	86,61%	92,42%	90,16%	92,92%	90,64%	**93,03%**	**90,90%**	*92,74%*	*90,40%*
Ukrainian-IU	88,29%	85,25%	91,65%	89,36%	93,40%	91,37%	**93,89%**	**91,89%**	*93,35%*	*91,30%*
Average	88.88%	85.60%	**91.10%**	**88.26%**	90.37%	86.96%	90.72%	87.42%	*90.48%*	*87.16%*

Table 3.1: Results in terms of UAS and LAS for UDPipe 2.0, UDPipe 2.0++ with the different embeddings, U-DepPLLaMA 2 7B and 13B with Gold Standard (GS) Tokenization and the End-to-End version of U-DepPLLaMA 2 7B. These last results are in italics as they are not comparable with the others because the End-to-End model is trying to solve a much more complex task. The best result for each treebank is highlighted in bold.

wegian, Romanian, Russian, Serbian, Swedish, and Ukrainian. Despite these improvements, performance drops are notable for languages such as Korean, Chinese, and Finnish, which

are significantly different from English, the primary pre-training language for LLaMA models. For example, in Chinese-GSD, `U-DepPLLaMA` lags behind UDPipe2.0++ by over 4%, and by nearly 10% for Korean-GSD. Exploring models with balanced linguistic exposure during pre-training could mitigate such limitations. The performance of `U-DepPLLaMA` 7B End-to-End, which directly processes raw sentences into dependency trees without relying on GS tokenization, is also reported in Table 3.1 (in italics). Notably, this model matches, and in some cases even surpasses, the results of its GS-tokenized counterpart. This outcome suggests that the language model may perform better on unaltered, real-world text, free from artificial tokenization constraints. Moreover, the added task complexity may help mitigate overfitting effects, encouraging better generalisation. The final row of Table 3.1 reports the mean UAS and LAS scores for each model. While UDPipe2.0++ remains the leading solution, the `U-DepPLLaMA` models are highly competitive. The comparison highlights the trade-off between UDPipe, a task-specific, feature-engineered approach, and the universal architecture of LLaMA, which generates dependency trees consistently across languages without requiring modifications. Across 50 treebanks, both architectures achieve top performance in 25 cases, demonstrating the potential of embedding syntactic expertise into language models for synthetic outputs.

However, the computational demands of LLaMA models, both during training and inference, are significantly higher than those of UDPipe, due to their larger parameter sizes. To ensure applicability, the dependency tree output by the model must form a valid tree structure. For the 7B parameters model, this requirement fails in 0.45% of generated sentences, resulting in those sentences being discarded. This error rate decreases to 0.31% for the 13B model. Without GS tokenization, the error rate for the 7B model rises to 1.44%. Larger models may reduce these issues, but further experimentation is required.

Occasionally, the model experiences hallucination issues: while it does not add new terms, it may omit some words or switch their order, particularly in non-projective tree cases (as discussed earlier). Most of these errors can be corrected using Algorithm B.1, which restores word order while preserving dependency relations. Missing words can be connected directly to the root node of the tree, which is always present. Such recovery measures are required in less than 7% of cases. However, a more in-depth manual analysis is needed to confirm whether non-projectivity is consistently the root cause of these issues.

An in-depth error analysis is reported in the Appendix B.2, while an error based on the sentence level is reported in Appendix B.3.

The ability of `U-DepPLLaMA` to handle multiple languages without task-specific architecture modifications marks a significant advancement, achieving state-of-the-art results in dependency parsing across various languages. These models (`DepPLLaMA` and `U-DepPLLaMA`)

offer several key insights into the application of LLMs for dependency parsing: *i)* the use of low-rank adapted parameters in `U-DepPLLaMA` demonstrates the viability of scaling models to handle multilingual data without drastically increasing computational demands; *ii)* the broader training dataset in `U-DepPLLaMA` enhances its ability to generalise across syntactically diverse languages, leading to improved accuracy and adaptability; *iii)* both models illustrate the adaptability of language models for parsing tasks, while `U-DepPLLaMA` highlights the potential for universal, task-agnostic designs that do not require specialised architectures per language. However, the vast capabilities of such models suggest a natural extension beyond syntactic parsing. A real-world system, such as a robot, would likely be tasked with more than just syntactic analysis. In the next section, we explore how to instruction-tune a Large Language Model for a multi-task scenario, encompassing tasks such as Sentiment Analysis, Information Extraction, and Discourse Coherence.

3.2 Multi-task scaling to the extreme: ExtremITA

`U-DepPLLaMA` is a strong model for parsing sentences and producing parenthetic forms as output. It exploits a consolidated Large Langauge Model to do so and has been fine-tuned using the Q-LoRA approach. However, in more realistic scenarios, we would expect a system to solve more tasks and to adapt its responses based on the different input it receives .Ideally, a model should handle diverse instructions, whether stylistic (e.g., responding in rhymes or biblical terms) or functional (e.g., constrained answer length or specific output formats), i.e. it should be able to follow *instructions*. This word has become pivotal in the last years, as models more and more able to follow instructions. Researchers haven even coined a term: Instruction-Tuning. It refers to the process where a model is fine-tuned to follow instructions, that can be either stylistic, such as using biblical terms, or answering in rhymes or just a constraint about the length of the answer. And this methodology gave birth to ExtremITA [109].

The generative power exhibited by models like GPT [216] and GPT3 [40], as well as the recent development of LLaMA [261] foundational models, has opened up new avenues for leveraging the concept of "prompting". This approach allows for modeling inductive tasks by formulating them linguistically, as natural language queries or instructions, enabling the model to provide accurate responses based on it. By combining the power of these models and the prompting technique, complex tasks can be addressed using a straightforward and intuitive interaction paradigm, without the need for task-specific feature engineering or neural architectures. This is a contraposition and a further expansion of `U-DepPLLaMA`, where a single Large model was used to solve a single task.

`ExtremITA` was developed for the EVALITA 2023 campaign [146], which is dedicated to advancing Natural Language Processing and speech technologies for the Italian language. Since its beginning in 2007, EVALITA has provided a structured framework for evaluating and comparing diverse systems across a spectrum of tasks pertinent to Italian linguistic processing. By facilitating consistent evaluation practices, EVALITA plays an important role in the creation of standardized resources and tools, thereby promoting the growth and refinement of NLP and speech technologies in the Italian context. Its primary objective is to foster the development of language and speech technologies tailored for Italian by offering shared tasks that encourage collaboration and innovation within the research community. These tasks encompass a wide array of challenges, including sentiment analysis, named entity recognition, and hate detection, among others. Traditionally, participants choose one task and participate by optimizing a specialised model to solve the specific task. Each shared task provides a set of data for the participants to use, such as training and evaluation, while the test set usually stays undisclosed. If participating in multiple tasks, researchers typically fine-tune distinct models for each. The approach of `ExtremITA` is extreme: inspired by the seminal work of [55] on instruction-tuned models, we developed a single monolithic architecture capable of solving all linguistic tasks simultaneously. To the best of my knowledge, this is the first attempt to apply such an approach in the context of EVALITA.

The objective of this work is to push the limits of LLMs by proposing a single model capable of tackling a wide array of heterogeneous tasks. Adapting a Large Language Model using techinques such as LoRA [119] enables the model to be fine-tuned on the downstream tasks. Importantly, in robotics one expects these model to solve a plethora of tasks, mainly linguistics, with a particular focus on the Situated topic, which will be further explored in the next Chapter. It leverages a Decoder-only model, trained on the union of all the available datasets for the challenge. By adopting a multi-task learning framework, the objective is to evaluate the applicability of a single model in effectively solving multiple tasks at once. Notably, the approach offers a significant advantage as it enables the resolution of diverse tasks by employing a unified architecture and fine-tuning based on input-output pairs. The EVALITA challenge serves as a robust testing ground for assessing the capabilities of LLMs across various Italian linguistic tasks, without any specific architectural requirement. Instead, the model is triggered with task-specific prompts, such as "*Is there any mention of a conspiracy in this text? Answer yes or no.*" or "*How much consistent is this sentence, on a scale of 0 to 5?*".This methodology differs significantly from traditional solutions, such as those for Sentiment Analysis [29, 46], Conspiracy Detection [237], or Semantic Role Labelling [30], which rely on task-specific models and architectures.

The complete list of tasks in which the `ExtremITA` approach participated is here reported,

in a wide range of semantic dimensions, including Affect Detection, Authorship Analysis, Computational Ethics, Named Entity Recognition, Information Extraction, and Discourse Coherence:

- **EMit** - Categorical Emotion Detection in Italian Social Media [15]: it presents two subtasks: *(i)* Subtask A, that consists in an emotion detection challenge, and *(ii)* Subtask B, that introduces a novel problem of target detection of the expressed emotion.
- **EmotivITA** - Dimensional and Multi-dimensional Emotion Analysis [85]: EmotivITA is a task of emotion regression based on EmoITA, the Italian version of the EmoBank dataset.
- **PoliticIT** - Political Ideology Detection in Italian Texts [224]: aims to extract political ideology information from Italian texts. For this, an automatic document classification task on clusters of texts is proposed. It consists of extracting the self-assigned gender as demographic trait, and the political ideology as a psychographic trait from a set of texts written in Italian from several authors that share those traits.
- **GeoLingIt** - Geolocation of Linguistic Variation in Italy [218]: a shared task on geolocation of linguistic variation in Italy from social media posts exhibiting non-standard Italian language. The task is meant to both advance natural language processing (NLP) in dealing with non-standard Italian language, and inform sociolinguistics with language variation insights derived from large-scale, quantitative analysis.
- **LangLearn** - Language Learning Development [11]: automatic language development assessment. The task consists of predicting the relative order of two essays written by the same student.
- **HaSpeeDe 3** - Political and Religious Hate Speech Detection [145]: explores hate speech in strong polarised debates, concerning in particular politics and religion. It is articulated in two different tasks: *A)* In-domain political hate speech detection and *B)* Cross-domain hate speech detection about political and religious tweets.
- **HODI** - Homotransphobia Detection in Italian [197]: a shared task that aims to identify Italian homotransphobia on Twitter. This will allow researchers to investigate a phenomenon that has received little attention from the worldwide NLP community and has never been built for the Italian one. It is critical to comprehend not only whether a text is hateful, but also why.

- **MULTI-Fake-DetectiVE** - MULTImodal Fake News Detection and VErification [36]: it focuses on the automatic detection of fake news in a multimodal setting including texts and images. The goal of the task is twofold: *i)* given a piece of content (e.g., a social media post or a news article) that includes both a visual and a textual component, determining its likelihood of being a real or a fake news; *ii)* understanding how the visual and textual components of news can influence each other, i.e., given a text and an accompanying image, is the combination of the two aimed at misleading the interpretation of the reader about one or the other, or not?

- **ACTI** - Automatic Conspiracy Theory Identification [225, 226]: identification of conspiracy theories, which have diffused dangerous ideas generating societal harm. This task aims to recognize and categorize brief texts from platforms such as Telegram, 4chan, and Parler, into conspiracy theory categories.

- **NERMuD** - Named-Entities Recognition on Multi-Domain Documents [205]: consisting in the extraction and classification of named-entities in a document, such as persons, organizations, and locations, both domain-specific and domain-agnostic data.

- **CLinkaRT** - Linking a Lab Result to its Test Event in the Clinical Domain [10]: is a relation extraction task based on clinical cases taken from the E3C corpus, i.e. on Italian written documents reporting statements of a clinical practice (thus including, for example, the reasons for a clinical visit, the physical exams undertaken, the assessment of the patient's diagnosis and subsequent treatments). The task consists in identifying test results and measurements and linking them to the textual mentions of the laboratory tests and measurements from which they were obtained.

- **WiC-ITA** - Word-in-Context task for Italian [45]: the general goal of the WiC-ITA task is to establish if a word w occurring in two different sentences s_1 and s_2 has the same meaning or not.

- **DisCoTEX** - Assessing DIScourse COherence in Italian TEXts [41]: it is conceived to address two distinct scenarios: a more traditional one, aimed at evaluating whether models are able to distinguish well-organized documents from corrupted ones; a less explored one, which assesses the performance of models on texts evaluated for coherence by human raters.

The aforementioned 13 tasks comprised 22 subtasks, where the proposed models ranked first in 9 subtasks (41%), and achieved a top-three position in 14 subtasks (64%). By framing tasks as natural language queries and leveraging LLaMA-based models, `ExtremITA`

achieved high performance while avoiding the need for task-specific architectures. This work underscores the potential of instruction-tuned LLMs to address heterogeneous challenges in NLP efficiently.

3.2.1 The Multi-task Prompting Approach

A first effective application of an Encoder-Decoder architecture in a multi-task scenario is presented in [217]. Specifically, the pre-training process of the T5 model involves training the model on a large corpus of diverse text data from sources such as books, articles, websites, and data related to tasks like machine translation, classification, and regression. During pre-training, T5 employs a denoising objective, similar to other popular Transformer-based models like BERT and GPT. This objective requires the model to reconstruct masked or corrupted input text, enabling it to learn meaningful representations and capture contextual information effectively. One of the key strengths of T5 is its versatility. By casting various NLP tasks into a text-to-text problems, it can be fine-tuned for specific tasks simply by adding a prefix that describes the task, along with appropriate input-output pairs during fine-tuning. In practice, such an architecture can be triggered by concatenating the name of the task it is trained on with an input text, prompting the model to generate the corresponding solution, e.g., a class label in a classification task or a text span question answering. This flexibility eliminates the need for task-specific architectures or modifications, simplifying the application of T5 to diverse scenarios. Recently, T5 has been applied to hundreds of tasks [55], and large-scale systematic pre-training efforts [54] have demonstrated its effectiveness "zero-shot" or "few-shot" learning scenarios.

Decoder-only models, on the other hand, are designed to be triggered by natural language inputs, such as requests or text intended for processing. These models generate text one word at a time, producing outputs that align with the given prompt. Such models have the ability to essentially follow instructions, as exemplified by the recent release of ChatGPT. This characteristic holds a greater appeal, as tasks can be linguistically described using prompts, where the input sentence serves as contextual information. InstructGPT [203] is an extension of the GPT language model [40] explicitly designed to excel in multi-task scenarios when used with natural language prompts. It combines the power of language models with the ability to follow instructions provided in the form of natural language prompts. Unlike conventional language models that generate text freely, InstructGPT is fine-tuned using human feedback to understand and generate text based on a given prompt and to select the best sequence that humans prefer.

Another noteworthy instruction-tuning model is Alpaca [255], which builds upon the

Task Name	Natural language instruction
EMit A	"*Quali emozioni sono espresse in questo testo? Puoi scegliere una o più emozioni tra 'rabbia', 'anticipazione', 'disgusto', 'paura', 'gioia', 'amore', 'tristezza', 'sorpresa', 'fiducia', o 'neutro'.*"
EMit B	"*Di cosa parla il testo, tra 'direzione', 'argomento', 'entrambi', 'non specificato'?*"
EmotivITA	"*Scrivi quanta valenza è espressa in questo testo su una scala da 1 a 5, seguito da quanto stimolo è espresso in questo testo su una scala da 1 a 5, seguito da quanto controllo è espresso in questo testo su una scala da 1 a 5.*"
PoliticIT	"*Scrivi se l'autore del testo è 'uomo' o 'donna', seguito dalla sua appartenenza politica tra 'destra', 'sinistra', 'centrodestra', 'centrosinistra'.*"
GeoLingIt	"*Scrivi la regione di appartenenza di chi ha scritto questo testo, seguito dalla latitudine, seguita dalla longitudine.*"
LangLearn	"*Questi due testi separati da [SEP] sono presentati nell'ordine in cui sono stati scritti? Rispondi sì o no.*"
HaSpeeDe 3	"*In questo testo si esprime odio? Rispondi sì o no.*"
HODI A	"*In questo testo si esprime odio omotransfobico? Rispondi sì o no.*"
HODI B	"*Con quali parole l'autore del testo precedente esprime odio omotransfobico? Separa le sequenze di parole con [gap].*"
MULTI-Fake -DetectiVE	"*L'evento riportato nel testo è 'certamente vero', 'probabilmente vero', 'probabilmente falso', o 'certamente falso'?*"
ACTI A	"*In questo testo si parla di una cospirazione? Rispondi sì o no.*"
ACTI B	"*Di quale teoria cospirazionista parla questo testo, tra 'Covid', 'Qanon', 'Terrapiattista', 'Russia'?*"
NERMuD	"*Scrivi le menzioni di entità nel testo, indicandone il tipo: [PER] (persona), [LOC] (luogo), [ORG] (organizzazione).*"
CLinkaRT	"*Trova i risultati dei test e delle misurazioni nel testo. Per ogni risultato, scrivi '[BREL]', seguito dal risultato seguito da '[SEP]', seguito dal test, seguito da '[EREL]'. Se non trovi nessun risultato, scrivi '[NOREL]'.*"
WiC-ITA	"*La parola compresa tra [TGTS] e [TGTE] ha lo stesso significato in entrambe le frasi? Rispondi sì o no.*"
DisCoTEX 1	"*Le due frasi precedenti, separate da '[SEP]', sono coerenti tra loro? Rispondi sì o no.*"
DisCoTEX 2	"*Quanto è coerente questa frase, su una scala da 0 a 5?*"

Table 3.2: List of the natural language instruction definition for all tasks in the `ExtremITA` model. Notice that these instructions have not been heavily optimized against individual tasks, also due to time constraints during the EVALITA challenge.

LLaMA foundational models [261]. The authors of Alpaca created 175 sets of instructions paired with input sentences and corresponding outputs. These were expanded using GPT-3.5, resulting in a collection of approximately $52,000$ instruction examples. The LLaMA model was further fine-tuned using this extensive dataset, a process referred to as instruction-tuning. The outcome of this effort was the Stanford Alpaca [255] as an instruction-following LLaMA model. Recently, an Italian counterpart named Camoscio [230] underwent a similar instruction-tuning process to Alpaca on Italian data, essentially serving as the Italian equivalent. Also based on the same LLaMA model, it was instruction-tuned on the same 52.000 instructions which were automatically translated into Italian using ChatGPT [230]. This

adaptation has made Camoscio a viable instruction-following model for Italian. Following the discussion from the previous Section, the LoRA [119] approach was used in `ExtremITA` as well to adapt the model to solve the EVALITA tasks.

The original LLaMA model is pre-trained to execute instructions by leveraging a structured prompt that includes both a textual description of the task and a specification of the desired output format. For example, in the ACTI task, the provided instruction is "*In questo testo si parla di una cospirazione? Rispondi sì o no.*"[8]. The subsequent sentence to be evaluated is appended to this instruction. A complete list of such instructions is available in Table 3.2.

The decoder is then expected to continue the prompt by generating an appropriate response. For instance, given the input sentence "*ACTI: Hanno votato tutti obbligo vaccinale, green pass, persecuzioni varie*"[9], the model is expected to generate a simple output: either *Yes* or *No*.

Task	Output Templates
EMit A	{"*Rabbia*", "*Anticipazione*", "*Disgusto*", "*Paura*", "*Gioia*", "*Amore*", "*Tristezza*", "*Sorpresa*", "*Fiducia*"}$^{+}\vee$ "*Neutrale*"
EMit B	{"*Direzione*", "*Argomento*", "*Entrambi*", "*Non specificato*"}
EmotivITA	"*Valenza: {0-5} Stimolo: {0-5} Controllo: {0-5}*"
PoliticIT	"*Gender*: {"*Uomo*", "*Donna*"} *PIB*: {"*Sinistra*", "*Destra*"} *PIM*: {"*Sinistra*", "*Destra*", "*Centro Sinistra*", "*Centro Destra*"}"
GeoLingIt	"*Regione*: {*Abruzzo*, .., *Veneto*} *Latitudine*: {} *Longitudine*: {}"
LangLearn*	{"*Corretto*", "*Non Corretto*"}
HaSpeeDe 3*	{"*Odio*", "*Non Odio*"}
HODI A*	{"*Omotransfobico*", "*Non Omotransfobico*"}
HODI B	<HOMOTRANSPHOBIA_MENTION>
MULTI-Fake -DetectiVE	{"*Certamente Falso*", "*Probabilmente Falso*", "*Probabilmente Vero*", "*Certamente Vero*"}
ACTI A*	{"*Cospirazione*", "*Non Cospirazione*"}
ACTI B	{"*Terrapiattista*", "*Covid*", "*Qanon*", "*Russia*"}
NERMuD	[<ENTITY_TYPE>] <TEXT_SPAN_THAT_EVOKES_ENTITY>
CLinkaRT	"[*BREL*] <RML_ENTITY_MENTION> [*SEP*] <EVENT_ENTITY_MENTION> [*EREL*]"
WiC-ITA*	{"*Uguale*", "*Differente*"}
DisCoTEX 1*	{"*Coerente*", "*Non Coerente*"}
DisCoTEX 2	{*0-5*}

Table 3.3: Output templates for the `ExtremITA` model. In EMit A the model is requested to generate one or more labels from the first set ($^{+}$) or the text "*Neutrale*" if no emotion is expressed. In the tasks with * the model is requested to respond with {"*Si*", "*No*"}, for more details see Table 3.2.

[8] In English, "*Is there any mention of a conspiracy in this text? Answer yes or no.*"
[9] In English, "*They all voted compulsory vaccination, green pass, persecution*"

The output of the `ExtremITA` model depends on the specific task. For a complete overview of output formats across tasks, please refer to Table 3.3. Below are examples of the various output formats used in different tasks:

- In classification tasks involving only one label (such as EMit B, LangLearn, HaspeeDe 3, HODI A, MULTI-Fake-DetectiVE, ACTI A, ACTI B, WiC-ITA and DisCoTEX 1) the output is just the label of the target class. In the above example, the output would be "*Cospirazione*" as the input text reflects some conspiracy theory.

- In multi-label tasks, such as PoliticIT [224], where a text is expected to be associated with the gender and the political inclination of the author, multiple labels reflecting such different dimensions are used, e.g., "*uomo sinistra centro-sinistra*". In EMit A [15] where multiple emotions can be triggered, these are provided as a sequence of labels.

- For regression tasks, such as EmotivITA [85] and DisCoTEX 2 [41], the output is the number to be predicted within a specific range.

- In GeoLingIt [218], the model is requested to determine the region of origin of the tweet and the corresponding coordinates (latitude and longitude) based solely on the text. For example, if the input sentence is "*Daje che je 'a famo!*", the model should provide the answer "*Lazio 41.8984164 12.54514535*", considering the use of the typical Roman dialect. This particular task combines both multi-label classification and regression, as it requires determining the region (classification) and providing the precise coordinates (regression) simultaneously.

- In HODI B [197] where the span of the offending text is expected to be extracted, it is simply requested as output.

- In NERMuD [205], the list of expected Named Entities is reported as a sequence of text spans, each associated with the corresponding entity type.

- CLinkaRT [10] focuses on extracting the names of medical tests performed on patients from an input text and linking them to the corresponding test results, treating it as a Relation Extraction problem. Here the relations are encoded with a slightly more complex form to summarize a list of relations, each associating an EVENT with a corresponding measure (or RML); as an example, the sentence "*Il PSA aumentava da 2 a 62 ng/ml.*" is associated with "[*BREL*] *2* [*SEP*] *PSA* [*EREL*] [*BREL*] *62 ng/ml* [*SEP*] *PSA* [*EREL*]" (where 2 and 62 reflect the RML while PSA is the test event).

3.2.2 Experimental Setup and Evaluation

The evaluation section aims to assess the effectiveness of a unified large language model (LLM) in handling a diverse set of natural language processing (NLP) tasks in Italian. The primary objective is to determine whether a single, instruction-tuned decoder-only LLM can achieve competitive performance across multiple tasks without task-specific architectural modifications. This approach sought to streamline NLP workflows by reducing the need for bespoke models for each task, thereby enhancing scalability and efficiency. This ability is reflected in the evaluation score of each task, as reported hereafter.

The model was trained on the unified dataset of all the tasks of EVALITA. Generally, one example in an EVALITA task corresponds to an example in this learning setting. Below are some exceptions where datasets underwent preprocessing or augmentation:

- **ACTI**: The dataset was expanded by incorporating positive examples (those involving conspiracy theories) from the ACTI B dataset into ACTI A, and vice versa. This process increased the number of examples from 460 to 1,909 for subtask A and from 300 to 777 for subtask B.

- **CLinkaRT**: The available medical reports were segmented into smaller parts using the Spacy library, respecting sentence boundaries. Each segment contained between 50 characters and 30 words. The decision of the model for each report was derived from the combined predictions of its segments. Additionally, examples from the *TESTLINK@IberLEF 2023* dataset[10], containing Spanish medical reports, were used to augment the dataset. Despite the language difference, these texts exhibited language-invariant phenomena related to events and measures, making them useful for augmentation. This process expanded the dataset from 83 large documents to 3,903 shorter sentences, recovering over 95% of annotated relations.

- **EMit**: Emojis in the dataset were converted into textual descriptions to improve compatibility with language models.

- **GeoLingIt**: Tasks A and B were merged to allow simultaneous prediction. By generating both answers in the same process call, the shared context was expected to enhance performance.

- **HODI B**: Only sentences expressing homotransphobia were retained, reducing the dataset from 5,000 to 1,914 examples. Empty examples were excluded as they were deemed uninformative for training.

[10] https://e3c.fbk.eu/testlinkiberlef

- **LangLearn**: Sentences were truncated to a maximum of 100 tokens, and additional examples were generated by flipping sentence pairs (e.g., swapping positive and negative labels). This augmented the dataset from $3,377$ to $6,438$ examples.
- **MULTI-Fake-DetectiVE**: Since the model is text-only, images were excluded. Duplicate examples (i.e., identical text paired with different images) were removed, reducing the dataset from $1,058$ to 860 examples.
- **NERMuD**: The dataset was transformed from a token classification task into a sequence-to-sequence format, where the model outputs only the identified entities along with their classes.
- **PoliticIT**: Texts were divided into sentences of up to 200 tokens, making them more manageable for the language model. A majority voting strategy was used at classification time to determine the final class (gender and political inclination) for all sentences authored by the same individual.
- **WiC-ITA**: Positive examples were augmented by flipping sentence orders while maintaining their labels to rebalance class distribution. This increased the dataset size from $5,610$ to $6,600$ examples.

Overall, the entire dataset is composed of a total of $134,018$ examples[11]. The model underwent 2 epochs of training with a learning rate of $3 \cdot 10^{-4}$, employed a batch size of 32 and a linear scheduler with warmup was applied, utilizing a warmup ratio of 0.1. The training process utilized LoRA to refine the W_q, W_k, W_v and W_o modules of the transformer (as previosuly detailed), incorporating a matrix rank $R = 8$ and a parameter $\alpha = 16$ for the LoRA matrices. The decoding strategy in the generation phase used a beam search equal to 4, temperature of 0.2, with a top probability of 0.75 amongst the first 40 candidates. A single Tesla T4 GPU with 16GB of memory was used for this model and its training duration exceeded 144 hours. The training data was divided into a 95% training set and a 5% validation set initially for hyper-parameter optimization. The test set was used from each shared task from the competition.

Results and Discussion. The experimental results are summarised in Table 3.4. Tasks are categorised by subtask, followed by the evaluation metric, and the scores and ranks achieved by `ExtremITA` compared to the best-performing competitor. For each subtask, the best-performing method is highlighted in bold. The system ranked first in 9 out of

[11]The code made available at https://github.com/crux82/ExtremITA allows replicating the data generation process.

22 subtasks (41%) in the EVALITA 2023 competition. Furthermore, `ExtremITA` secured a top-three position in 14 subtasks (64% of all subtasks). Despite these successes, `ExtremITA` encountered challenges in certain tasks, including GeoLingIt, LangLearn, and WiC-ITA. These tasks require detecting and analysing nuanced changes in writing style (e.g., author-specific features) or the contextual meaning of words. Since the monolithic architecture was primarily designed for tasks such as sentence classification or rewriting spans of input text (e.g., HODI), it showed limitations in these more specialised scenarios.

Enhancing the prompting phase by optimising task-specific prompts could improve performance. Reformulating certain tasks differently is another potential avenue. For instance, in GeoLingIt, keeping the two subtasks (region classification and coordinate prediction) distinct, as in the original shared task, could yield better results by reducing task complexity during generation. Another area for improvement is incorporating real-world knowledge into the system. For example, in GeoLingIt, the generated coordinates strongly depend on the region from which a tweet originates. The model lacked explicit constraints on the coordinates it could generate, relying solely on patterns within the training data. Injecting geographical knowledge, such as region boundaries, during generation could significantly enhance accuracy. This approach proved essential for the winning system [138], which achieved a lower error distance in kilometres by employing a rectification module. This module adjusted predictions falling outside land boundaries to the nearest point within the predicted boundaries of a region, resulting in a more accurate and realistic output.

On the other hand, the zero-shot application of Camoscio demonstrated poor performance across all tasks. This outcome was expected, as the model was used without additional fine-tuning, relying solely on the original instruction-tuning of the $52,000$ Alpaca instructions. Most often, the model either predicted the most frequent class or hallucinated by rewriting portions of the input text. These results highlight the necessity of instruction-tuning tailored to the specific tasks using natural language descriptions. Such tuning is essential to guide the model towards generating accurate and meaningful answers, especially in Italian.

The results also bring attention to the computational cost of both training and inference. Training the 7-billion-parameter model took over 144 hours, and during inference, the model could process only 2 to 3 sentences per second. This substantial delay in processing speed makes the model less practical for real-world applications, despite its strong performance across a wide array of tasks. Furthermore, the number of parameters in the model typically exceeds those of competing task-specific solutions by one order of magnitude, further contributing to its resource-intensive nature.

Overall, the results are remarkable, especially considering that no task-specific architectural designs were applied. Instead, a single LLM was used, achieving competitive perfor-

Task	SubTask	Eval metric	Camoscio 0-shot Score	ExtremITA Score	ExtremITA R	Best Competitor Score	Best Competitor R
EMit	A	F1	0.0092	**0.6028**	**1**	0.5086	2
	B	F1	0.1325	**0.6459**	**1**	0.6331	2
EmotivITA	B	Pears Val	0.0000	**0.8110**		0.8110	
		Pears Aro	0.0931	**0.6330**	**1**	0.6520	2
		Pears Dom	0.0000	**0.6300**		0.6540	
PoliticIT	-	F1	0.2965	0.7719	3	**0.8241**	**1**
GeoLingIt	A	F1	0.0205	0.3818	11	**0.6630**	**1**
	B	Avg Km	280.14	145.15	9	**97.74**	**1**
LangLearn	COWS	F1	na	0.5500	8	**0.7500**	**1**
	CITA	F1	na	0.6100	8	**0.9300**	**1**
HaSpeeDe 3	A	F1 - text.	0.3333	0.9034	3	**0.9128**	**1**
		F1 - context.	0.3333	0.9034	3	**0.9128**	**1**
	B	F1 - xRel.	0.4558	**0.6525**	**1**	0.6461	2
		F1 - xPolitic.	0.3333	0.9034	3	**0.9128**	**1**
HODI	A	F1	0.3284	0.7942	5	**0.8108**	**1**
	B	F1	0.4790	**0.7228**	**1**	0.7051	2
MULTI-Fake-DetectiVE	A	F1	0.3800	0.5070	2	**0.5120**	**1**
	ATD	F1	0.2900	**0.4640**	**1**	0.4600	2
ACTI	A	F1	0.3306	0.8565	2	**0.8571**	**1**
	B	F1	0.1603	0.8556	5	**0.9123**	**1**
NERMuD	DAC	F1	na	**0.8900**	**1**	na	na
CLinkaRT	-	F1	na	0.5916	2	**0.6299**	**1**
WiC-ITA	A	F1 it-it	0.3333	0.5100	10	**0.7300**	**1**
		F1 it-en	0.3333	0.5400	8	**0.7400**	**1**
	B	F1 all	0.3333	0.5100	10	**0.7300**	**1**
DisCoTEX	1	Acc	na	**0.8150**	**1**	0.7200	2
	2	HM*	na	**0.6500**	**1**	0.6300	2

Table 3.4: ExtremITA ranks and results. Here each task is divided into the subtasks we participated in. We reported Camoscio in 0-shot (i.e. no training), the `ExtremITA` model and as a comparison the best competitor (either that won or placed higher in the ranking). In bold the rank and the scores of the winning systems. The HM* measure for the DisCoTEX task refers to the Harmonic Mean between Pearson's and Spearman's correlations. The "*na*" value is due to missing the official evaluation scripts for the specific tasks at the time of writing.

mance on nearly all tasks. The key to success lies in effectively prompting the model with natural language instructions or employing task-specific encoding techniques for outputs. Future improvements could include leveraging larger LLMs, such as LLaMA 65B or the recently introduced LLaMA 3.1 405B, which might offer even higher performance. Additionally, to enable a more comprehensive evaluation and optimisation, a broader exploration of

architectures and hyperparameters would have been beneficial. Unfortunately, the time constraints imposed by the EVALITA deadlines and the substantial effort required to complete all 13 tasks in parallel limited the ability to perform a thorough hyperparameter search. With more time and resources, fine-tuning larger models, refining task-specific prompts, and systematically optimising hyperparameters could further enhance the capabilities of the model and address its limitations.

Task	SubTask	Eval metric	TOTAL examples	0	100	500	1000	All
EMit	A	F1	5688	0.0092	0.2438	0.3526	0.4440	0.6028
	B	F1		0.1325	0.4013	0.5797	0.4450	0.6459
EmotivITA	B	Pears Val	7608	0.0000	0.2115	0.6399	0.7043	0.8110
		Pears Aro		0.0931	-0.0218	0.3447	0.4398	0.6330
		Pears Dom		0.0000	0.1836	0.4441	0.5627	0.6300
PoliticIT	-	F1	16047	0.2965	0.3129	0.5956	0.5548	0.7719
GeoLingIt	A	F1	13669	0.0205	0.1163	0.1351	0.1523	0.3818
	B	Avg Km		280.14	277.32	281.14	267.48	145.15
LangLearn	COWS	macro-F1	320	0.3347	0.5428	0.4875	0.4804	0.5739
	CITA	macro-F1	307	0.3326	0.3725	0.4845	0.4918	0.5954
HaSpeeDe 3	A	F1 - text.	5340	0.3333	0.4212	0.7239	0.8498	0.9034
		F1 - context.		0.3333	0.4212	0.7239	0.8498	0.9034
	B	F1 - xRel.		0.4558	0.5079	0.6199	0.6500	0.6525
		F1 - xPolitic.		0.3333	0.4212	0.7239	0.8498	0.9034
HODI	A	F1	4770	0.3284	0.3653	0.6126	0.6606	0.7942
	B	F1	1914	0.4790	0.4831	0.5782	0.6122	0.7228
Multi-Fake-Detective	A	F1	860	0.3800	0.4200	0.4400	0.4700	0.5070
	ATD	F1		0.2900	0.4200	0.3900	0.4700	0.4640
ACTI	A	F1	1909	0.3306	0.6129	0.7850	0.7944	0.8565
	B	F1	777	0.1603	0.3691	0.8471	0.8440	0.8556
WiC-ITA	A	F1 it-it	5610	0.3333	0.3333	0.3333	0.3333	0.5100
		F1 it-en		0.3333	0.3378	0.3333	0.3333	0.5400
	B	F1 all		0.3333	0.3333	0.3333	0.3333	0.5100

Table 3.5: Scores of different `ExtremITA` models, where 0, 100, 500 and *All* refer to the number of examples the models were trained on. Not all the tasks from EVALITA are reported in this table as some official evaluation scripts are missing at the time of writing and thus the evaluation could not be performed.

The Impact of different training set sizes. The `ExtremITA` model was trained using about one hundred thousand examples, with a significant portion (30-40%) derived from the NERMUD dataset. While LLaMA and its derived models, Alpaca and Camoscio, are intended to be used in a 0-shot manner, we conducted additional instruction tuning using the EVALITA data. In order to evaluate the generalization capabilities of the model across

these domains and its reliance on specific data, we conducted more in-depth experiment to analyze its learning curve based on different training set sizes. For this experiment on the EVALITA data, we followed the methodology outlined previously for instruction-tuning, with a differnce: we utilized up to 100, 500, and 1000 examples for each task, whenever available. Additionally, we used the 0-shot application of Camoscio as is from before as a comparison, i.e. using 0 examples with no additional instruction-tuning. Table 3.5 displays the results for selected tasks, for which official evaluation scripts were available, categorized by the trained models: 0, 100, 500, 1000 and the "*All*" model, which corresponds to `ExtremITA` as it was trained using *all* the available data. This analysis provides insights into the performance of the model across different levels of training data availability and highlights the benefits of instruction-tuning for domain-specific tasks.

Unsurprisingly, the "*All*" model consistently outperforms others acrross all tasks. For example, in the EMit task, where the model achieved first place in both subtasks, we observe a clear improvement in performance as the training set increases from 0 to 1000 examples. This suggests that performance gains are directly correlated with the size of the training set. Similar trends are observed in other tasks, such as EmotivITA, HaSpeeDe3, HODI, and ACTI. However, some tasks, such as WiC-ITA, remain more challenging, with models frequently defaulting to predicting the most frequent class. Similarly, in GeoLingIt, even with 1000 examples, the performance remains low, indicating that this number of examples is insufficient to capture the complex relationships between the input and output. For the LangLearn task, the evaluation metric was switched from F1 of the positive class (used in the official script and Table 3.4) to macro-average F1. This adjustment better captures the performance of biased models. Notably, the official metric for this task is artificially high for the 0-shot model, which always outputs the positive label, while the macro-F1 is more sensible with respect to this aspect. Conversely, the 0-shot application of Camoscio performed the poorest. While designed to comprehend instructions, respond to queries, and provide argumentation for its decisions in natural language, Camoscio often fails to strictly adhere to the instructions provided in the prompt. This inconsistency leads to deviations in the generated output and reduces accuracy when classifying examples. These results highlight the significant challenges presented by the EVALITA tasks and emphasize the necessity of instruction-tuning for enhancing the performance of Large Language Models (LLMs). Even with computationally efficient models like LLaMA 7B, instruction-tuning using a sufficient number of representative examples remains essential for achieving competitive results in complex tasks. Of course, a more thorough evaluation is needed for deriving more significant conclusions, but this simple application is a first result.

To explore the few-shot learning capabilities of LLMs, we experimented with a limited

number of input-output pairs embedded in the prompt, without training. The HaSpeeDe 3 task, involving binary classification to detect hate speech, was selected for this evaluation. Using the Camoscio model without additional instruction-tuning, we modified the prompt to include two examples: one positive example and one negative example, randomly-selected from the training set. The resulting prompt included the task description (see Table 3.2) to which was added the following structure:

"*This text does not contain hate: <Negative Example>, and you should answer no. This text contains hate: <Positive Example>, and you should answer yes.*".

The current example to be predicted was then appended to the prompt. This few-shot version achieved an F1 measure of 0.4486, surpassing the fine-tuned version on 100 examples by more than 2%. This result demonstrates the few-shot learning capabilities of LLMs as they exhibit a remarkable ability to capture dependencies and relationships with just minimal examples. Moreover, we evaluated the use of prompts by the 2-shot strategy over the model instruction-tuned on 100, 500, and 1000 examples, with results of 0.60, 0.67, and 0.83, respectively: this thus shows that a good increase in performance is observed at 100 examples (from 0.42 to 0.60) but then slightly poorer performances are obtained. Applying the same strategy on the "ACTI" task, which involves detecting if a text concerns a conspiracy theory, yielded different results: the 2-shot Camoscio model achieved an improvement up to 0.4819 of F1, while all the other versions (instruction-tuned on 100, 500 and 1000 examples) exhibited a drop in performance. This discrepancy may arise from the inherent complexity of recognising conspiracy elements in brief texts, which could demand capabilities beyond those of the current models, even in a few-shot setup. To validate these findings conclusively, further comprehensive experimentation is necessary. However, this is the first application of such strategies on the EVALITA data and we believe that this paves the way for diverse evaluations and prompting engineering tests of such models.

Error Analysis. To gain deeper insights into the inner workings of the models, an error analysis is conducted on two tasks: one where the system performed exceptionally well and another where it performed poorly.

We selected EmIt Task A, where `ExtremITA` ranked first in the official rankings, as the high-performing task. This task involves multi-label classification, assigning one or more of eight emotions defined by Plutchik [209], along with additional labels for "love" and neutral texts. Table 3.6 presents the performance of the system broken down by individual labels. Interestingly, the overall advantage of `ExtremITA` on the aggregated result is largely influenced by a skewed label distribution. The Best Competitor struggles significantly with "fear", which is also the least represented label in the test set. Moreover, there is a clear

Label	ExtremITA			Best Competitor			Δ			Support
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	
Anger	0.759	0.393	0.518	0.500	0.464	0.481	0.259	-0.071	0.037	56
Anticipation	0.675	0.612	0.642	0.690	0.471	0.559	-0.015	0.141	0.083	85
Disgust	0.674	0.588	0.628	0.554	0.594	0.573	0.120	-0.006	0.055	165
Fear	0.636	0.538	0.583	1.000	0.077	0.143	-0.364	0.461	0.440	13
Joy	0.648	0.590	0.618	0.684	0.520	0.591	-0.036	0.070	0.027	100
Love	0.745	0.398	0.519	0.708	0.330	0.450	0.037	0.068	0.069	103
Neutral	0.657	0.757	0.704	0.705	0.614	0.656	-0.048	0.143	0.047	210
Sadness	0.750	0.537	0.626	0.584	0.474	0.523	0.166	0.063	0.103	95
Surprise	0.632	0.422	0.506	0.344	0.539	0.420	0.288	-0.117	0.086	102
Trust	0.698	0.673	0.685	0.679	0.699	0.688	0.019	-0.026	-0.003	272

Table 3.6: Precision, Recall and F1-measure of ExtremITA on the EmIt A task, where the Δ column is the difference between `ExtremITA` and the best competitor of this shared task, that achieved second position.

inverse correlation between the frequency of positive instances in the test set and the gain in performance of `ExtremITA` over the Best Competitor. This suggests that `ExtremITA` excels at classifying sparser phenomena. In addition to its strength in handling less common labels, `ExtremITA` demonstrates superior performance across most emotions. The only exception is "trust", where the Best Competitor achieves better results. This analysis highlights the nuanced advantages of `ExtremITA`, particularly its ability to model and correctly predict underrepresented labels. These insights provide valuable directions for future refinements in multi-label classification tasks, especially for addressing imbalances in label distributions.

Label	0			100			500			1000			All			Sup.
	Pr.	R.	F1	Pr.	R.	F1	Pr.	R.	F1	Pr.	R.	F1	Pr.	R.	F1	
Anger	.077	.018	.029	.111	.018	.031	.488	.357	.412	.500	.268	.349	.759	.393	.518	56
Antic.	.000	.000	.000	.310	.106	.158	.542	.376	.444	.621	.424	.503	.675	.612	.642	85
Disgust	.480	.073	.126	.476	.412	.442	.632	.333	.437	.586	.455	.512	.674	.588	.628	165
Fear	.071	.077	.074	.500	.077	.133	.000	.000	.000	.750	.231	.353	.636	.538	.583	13
Joy	.068	.240	.106	.675	.270	.386	.571	.360	.442	.588	.470	.522	.648	.590	.618	100
Love	.190	.039	.065	.571	.078	.137	.882	.146	.250	.683	.272	.389	.745	.398	.519	103
Neutral	.000	.000	.000	.364	.495	.419	.400	.824	.538	.513	.743	.607	.657	.757	.704	210
Sadness	.244	.305	.271	1.000	.105	.190	.742	.242	.365	.667	.358	.466	.750	.537	.626	95
Surprise	.081	.157	.107	.000	.000	.000	1.000	.039	.075	.382	.333	.356	.632	.422	.506	102
Trust	.339	.618	.438	.429	.735	.542	.556	.570	.563	.648	.562	.602	.698	.673	.685	272

Table 3.7: Precision, Recall and F1-measure of ExtremITA on the EMit A task. The last column (Support) is the number of positive instances per label in the test set.

In Table 3.7, we present a learning curve analysis for the EMit A task with an increasing number of examples in the training set. While the overall performance of `ExtremITA` is competitive, the figures illustrate its strong reliance on fine-tuning. Notably, even with 1000 examples for instruction-tuning, the performance remains significantly lower than that achieved with access to the full EMit training set, across all labels. This highlights the importance of larger datasets for fine-tuning to capture the nuances of multi-label classification effectively.

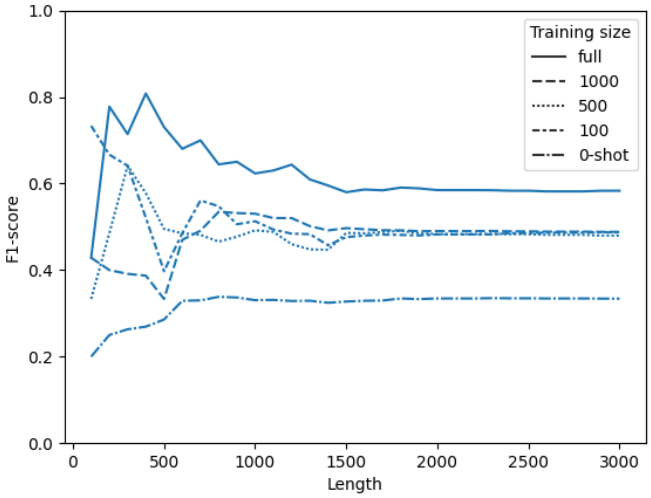


Figure 3.5: F1-measure of ExtremITA on the LangLearn test set, with texts removed that are longer than an increasing threshold (horizontal axis).

In contrast, `ExtremITA` performed poorly on the LangLearn task, a text-pair classification problem. This task primarily relies on stylistic features rather than semantic information to identify the development in language learning of the text author. Given this premise, we anticipated subpar performance from the transformer-based model, which is better suited to capturing semantic relations. A further challenge in this task is the length of the texts. For computational reasons, we truncated texts to 100 tokens or fewer, which resulted in retaining only 24.6% of the tokens from the two training sets combined. This significant reduction in input data is likely a key factor in the lower performance. We hypothesised that longer texts in the test set, which were truncated more heavily, are penalised due to the loss of information. To investigate this, we plotted the accuracy of the different `ExtremITA` models against portions of the test set filtered by text length. In Figure 3.5, the horizontal axis represents a threshold on the minimum size (in characters) of the two texts forming an instance. The downward trend for the "full" model confirms that the predictions are more accurate on shorter pairs of texts, while errors increase as text length grows. The trend is consistent, albeit noisier, across different training sizes, indicating that additional training examples provide a greater performance gain on shorter texts compared to longer ones. This marked trend was unexpected, since the Transformer architecture is well-known for its long-range attention. More experiments in this direction are needed to derive significant conclusions.

The `ExtremITA` approach demonstrates how instruction-tuned Large Language Models can effectively address a wide range of linguistic tasks within a single, monolithic framework. By

training on a diverse set of examples and relying on prompt-based methodologies, the model achieves competitive performance across many tasks, ranking first in several while remaining robust in others. However, the results also highlight limitations, particularly in tasks requiring stylistic analysis or involving lengthy input texts. These challenges underline the importance of adapting large models to specific task characteristics, balancing efficiency and generalization. While `ExtremITA` focuses exclusively on textual input, the natural evolution of these systems involves integrating multimodal capabilities. This is especially relevant for tasks that require reasoning across diverse modalities, such as visual and textual data. The next section explores how to incorporate Multimodal evidence into LLMs, introducing `MM-VQA-it`, and evaluating the simple application on a Visual Question-Answering task, where the model is requested to reason about both images and texts.

3.3 Incorporating Multimodal Evidence into LLMs: MM-VQA-it

Visual Question-Answering (VQA) is a rapidly advancing and highly challenging task in Artificial Intelligence (AI). It requires systems to generate accurate answers to questions posed in natural language, relying on the visual content of an image. These questions often depend on intricate details such as objects, relationships, actions, or other visual attributes. Successfully addressing VQA tasks demands the integration of two complex domains, vision and language, along with reasoning and inference capabilities to produce contextually appropriate responses [14, 244, 116].

For this study, we focused on GQA-it [65], the first large-scale dataset specifically designed for Visual Question-Answering in Italian. Adapted from the original GQA dataset [121], which was created to evaluate real-world visual reasoning in English, GQA-it offers a wealth of visual and linguistic data tailored for Italian. The original GQA dataset comprises over 22 million question-answer pairs associated with images sourced from the Visual Genome dataset, where each image is enriched with detailed scene graphs that capture objects, attributes, and relationships present in the scene. Figure 3.6 illustrates an example image, showcasing the diversity of questions that can be posed, each with corresponding answers in both English and Italian:

- $\mathbf{Q}_{en}$: *Is the remote to the right or to the left of the book?* $\mathbf{A}_{en}$: *Right*
- $\mathbf{Q}_{it}$: *Il telecomando è a destra o a sinistra del libro?* $\mathbf{A}_{it}$: *Destra*
- $\mathbf{Q}_{en}$: *How thick is the book to the left of the remote?* $\mathbf{A}_{en}$: *Thick*

Figure 3.6: Example of an image from the GQA-it dataset, taken from [65] (image `id`: `n90294`)

- $\mathbf{Q}_{it}$: *Quanto è spesso il libro a sinistra del telecomando?* $\mathbf{A}_{it}$: *Spesso*
- $\mathbf{Q}_{en}$: *What device is to the left of the calculator made of plastic?* $\mathbf{A}_{en}$: *Charger*
- $\mathbf{Q}_{it}$: *Quale dispositivo si trova a sinistra della calcolatrice di plastica?* $\mathbf{A}_{it}$: *Caricabatterie*
- $\mathbf{Q}_{en}$: *What's the charger made of?* $\mathbf{A}_{en}$: *Plastic*
- $\mathbf{Q}_{it}$: *Di cosa è fatto il caricabatterie?* $\mathbf{A}_{it}$: *Plastica*
- $\mathbf{Q}_{en}$: *Are there any phones?* $\mathbf{A}_{en}$: *No*
- $\mathbf{Q}_{it}$: *Ci sono dei telefoni?* $\mathbf{A}_{it}$: *No*

These examples demonstrate the complexity and variability of the task. Some questions, such as identifying the colour of an object or its position, can be answered directly through image observation. Others may require prior knowledge, such as understanding the typical materials of a charger. Additionally, ambiguities arise when questions lack clear focus, involve objects not present in the image, or demand follow-up clarifications, often leading to the necessity of dialogical interaction [117, 118, 37], topics further elaborated in the next Chapter. This diversity of challenges underscores the importance of robust multimodal architectures capable of integrating visual and textual data. In the following Section, we will present the method of `MM-VQA-it` [233] for adapting a multimodal model to this complex downstream task, enabling it to generate accurate and context-aware answers to image-based questions.

3.3.1 A Multimodal Approach to Visual Question Answering

Visual Question-Answering (VQA) has emerged as a critical benchmark for evaluating the ability of AI systems to perform tasks requiring deep semantic understanding and cross-modal reasoning between language and vision. Over recent years, substantial advancements in models and resources, particularly for the English language, have driven progress in this domain [121, 284]. These developments have enabled systems to tackle increasingly sophisticated reasoning tasks. Surveys such as [159] offer a comprehensive overview of the diverse methodologies employed in VQA, with a particular emphasis on integrating vision and language, a pivotal area in modern AI research.

Multimodal Large Language Models (MLLMs) have emerged as particularly promising solutions in this space [285]. Unlike traditional language models, these models process inputs from multiple modalities, such as text and images, by incorporating embeddings from both visual and textual data. Visual inputs are typically encoded using specialized encoders, such as Convolutional Neural Networks (CNNs) or Vision Transformers (ViT, [75]). By merging these embeddings, MLLMs enable reasoning over combined linguistic and visual contexts.

State-of-the-art MLLMs like LLaVA [160], CogVLM [274], InternVL2-8B [49], MiniCPM-V [284], GPT-4 [199], and Gemini [17] extend foundational language models such as LLaMA [261, 262] or the GPT series [199] by incorporating visual encoders. These models utilize techniques like cross-modal attention [253] and visual grounding to align visual information within the input space of LLMs, allowing for accurate, context-aware responses in multimodal tasks [106]. A notable exception is GPT-4o [200], designed as an *omni-modal* system capable of processing and generating outputs across all modalities. While promising, this model remains inaccessible at the time of writing due to resource and availability constraints.

These multimodal systems often reuse pre-trained LLMs, which have been fine-tuned within a multimodal architecture [160]. However, their effectiveness is tied closely to the original language model, including its inherent limitations. For example, models like LLaMA-2 [262], pre-trained on trillions of words, exhibit a significant language imbalance: over 90% of the text is in English, with less than 0.1% in Italian. As detailed in Appendix A, such imbalances risk reducing performance in multilingual contexts. Additionally, fine-tuning for visual reasoning is overwhelmingly conducted on English datasets or, in some cases, bilingual English-Chinese datasets, as seen with CogVLM [274, 105]. The scarcity of large-scale multimodal datasets in non-English languages, such as Italian, further hinders the robustness, generalizability, and performance of VQA models in multilingual settings.

3.3.2 Experimental Setup and Evaluation

This section addresses two key research questions: how well state-of-the-art multimodal models perform on VQA tasks in non-English contexts, and whether fine-tuning on Italian-specific data enhances their performance. To investigate this, we applied MiniCPM-V 2.6 to the GQA-it dataset, evaluating its off-the-shelf performance and its results after fine-tuning. The primary objective was to measure the impact of fine-tuning on the target language and analyze common error types in both setups. The evaluation was conducted on the 3,000 manually validated (gold) examples from the GQA-it test set [65]. This high-quality subset ensures reliable results and allows direct comparisons with LXMERT [253], a BERT-based model that serves as a strong baseline in this domain.

Each question in GQA-it is presented in natural language, with the expected answer being a concise expression, typically one to four tokens long. Model performance was assessed using accuracy, calculated as the percentage of questions for which the generated response exactly matched the expected answer. This straightforward metric provides a clear measurement of the effectiveness of the model in generating correct responses. Two distinct experimental setups were designed to evaluate the MiniCPM-V model:

1. **Zero-Shot Setting**: In the first setup, the model was applied in a zero-shot setting, meaning no additional training or fine-tuning was performed on the target dataset. MiniCPM-V 2.6, sourced from Huggingface[12], was used as-is. While the model was primarily trained on data dominated by English, it is expected to exhibit some understanding of other languages, including Italian. This setup evaluates the ability to generalize to a new language and domain without any task-specific adjustments. The following prompt[13] was used:

   `<IMAGE>`
   Rispondi alla seguente domanda con una sola parola o poche parole ma solo se necessario, non aggiungere ulteriori informazioni:
   `<QUESTION>`

   where `<IMAGE>` is a fictitious token and at run time it will be replaced with the embedding of the image and `<QUESTION>` is a placeholder for the input question. This formulation was designed to be clear and direct, ensuring that the model understood it needed to provide a concise, single-word (or few words) response without adding any extra information.

[12] https://huggingface.co/openbmb/MiniCPM-V-2_6

[13] In English: *Answer the following question with one word or a few words, but only if necessary. Do not add further information.*

2. **Fine-Tuned Setting**: In the second setup, MiniCPM-V was fine-tuned on the GQA-it training set to assess the extent to which task-specific data in Italian improves its performance. Fine-tuning involved adapting the model to the unique linguistic and domain-specific features of the dataset, enabling a more tailored evaluation of its capabilities. The training was conducted using the DeepSpeed framework[14] on a cluster of four A100 GPUs, each equipped with 80GB of memory. TStandard fine-tuning parameters[15] were used, with a learning rate of $1 \cdot 10^{-6}$, and the model was trained for a single epoch given the large number of examples in the dataset. Both the vision encoder and the language model components of MiniCPM were fine-tuned. To manage memory usage efficiently, the batch size per device during training was set to 4. The fine-tuning process utilized a streamlined prompt format, where only the original question and the image itself were presented to the model:

   ```
   <IMAGE>
   <QUESTION>
   ```

   This minimalistic prompt design was intentionally chosen to allow the model to focus exclusively on the content of the question and the corresponding visual context. Additional instructions were considered unnecessary for the task, as the repetitive use of the same prompt structure during both training and application naturally guided the learning. By simplifying the prompt, the fine-tuning process emphasized direct question-answering in Italian, enabling the model to adapt to the task without extraneous distractions. However, this approach also introduced a drawback: the model lost its instruction-following capabilities, which are typically integral to many general-purpose language models. After fine-tuning, MiniCPM-V became task-specific, designed solely to respond to image-related questions with concise answers, limiting its versatility in broader multimodal applications.

Results and Discussion. The results of the experiments are presented in Table 3.8. The first row reports the baseline model from [65], which employs a naive approach of predicting the most frequent response in the dataset ("*yes*" or "*sì*"). This method achieves an accuracy of 17.6%, providing a minimal benchmark for comparison.

A more sophisticated baseline, LXMERT-it, was introduced in [65] and specifically trained on the GQA-it dataset. This model achieves an accuracy of 51.0%, representing the current state-of-the-art on this benchmark prior to the current work. In comparison,

[14] https://github.com/microsoft/DeepSpeed
[15] https://github.com/OpenBMB/MiniCPM-V/blob/main/finetune/finetune_ds.sh

Model	Accuracy
Baseline [65]	17.6%
LXMERT-it [65]	51.0%
MiniCPM-V 2.6 (Zero-shot)	33.4%
MiniCPM-V 2.6 (Fine-tuned)	59.4%

Table 3.8: Performance comparison of various models on GQA-it.

when MiniCPM-V 2.6 is applied in a zero-shot setting, it achieves an accuracy of 33.4%, which surpasses the naive baseline but still falls short of the performance of LXMERT-it. A significant portion of errors in this zero-shot scenario stems from the model generating responses in languages other than Italian, as well as its struggles with yes/no questions. This behaviour likely reflects the predominance of English and Chinese data in the pre-training and evaluation of these models, leading to weaker generalization for Italian.

Fine-tuning MiniCPM-V 2.6 on the GQA-it dataset produces a substantial improvement, achieving an accuracy of 59.4%. This surpasses both the LXMERT-it model and the zero-shot MiniCPM-V performance. These results highlight the importance of fine-tuning on target language datasets, enabling the model to adapt to the specific linguistic nuances and complexities of Italian. Moreover, fine-tuning helps the model address the unique visual reasoning challenges posed by the dataset, resulting in more accurate and contextually relevant responses.

It is also important to consider the architectural differences between the models. LXMERT-it, despite being specifically designed for this dataset, has at least one order of magnitude fewer parameters compared to MiniCPM-V 2.6. However, MiniCPM-V benefits from the extensive pre-training of its Qwen2-based LLM component [283], which was trained on significantly larger and more diverse datasets. This broader training base enables it to better handle the complexities of multimodal reasoning and cross-lingual understanding, particularly when fine-tuned. Fine-tuning both the language and vision components of MiniCPM-V contributed significantly to its performance. Future work could explore selective fine-tuning strategies or parameter-efficient techniques like LoRA [119] to further optimize the process. These methods may provide insights into balancing computational efficiency with fine-tuning efficacy. Notably, this study represents the first instance where a multimodal model of this scale has been fully fine-tuned for the GQA-it task, marking a significant step forward in adapting multimodal architectures for language-specific applications.

Error Analysis. The error analysis aimed to identify the most common types of misclassifications made by the system. The percentages are summarized in Table 3.9 and include a comparison with the analysis from [65], which examined errors on a random 10% sample

Error Type	**Example(s)**	Our	[65]
Object	*tavola* (‘*table*’) vs *sedia* (‘*chair*’)	39%	30%
Syn or hyp	*persona* (‘person’) vs *donna* (‘*woman*’)	16%	17%
Attributes	blu (‘*blue*’) vs *nero* (‘*black*’); *chiuso* (‘*closed*’) vs *aperto* (‘*open*’)	17%	14%
Morph. feat.	*bella* (‘*beautiful*’) vs *bello* (‘*beautiful*’); *persona* (‘*person*’) vs *persone* (‘*people*’)	5%	3%
Actions	*sta dormendo* (‘*sleeping*’) vs *sta sdraiato* (‘*is lying down*’)	3%	3%
Spatial feat.	*destra* (‘*right*’) vs *sinistra* (‘*left*’)	2%	2%
Binary	*sì* (‘*yes*’) vs *no* (‘*no*’)	18%	31%

Table 3.9: Distribution of errors of MiniCPM-V 2.6 (*Our* fine-tuned) and LXMERT-it on GQA-it gold test set into the predefined classes from [65].

of the validated test set. In contrast, the current analysis was manually conducted on the entire test set, providing a more comprehensive overview of performance. While the results are not directly comparable due to differences in sampling, the error types and their distribution offer valuable insights into the differing behaviours of the LXMERT-it model and the fine-tuned version of MiniCPM-V. These insights serve as a foundation for guiding future improvements to both models.

The most striking difference lies in binary-type questions, particularly those involving yes/no answers. These seemingly straightforward questions often depend on subtle contextual cues, making them challenging for models to handle. For example, as shown in Figure 3.7, when asked, “*C’è del vino in questa foto?*”[16], the model incorrectly responds with “*sì*” (“*yes*”), despite the image showing only empty wine glasses, not actual wine. MiniCPM-V reduces errors in this category to 18%, compared to 31% for LXMERT-it, likely due to better collaboration between its vision and language components, which aids in more nuanced reasoning.

Object-related errors are another prominent category, accounting for 39% of the errors, compared to 30% for LXMERT-it. These errors often involve confusion between visually or semantically similar objects. For instance, in Figure 3.8, the model misinterprets the question “*Quale tipo di mobile è nero?*”[17], predicting “*tavola*” (“*table*”) instead of “*sedia*” (“*chair*”). This suggests that improvements in object recognition and feature differentiation are necessary to reduce such errors.

Attribute-related errors, constituting 17% of the misclassifications, highlight the challenges in fine-grained feature extraction. Common mistakes include confusing similar attributes, such as “*blue*” and “*black*,” indicating limitations in precise color or texture recog-

[16]In English: “*Is there any wine in this picture?*”
[17]In English: “*What type of furniture is black?*”

Figure 3.7: Example from the GQA-it gold test set (`image id n28572`) where the fine-tuned MiniCPM-V 2.6 model predicts "*si*" instead of "*no*" for the question "*C'è del vino in questa foto?*".

Figure 3.8: Examples from the GQA-it gold test set (`image id n283587`) where fine-tuned MiniCPM-V 2.6 model predict "tavola" instead of "sedia" when answering the question "*Quale tipo di mobile è nero?*".

nition. Semantic errors involving synonyms (syn) or hypernyms (hyp) represent 16% of the errors. Examples include confusing "*donna*" and "*ragazza*" ("*woman*" vs. "*girl*") or "*uomo*" and "*persona*" ("*man*" vs. "*person*"), reflecting the difficulty with linguistic nuances and hierarchical semantic relations. This limitation suggests a need for better linguistic grounding

to distinguish related but distinct concepts.

Ambiguity in spatial relationships also contributes to errors. For instance, in Figure 3.9, the model answers "*scrivania*" ("*desk*") instead of the annotated "*tappeto*" ("*carpet*") for the question "*Cosa c'è davanti alla felpa?*"[18]. This discrepancy may stem from differing interpretations of "*in front*", when the annotator's perspective versus the spatial reasoning of the model. The ability to ask clarifying questions, such as "*What do you mean by 'in front'?*", could help resolve such ambiguities.

Figure 3.9: Examples of ambiguity from the GQA-it gold test set (`image id n398257`) where fine-tuned MiniCPM-V 2.6 model answered "*scrivania*" instead of "*tappeto*" to the question "*Cosa c'è davanti alla felpa?*"

Similarly, attentional focus differences can result in errors. As shown in Figure 3.10, the model answers "*scrivania*" ("*desk*") instead of the annotator's "*tappetino*" ("*mouse pad*") for the question "*Su cosa è sdraiato il gatto?*"[19]. These errors highlight divergent focal points: the annotator focuses on the specific mat under the cat's head, whereas the model considers the larger desk. This suggests the need for enhanced attentional mechanisms to align the focus of the model with human annotators' expectations. These findings provide valuable directions for improving multimodal models. Enhancing object recognition, refining linguistic grounding, and addressing ambiguities in spatial reasoning and attentional focus could significantly improve model performance in VQA tasks.

[18]In English: "*What is in front of the sweatshirt?*"
[19]In English: "*What is the cat lying on?*"

Figure 3.10: Examples of attention ambiguity from the GQA-it gold test set (`image id n433692`) where fine-tuned MiniCPM-V 2.6 model responded "*tappetino*" instead of "*scrivania*" to the question "*Su cosa è sdraiato il gatto?*"

The exploration of multimodal approaches to Visual Question Answering through MiniCPM-V on the GQA-it dataset has highlighted the transformative potential of integrating visual and linguistic reasoning within AI systems. While fine-tuning significantly enhanced the performance of the model, especially in adapting to Italian-specific datasets, it also underscored critical limitations, such as difficulties in resolving spatial ambiguities and object-related misclassifications. These challenges reflect broader issues in aligning vision and language processing, particularly in multilingual contexts where pre-training biases may persist. Despite these limitations, the ability of multimodal systems to bridge the gap between visual understanding and textual reasoning is a step forward in developing AI capable of complex, context-aware interactions. This progress connects directly to the broader challenge of interpreting and interacting in natural language, particularly in Human-Robot Interaction (HRI). In the next chapter, we will go deeper into how advancements in Natural Language Understanding (NLU) are essential for enabling robots to comprehend, interpret and respond effectively to natural language inputs. Moreover, a specific focus on collaborative environments will addressing issues of ambiguity and intent interpretation, as observed in multimodal tasks like VQA.

Chapter 4

Natural Language Understanding in HRI

In the field of Human-Robot Interaction (HRI), one of the most significant challenges is enabling robots to understand and respond to natural language effectively. For humans, comprehending a sentence involves interpreting its meaning within the context of experiences shaped by social and cognitive processes. For robots, however, language understanding is a far more intricate, multi-layered task that extends beyond parsing words or executing predefined commands [18]. While the previous chapter focused on systems addressing isolated linguistic or visual tasks, this chapter shifts the focus to architectures that interpret language comprehensively, decipher its implications, and decide how to act accordingly.

To achieve genuine understanding, robots must go beyond syntactic analysis [112, 114, 115] and grasp the intent behind utterances. This intent is shaped by the environment, the speaker's goals, and the task at hand [164]. Such understanding becomes particularly challenging in dynamic, collaborative settings where instructions may be ambiguous or heavily reliant on shared context. For instance, an instruction like, "*Could you hand me that?*" requires the robot to infer what "*that*" refers to, considering the objects present, the surroundings, and the current activity. Thus, natural language understanding in robots necessitates grounding language in the physical world [269, 242, 287]. Typically, this grounding involves processing images captured through the cameras [102, 222, 156], which provide vital context about the environment, complementing spoken or written instructions. Integrating these modalities, visual perception and language, is a hallmark of Multimodal Systems [35, 12, 25].

Effective human-robot communication depends on several critical factors. First, there must be a shared framework for interpreting language, where words and phrases are mapped to real-world entities and actions, commonly referred to as Grounded Language Understanding [189]. Second, robots must align their goals with those of their human collaborators, fostering seamless cooperation [264, 272]. Third, feedback mechanisms, such as clarification and correction, are essential for maintaining robust communication [245, 164]. These mech-

anisms enable both humans and robots to refine their mutual understanding continuously. For example, if a robot cannot resolve the referent of "*that*", it might ask, "*What do you mean by 'that'?*". Alternatively, when presented with multiple potential referents, the robot might hypothesize and seek confirmation: "*Do you mean the book?*". Such adaptive feedback loops are crucial for navigating the inherent ambiguity of human language.

This chapter will go deeper into two critical aspects of natural language understanding in HRI. The first section, Interpreting Natural Language, explores how robots connect linguistic input to their perception and understanding of the physical world, emphasizing grounded language understanding. The second section, Interacting in a Collaborative Environment, examines how robots use this understanding to cooperate with humans effectively. This section highlights how robots can actively participate in conversations, pose relevant questions, and make informed decisions to support collaborative tasks.

4.1 Interpreting Natural Language

Interpreting a sentence involves far more than identifying keywords: it requires an understanding of the relationships between words, their syntactic roles, and their contribution to the overall meaning of the sentence [60, 171, 84, 290, 265, 34, 271]. For robots, this entails transforming a sequence of words into actionable or perceptible meanings, firmly grounded in their operational environment.

How can a robot truly interpret a sentence? In Natural Language Processing (NLP), sentence interpretation involves mapping linguistic input (phrases, sentences, or entire dialogues) onto a structured representation that a machine can use to take actions or make decisions. This process requires robots to understand not only the literal meaning of an utterance but also its contextual implications, discerning what each word or phrase means in relation to the task, environment, and interaction at hand. This interpretation typically unfolds across several layers:

- **Syntactic Analysis**: Identifying the sentence structure (e.g., subjects, verbs, objects) to extract relevant information and understand how words relate to one another.
- **Semantic Analysis**: Deriving meaning from the syntactic structure, such as determining the primary action and the entities involved.
- **Pragmatic Understanding**: Interpreting the intent behind the utterance, understanding the domain of discussion, and situating the utterance within the goals of the interaction.

At its core, language interpretation involves mapping linguistic expressions onto executable actions and real-world entities, informed by the prior knowledge and shared context of the robot. A robust framework for interpreting natural language must satisfy several criteria: it should comprehensively capture linguistic nuances and intentions, operate independently of specific domains or languages, and be adaptable for use by any robotic entity to facilitate human-robot interactions. In addition to computational approaches, psycholinguistics [263, 87, 246] offers valuable insights by combining methods and theories from psychology and linguistics to evaluate the psychological underpinnings of linguistic rules and processes. Psycholinguistics also bridges the gap between word and sentence processing and the deeper processes of constructing and interpreting messages. Resources such as FrameNet, based on Frame Semantics theory, along with VerbNet and PropBank, play a crucial role in this domain. These frameworks provide structured representations of meaning that help robots interpret language more effectively, linking linguistic input to actionable understanding.

FrameNet[23] is a comprehensive lexical database that organises semantic frames [83], which are conceptual structures describing specific types of events or situations. It structures predicates into frames, where arguments are mapped to frame elements. Each frame consists of a verb (or related action) and the associated roles required to complete its meaning. For instance, the KILLING frame, evoked by words like assassinate, behead, or kill, is notable for its extensive set of evoking words, making it among the most polysemous.

FrameNet serves as a crucial resource for systems aiming to generalise language understanding across various contexts. Each frame defines its frame elements, specifying the roles of participants or attributes within the event or situation. For example, the BRINGING and TAKING frames include elements such as:

- AGENT: The entity performing the action.
- THEME: The object being moved.
- SOURCE: The starting location.
- GOAL: The destination.

The difference between these frames depends on the context of the utterance. Consider the sentence: "*Take the book on the table*":

- If *the book* is already *on the table*, the frame is TAKING, and the THEME is "*the book on the table*". Here, the required action is to simply pick up the book.

- If *the book* is elsewhere, the frame is BRINGING, the THEME is "*the book*", and the GOAL is "*on the table*". In this case, the instruction implies moving the book to the table.

This example illustrates the importance of integrating contextual knowledge to resolve ambiguities and correctly interpret frame structures.

VerbNet[235] is the largest online verb lexicon for English, offering a hierarchical, domain-independent classification of verbs based on shared syntactic and semantic properties. Building on Levin's original verb classification, VerbNet refines this framework by organising verbs into classes characterised by thematic roles, selectional restrictions, and syntactic frames linked to semantic predicates. A notable feature of VerbNet is its mappings to other lexical resources, such as WordNet, PropBank, and FrameNet, enhancing its applicability across NLP tasks. For instance, the verb *take* is associated with the `steal-10.5` class, which includes verbs where an agent removes a theme from a source. Thematic roles for this class include:

- AGENT: the doer of the action
- THEME: the object being taken
- SOURCE: the origin of the object

In the sentence "*Take the book from the table*", VerbNet would parse the roles as:

- AGENT: (implicit, e.g., "you")
- THEME: "the book"
- SOURCE: "the table"

This classification supports a wide range of NLP tasks, including semantic role labelling, machine translation, and information extraction, by providing a clear understanding of verb behaviour.

PropBank[204] is another widely used resource, offering verb annotations that link predicates to their arguments using a simplified role set. While less conceptually detailed than FrameNet, PropBank excels in providing fast and straightforward mappings between verbs and their arguments, making it a valuable tool for training systems requiring efficiency and scalability. PropBank annotations are often integrated with syntactic parsers to identify the core arguments of verbs within sentences. It defines core roles (e.g., ARG0 for Agent, ARG1 for Patient) specific to each verb sense. For example, the verb take might have the following role definitions:

- If *the book* is already *on the table*: Verb sense is `take.01` (to physically pick up an object), ARG0 (Agent) is the person performing the action (e.g., "*you*" in an implicit context) and ARG1 (Patient) equals to "*the book on the table*".
- If *the book* is far away: Verb sense is `take.01` (the same sense), ARG0 (Agent), the person performing the action (e.g., "*you*"), ARG1 (Patient) is "*the book*" and ARG2 (Goal) equals to "*on the table*".

In FrameNet, examples are framed in terms of abstract conceptual models like TAKING or BRINGING, with richly annotated frame elements such as THEME and GOAL, which adapt to the context of the utterance. In contrast, PropBank approaches examples with a narrower, verb-specific focus (e.g., `take.01`), using numbered arguments like ARG0 (Agent), ARG1 (Patient), and ARG2 (Goal). These roles are consistent across different verbs but lack the expressiveness to capture nuanced conceptual distinctions. Both FrameNet and PropBank have been instrumental in advancing natural language understanding by providing standards that enable models to assign roles across various contexts. VerbNet bridges the gap between these resources by combining the broad coverage of verbs with thematic roles and syntactic patterns. It offers a hierarchical, unified framework that links directly to both FrameNet and PropBank. While FrameNet excels in conceptual depth and flexibility, PropBank prioritises simplicity and efficiency, and VerbNet provides a balanced integration of syntax and semantics. Together, these resources form a complementary trio, each with unique strengths that support semantic analysis. For this thesis, FrameNet has been chosen as the foundation for the next work discussed, due to its rich conceptual structures and detailed semantic frames, which offer the depth and flexibility required to interpret language in diverse and complex contexts. One of the key methods for interpreting language, leveraging such psycholinguistic theories, is Semantic Role Labeling.

4.1.1 Semantic Role Labeling

Semantic Role Labeling (SRL) is a fundamental task in Natural Language Processing (NLP) that focuses on identifying the roles that words play in a sentence, particularly in relation to a main action or verb [82]. SRL is instrumental in interpreting utterances by providing a structured, unambiguous representation of their meaning. Its goal is to answer questions such as "Who did what to whom?" by labelling each word or phrase with its corresponding semantic role. This process typically involves identifying predicates (often verbs) that denote actions or events and their associated arguments, which define roles such as the agent, patient, or instrument. SRL acts as a bridge between syntactic structures and their semantic meanings, offering machines a deeper understanding of natural language.

For instance, consider the sentence: "*Bring me the book on the table*".

- Who? → Implicitly *you* (the AGENT of the action)
- What? → *the book* (the THEME being acted upon)
- To whom? → *me*, the speaker (the RECIPIENT of the action)

SRL has undergone significant evolution, transitioning from traditional rule-based systems to modern machine learning and deep learning approaches. Early methods relied on rule-based techniques and annotated corpora to manually define predicate-argument structures. Machine learning models, such as Conditional Random Fields [143, 57] and Structured Support Vector Machines [62, 213], later automated this process, leveraging statistical patterns to improve performance. In recent years, transformer-based models, such as BERT [70] and GPT [216], have revolutionised SRL by capturing long-range dependencies and contextual nuances with remarkable accuracy. These models excel at understanding complex sentence structures and disambiguating semantic roles. Despite these advancements, SRL still faces several challenges:

- **Ambiguity**: Resolving syntactic and semantic ambiguities, such as prepositional phrase attachment, remains a major hurdle. For example, the phrase "*on the table*" in the original sentence "*Take the book on the table*" can either modify the noun "*book*" (indicating its location) or the verb "*take*" (specifying the goal of the action).
- **Domain Adaptation**: SRL models often struggle to generalize across domains without extensive retraining. Usually, when changing the operational domain, a change of the specific knowledge is needed.
- **Resource Scarcity**: Developing SRL systems for under-resourced languages is hindered by the lack of annotated datasets, as resources like FrameNet, VerbNet or PropBank are primarily available for English.

The primary objective of Semantic Role Labeling (SRL) is to enable machines to interpret sentences in a manner that aligns with human understanding. By identifying semantic roles, robots can determine who is performing an action, what the action entails, and how various entities are interconnected within a task or environment. This capability is crucial for robots to follow instructions, perform tasks accurately, and engage in collaborative activities with humans. In Human-Robot Interaction (HRI), SRL empowers robots to process language beyond surface-level commands. For example, when a robot receives the instruction, "*Take*

the book on the table", SRL allows it to identify that "*book*" is the object to be grasped (the THEME) and "*table*" is the source location (the SOURCE) from which the book should be taken. This level of understanding is essential for the robot to execute the task correctly and efficiently. By establishing these roles, SRL enables a deeper comprehension of linguistic constructs, supporting advanced NLP tasks such as text summarisation, question answering, and machine translation. Resources like FrameNet, VerbNet, and PropBank, as discussed earlier, provide theoretical and practical frameworks for implementing SRL. Most SRL systems rely on one or more of these resources, leveraging their underlying psycholinguistic theories and associated dependencies to interpret language meaningfully.

For SRL to be effective in real-world applications, robots must integrate semantic understanding with their perception of the environment. While linguistic resources offer standardised frameworks for recognising common actions and their associated roles, they must be applied dynamically to the sensory inputs of the robots, such as visual or spatial data, to truly ground language understanding in context. This combination of linguistic knowledge and environmental awareness is critical for enabling robots to respond naturally to human commands. Whether following a direct instruction or resolving ambiguities in an utterance, robots need to link language processing with real-world perceptions. For instance, understanding the phrase "*Take the book on the table*" requires not only identifying semantic roles but also perceiving and locating the book and table within the physical environment. Grounding SRL in sensory and contextual data bridges the gap between abstract linguistic frameworks and practical task execution. This integration allows robots to interpret and act upon language in a natural, human-like manner, making them more capable collaborators in diverse scenarios.

4.1.2 The SRL Cascade

A widely adopted approach for addressing the Semantic Role Labeling (SRL) task is to decompose it into a sequence of classification tasks, organised in a processing cascade. This methodology mirrors the layered structure of SRL, with each step building upon the outputs of previous ones. However, a known limitation is that errors in earlier stages can propagate downstream. To mitigate this, each stage incorporates the outputs of prior steps as additional context, effectively narrowing the decision space for subsequent tasks:

- **Frame Detection**: The first step is to identify the frames evoked by predicates in the input sentence. Each word or phrase is classified as either triggering a specific frame (e.g., BRINGING, TAKING) or being irrelevant for frame detection. This step relies on linguistic features such as word embeddings, lexical units, and syntactic information.

Additional features, such as the capabilities of the robot or domain-specific frames, may also be included to enhance accuracy.

- **Argument Identification**: After detecting the frames, the next step is to identify the arguments associated with each frame. This involves determining the boundaries of argument spans (e.g., "*the book*" or "*on the table*"). Typically, sequential tagging schemes like IOB2 are used, where tokens are labelled as the Beginning (B), Inside (I), or Outside (O) of an argument span. This stage plays a crucial role in establishing which parts of the sentence correspond to frame elements.
- **Argument Classification**: In the final step, identified argument spans are classified into semantic roles (e.g., Theme, Goal, Source). Features such as the syntactic head of the argument, domain-specific properties (e.g., whether an object is containable), and distributional semantics are used to generalise over lexical variations and ensure precise classification.

Traditionally, each of these tasks is modelled using supervised learning techniques, such as Conditional Random Fields (CRFs) or structured Support Vector Machines (SVMs) [252, 71, 63]. Recently, end-to-end neural models have shown promise in performing SRL holistically. However, the cascade framework remains advantageous for its modularity and the ability to integrate diverse sources of knowledge.

SRL with the LU4R Framework. One prominent implementation of the cascade approach is the LU4R framework [266], which extends traditional SRL by incorporating a linguistic grounding into the interpretation process. Grounded SRL connects linguistic representations to physical entities in the environment of the robot, enabling the interpretation of commands in a situated context. LU4R achieves this through the use of a Semantic Map, a structured representation of the environment that links the names of the entities and their attributes to lexical, but still linguistic, symbols in the command. For example, in LU4R, the grounding process resolves the prepositional phrase "*on the table*" in the instruction "*Take the book on the table*" by leveraging spatial relationships encoded in the Semantic Map. If the map indicates that the book is located on the table, the system interprets the frame as Taking. Conversely, if the book is elsewhere, the frame is interpreted as Bringing. This integration of contextual knowledge reduces syntactic and semantic ambiguities, ensuring that the correct frame is assigned based on the understanding of the physical world. While LU4R effectively integrates linguistic and spatial knowledge, it still operates predominantly at the linguistic level. The resulting frames and frame elements remain anchored to linguistic expressions, without directly linking the words to corresponding physical objects in the environment [290, 156, 102]. For instance, the word "*book*" does not directly connect to a unique,

identifiable entity in the Semantic Map. To address this limitation, an enhanced grounding process could introduce unique identifiers for physical entities, as provided by the Semantic Map. In such a system, instead of interpreting the THEME as "*the book*", the system could identify it as `m1`, where `m1` is the identifier for the specific book. This approach would force the interpretation cascade to link linguistic expressions to real-world objects, bridging the gap between abstract language processing and physical task execution.

The cascade structure for Semantic Role Labeling (SRL) involves training specialised models for each subtask, allowing for precise optimisation tailored to the unique requirements of tasks such as frame recognition or role assignment [88, 190]. This approach reflects the layered nature of SRL, where each subtask addresses a distinct linguistic challenge. For example, Frame Detection focuses on recognising semantic frames evoked by predicates, leveraging lexical and syntactic features. In contrast, Argument Classification depends on contextual and domain-specific features to assign roles appropriately. The specialisation inherent in this structure ensures that the nuances of each subtask are adequately captured. A critical implication of this architecture is its sequential dependency: the output of one classifier serves as additional input for the next, enriching the decision space of downstream models [190, 266]. For instance: The Frame Detection model identifies the frame (e.g., BRINGING, TAKING) and provides expectations about the argument structure to the Argument Identification model. The Argument Classification model relies on the boundaries identified by the previous stage to assign semantic roles only to the selected spans. This cascading flow enables intermediate knowledge to be incorporated at each step, refining the accuracy and contextual appropriateness of the final output. However, it also introduces the potential for error propagation: mistakes in upstream models can compound errors in later stages. For instance, a Frame Detection error, such as predicting KILLING instead of BRINGING, may result in incorrect argument spans and misclassified roles downstream [44, 260]. Consequently, each stage must produce robust and reliable outputs to minimise the impact on the overall system performance.

At inference time, the cascade operates as a unified pipeline, where the outputs of one model are fed into the next. While this design ensures modularity and interpretability, it also creates a strong interdependence between stages. Each classifier must excel both in isolation and in complementing downstream tasks. The integration of intermediate predictions as contextual inputs introduces additional complexity, as the system must balance local optimisation for individual classifiers with the global objective of accurate SRL [260, 214, 95].

The cascade framework, while modular, poses significant design and implementation challenges:

- **Specialised Models and Annotated Data**: Each subtask requires a dedicated

classifier trained on task-specific annotations, as Frame Detection needs annotations for predicates and their frames, Argument Identification requires span-level annotations for all arguments in a sentence, and Argument Classification demands role-specific labels for the identified arguments. All annotations must be aligned across tasks to ensure consistency. The resource-intensive nature of this annotation process [213, 130] requires significant expertise to maintain accuracy and coherence across these interdependent layers.

- **Error Propagation**: The reliance on intermediate predictions increases the risk of cumulative errors. A robust design is essential to mitigate cascading inaccuracies, ensuring that errors in one stage do not disproportionately affect subsequent stages.

- **Feature Engineering**: Traditional systems, such as LU4R [266], often rely on Support Vector Machines (SVMs), which demand extensive feature engineering. Features such as lexical patterns [266, 190], dependency relations [190, 214, 95], and semantic attributes [88, 213] must be manually crafted, requiring domain expertise and iterative refinement. This process is time-consuming [101, 73] and error-prone [97], with limited generalisability to unseen data or new domains [94].

- **Integration with Grounded Features**: In grounded SRL systems like LU4R, the cascade must integrate linguistic features with perceptual knowledge from a Semantic Map. These grounded features help resolve ambiguities, such as prepositional phrase attachment, by incorporating spatial and environmental information. However, this coupling adds layers of complexity to feature design and integration.

Despite its challenges, the cascade structure offers distinct advantages. Each stage can be independently optimised and evaluated, allowing researchers to isolate bottlenecks and target specific areas for improvement [190, 260, 214]. For example, errors in Frame Detection can be addressed without conflating them with issues in Argument Identification or Classification. The modular approach enables the use of different machine learning models or feature sets for each subtask, allowing for task-specific optimisation [94, 97]. In systems like LU4R, the cascade facilitates the integration of linguistic and perceptual features. By incorporating environmental knowledge from the Semantic Map, the system can reduce ambiguities and enhance contextual understanding. For instance, the phrase on the table is disambiguated by spatial relationships in the Semantic Map, ensuring accurate role assignment and frame interpretation.

Alternatively, Semantic Role Labeling (SRL) can be addressed using a monolithic architecture, where a single model handles the entire task in an end-to-end manner. Recent

advancements in deep learning have enabled such approaches [33, 111, 113], with models like Large Language Models (LLMs) capable of performing predicate identification, argument recognition, and role labelling simultaneously within a unified framework. This monolithic approach is significant because it allows the model to internalise the relationships between subtasks, making decisions holistically based on the input and context. Rather than relying on sequential predictions, where each stage constrains the subsequent one, the monolithic model considers all subtasks concurrently. This integration ensures that the final output reflects the best possible decision across all tasks, conditioned on the full context. Modern deep learning architectures, such as transformers, are particularly well-suited for this approach. Transformers excel at handling the complexities of natural language interpretation while integrating grounded sensory input. By processing linguistic information and real-world context simultaneously, these models enhance the accuracy and robustness of grounded language understanding.

In the next section, a methodology for applying transformers to grounded language understanding will be explored. This approach enables robots to interpret and act on natural language instructions by leveraging both textual and environmental data, coupled with structured representations from Frame Semantics. Such integration represents a significant advancement in creating robots capable of understanding and collaborating in complex, real-world environments.

4.1.3 Grounded Language Understanding via Transformers

Grounded language understanding involves mapping linguistic inputs to physical referents within the environment. In this context, Semantic Role Labeling serves as the theoretical foundation for interpreting the roles of entities within a command, enabling the robot to identify the agent, object, and location involved. When integrated with Transformers, this process becomes more effective due to the sophisticated context-aware architecture of these models. Transformers, such as BART or GPT, are particularly adept at capturing dependencies across long text sequences, making them suitable for interpreting complex linguistic instructions in real time. Capturing long-range dependencies is crucial in Human-Robot Interaction (HRI) because it allows the model to understand contextual relationships that span multiple parts of a command, ensuring accurate interpretation even when details are distributed across different segments of the input. Moreover, Transformers can maintain coherence over extended dialogue turns, enabling them to interpret commands embedded within complex interactions.

Consider the command "*Bring the book on the table*". In this example, SRL helps the

model break down the command by identifying the action (BRINGING), the object (book), and the goal location (on the table). Suppose the book is present in the room but located far from the table. SRL assigns these roles and fills the arguments based on the current configuration of the environment, enabling the model to infer that the action involves changing the state of the world by bringing the book to the table. If the command were instead "*Bring the book from the table*", the SRL structure would adapt, replacing the GOAL argument with SOURCE, indicating a change in the required action. This flexibility allows the robot to adjust its behaviour dynamically to subtle variations in commands.

This argument-filling process relies on the semantic map, which is reflected in a Knowledge Base (KB) storing attributes such as the positions of objects, their properties, and unique identifiers. These identifiers bridge the gap between the physical world and its linguistic representation. For instance, in the command "*Bring the book on the table*", the robot identifies the book (identifier `b1`) and the table (identifier `t1`) from the KB, enabling it to execute the instruction accurately.

In a home automation context, the environment encompasses the physical space in which the robot operates, including various objects, their attributes, and spatial relationships. The KB maintains this information, mapping each object to a unique identifier and storing attributes like position (x, y, z), size, and colour. This detailed representation allows the robot to interact effectively with its surroundings, as illustrated in Figure 4.1. Even when the command remains unchanged, SRL interpretations may vary based on environmental configurations. For example, if the book is not on the table, the command "*Bring the book on the table*" would result in the action being interpreted as BRINGING. Conversely, if the book is already on the table, the same command might be interpreted as TAKING, reflecting a different interaction based on the current state. This adaptability ensures that the understanding is always aligned with the environment, facilitating accurate and context-aware execution of tasks.

In this light, the process of grounding involves linking linguistic interpretation to corresponding physical entities, such as the book and the table. For example, when a command references "*the book*", the robot uses its Knowledge Base (KB) to identify the specific object being referred to and associates it with its unique identifier. Grounding enables the robot to translate abstract linguistic expressions into executable formulas tied to the physical world. Moreover, incorporating spatial relationships into the KB enhances the ability of the robot to interpret commands involving relative positioning. This critical feature, absent in the LU4R [266] architecture discussed earlier, allows the robot to reason about the world by focusing on real physical objects and their attributes. For instance, a command like "*Bring the book near the table*" requires the robot to not only recognise the book and table but also

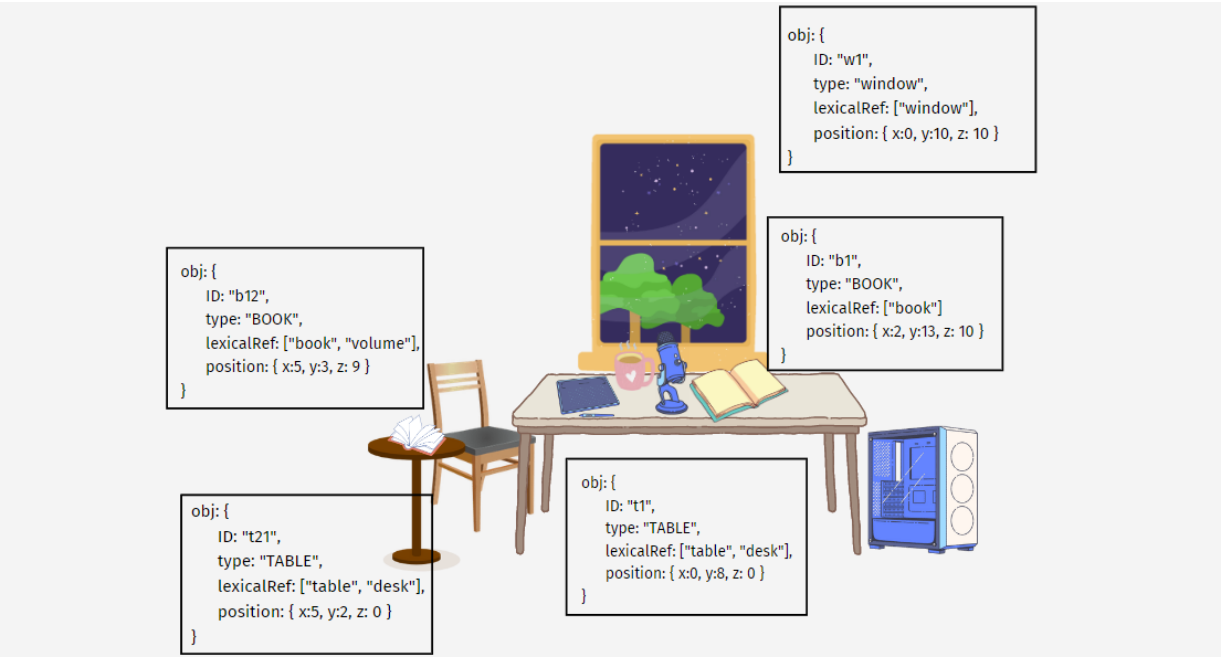


Figure 4.1: An example of the semantic map describing the situation faced by a robot: individual identifiers (`ID`) and types (`type`) are defined for the different objects, such as tables and books.

understand their spatial relationship. By maintaining detailed spatial information, the robot can accurately interpret such instructions and ensure that the resulting SRL representation aligns with the real-world configuration. It must grasp the meaning of "near" and determine how to alter the environment to fulfil the user's intent.

GrUT (Grounded Language Understanding via Transformers) [108] is an architecture designed to manage the end-to-end process of interpreting natural language commands. It leverages BART, a Transformer-based model, to directly process commands in relation to the environment. By integrating linguistic input with environmental data from the KB, GrUT enables robots to understand commands involving nearby objects and execute the appropriate actions. BART operates as a sequence-to-sequence model, taking an input sequence and generating an output sequence. To handle SRL, the task is reformulated as a sequence rewriting problem. GrUT employs a formalism inspired by FrameNet to produce semi-structured text that combines Frames and their Arguments with the relevant parts of the input text. This approach allows semantic roles to be assigned dynamically. Since roles depend on Arguments and Arguments depend on Frames, the output adopts a hierarchical parenthetic structure. During interpretation, the model first identifies the Frame and places its Arguments in parentheses. If multiple Frames are present, they are separated by dashes. For example, given the command "*Bring the book on the table*", the formalism appears as:

TAKING(THEME, GOAL)

Here, TAKING is the `Root` Frame, while THEME and GOAL are its direct descendants. To populate the Arguments, the model rewrites relevant spans of the input text within the parentheses, producing:

TAKING(THEME(*book*), GOAL(*table*))

This representation is valid when the book and table are spatially distant. However, if the book is already on the table, GrUT adjusts its interpretation to reflect the context. For the same command, the output becomes:

TAKING(THEME(*book*), SOURCE(*table*))

In this case, the model correctly identifies that the book is already on the table and that the action involves taking the book, potentially to move it elsewhere. Notice that these two scenarios reflect the two possible interpretations of the sentence, depending on the syntactic attachment of the *PP* (Prepositional Phrase) in the structure:

$$[VP\ Bring\ [NP\ the\ book\ [PP\ on\ the\ table\ ]]] \tag{4.1}$$

$$[VP\ Bring\ [NP\ the\ book]\ [PP\ on\ the\ table\ ]] \tag{4.2}$$

Grounding a command thus involves sharing the definitions and types of surrounding entities (e.g., "*book*" is a kind of instrument filled with words that people typically read), their spatial locations (e.g., the proximity of the "*book*" to the "*table*"), potential synonymous referring expressions (e.g., "*book*", "*volume*", "*tome*"), and their properties. These properties include attributes such as containability (e.g., false for a "*book*" but true for a "*cup*") or the ability to sustain objects placed on top (e.g., true for a "*table*", conditionally true for a "*book*", and always false for "*water*").

As described in [266], GrUT leverages a purpose-built world map describing the surrounding elements, represented in a Knowledge Base (KB) that should be derivable from the physical perception of the robot. Surrounding elements can be added by a knowledge engineer based on ontological assumptions (e.g., entities, classes, properties, and domains) or acquired interactively through dialogue with the user, who might explain, name, and demonstrate the objects. This knowledge can be maintained as Timed Knowledge Graphs, as proposed in [26], to model the arrangement of objects in the environment. The assumption is that objects typically remain stationary unless moved by an agent (human or robot), and

the Knowledge Map should only be updated when explicitly informed by the interlocutor or when the agent performs actions that alter the environment.

In both situations, the body of knowledge related to the environment is taken as given and used during interpretation to meaningfully disambiguate competing interpretations and produce accurate *grounded logic forms* (glf). In GrUT, the process of generating a grounded logic form glf, which expresses the interpretation of a command c, is viewed as a rewriting operation $\mathcal{M}(c) = glf$. Here, the command c is expressed in natural language, and the rewriting corresponds to a translation task. However, since spatial information from the map is essential, the input to $\mathcal{M}$ becomes the pair (sd, c) where sd is the spatial description of the map, and c is the user command. Thus, $\mathcal{M}(sd, c) = glf$ where glf represents the grounded logic form for c. To facilitate this process, a linguistic description lsd is defined for each map sd. In this way, the rewriting task becomes a plain translation: $\mathcal{M}(lsd, c) = glf$ whereas the pair (lsd, c) generates a micro-story for the robot. The direct interpretation of this micro-story produces the ambiguity-free grounded logic form glf, ensuring precise and meaningful grounding of commands in real-world contexts.

Towards describing a map in natural language. Much of the previous work using transformer-based architectures has approached knowledge injection by employing input graphs or synthetic descriptions during pre-training and/or fine-tuning [81, 149, 270]. The idea behind GrUT is to utilise a natural language description of the map and append it to the input sentence, as also suggested in [56] for tasks involving analogical reasoning.

In this approach, each entity is referenced by its (English) noun, which serves as its most commonly used lexical representation (e.g., the word book), along with its conceptual type. The association with the environment, or grounding, is achieved through its Existence Constraint (EC), which links the entity to its corresponding physical object in the environment. For example, the map paired with a command can generate the following description:

EC: "*b1, also known as book or volume or tome, is an instance of the class* Book *and t1, also known as table, is an instance of the class* Table."

If the book $b1$ and the table $t1$ are close to each other in the environment, an additional Proximity Constraint (PC) is added:

PC: "*b1 is near t1*"

For selected entities, further descriptions of relevant properties, such as containability, are included. This is captured through the Containability Constraint (CC). For instance, if there is a hypothetical cup $c1$:

CC: "*c1 can contain other objects*"

The combined description forms a micro-story, appended to the input as constraints separated by a "#" delimiter. This micro-story aids the SRL model in disambiguating

Algorithm 1 *GrUT* compilation Algorithm

```
1: procedure CONSTRUCT_INPUT(SENTENCE s = (w_1, ..., w_|s|))
2:     Entities ← ∅
3:     for i = 1, ..., |s| do
4:         Entities ← Entities ∪ get_candidate_entities(w_i)
5:     end for
6:     ec ← “”                                              ▷ Existence Constraints
7:     for e ∈ Entities do
8:         ec ← ec+“ # ”+get_ref(e)+“ also known as ”+get_lexical_ref(e)+
9:                                          “ is an instance of class ”+get_class(e)
10:    end for
11:    cc ← “”                                              ▷ Containability Constraints
12:    for e ∈ Entities do
13:        if containability(e) then
14:            cc ← cc+“ # ”+get_ref(e)+“ can contain other objects”
15:        end if
16:    end for
17:    pc ← “”                                              ▷ Proximity Constraints
18:    for e_1 ∈ Entities do
19:        for e_2 ∈ Entities do
20:            if e_1 ≠ e_2 ∧ distance(e_1, e_2) < τ then
21:                pc ← pc+“ # ”+get_ref(e_1)+“ is near ”+get_ref(e_2)
22:            end if
23:        end for
24:        Entities ← Entities − {e_1}
25:    end for
26:    return s + ec + pc + cc
27: end procedure
```

situations like those in (4.1) versus (4.2). When the spatial constraint PC is true, the model resolves the ambiguous syntax in (4.1) and produces the corresponding role-labeled logical form:

$$\textsc{Taking}(\textsc{Theme}(\text{“}\textit{the book on the table}\text{”})).$$

Here, the book $b1$, referenced through its proximity to the table $t1$, is identified as the THEME of the TAKING predicate. The extraction algorithm that processes the map and generates the linguistic description lsd for a given command c is detailed in Algorithm 1.

As demonstrated in [56], knowledge can be injected into Transformer models during training by appending it directly to the input. Similarly, in GrUT, since SRL is deeply rooted in linguistic theories like Frame Semantics, this knowledge was incorporated by adding textual information about Frames and Lexical Units to the command itself. This approach makes all potential frames associated with any lexical unit in the command explicit, enriching the input to allow the Transformer to learn the properties and relationships of frames and their roles. This enrichment helps prime frame annotation in the output and resolve underlying

ambiguities.

For instance, for the command "*Bring the book on the table*", the following description would be appended: "*bring can evoke TAKING or BRINGING*"[1] Similarly, for "*go along with me*", the appended description would be: "*go along can evoke COTHEME*". This approach highlights how a verb like "*bring*" can evoke multiple frames, such as TAKING or BRINGING, which can only be disambiguated by considering the input command and the current state of the environment.

Experimental Setup and Evaluation. The impact of the proposed approach on the semantic interpretation of user utterances was evaluated in a house Service Robotics scenario. The evaluation was conducted using the Human-Robot Interaction Corpus 2.0 (HuRIC)[2], which contains 656 English utterances paired with interpretations relative to explicit logically described maps, as discussed in [266]. Quantitative SRL measures were applied to assess various SRL aspects, and results for each run were reported in terms of F-Measure (F1). A 10-fold cross-validation schema with an 80/10/10 split between training, validation, and test data was employed.

The following aspects of `GrUT` are evaluated:

- **Frame Prediction (FP)**: This measures the ability of the model to correctly generate the names of frames evoked by the voice command. It is evaluated using the F1-score, where Precision and Recall reflect the capability of `GrUT` to recover the correct frame(s) expressed in the spoken command.

- **Argument Identification and Classification - Exact Match (AIC-EM)** This evaluation assesses the ability of the system to correctly generate the names of the arguments evoked by the command and associate them entirely with the entities that evoke those arguments. AIC-ExM is measured using the F1-score, where Precision and Recall are calculated for produced arguments that perfectly match the gold standard in their complete form. This includes the frame, type of argument, and corresponding span of text for the entities.

- **Argument Identification and Classification - Head Match (AIC-HM)** This relaxed evaluation requires the model to associate each argument with at least the correct Entity Head. AIC-HM is measured using the F1-score, where Precision and

[1]The frames linked to a lexical unit, such as *bring*, are only those defined in FrameNet and relevant to the domain knowledge base. In this case, *TAKING* and *BRINGING* are the only possible frames for *bring* in the HuRIC dataset, which focuses on robotic command language.

[2]https://github.com/crux82/huric

Recall reflect the capability of `GrUT` to recover the correct arguments by type and span of text of the entity as expressed in the spoken command.

For example, for the SRL of the phrase "*take the book*" as TAKING(THEME("*book*")), AIC-EM would score 0 because only "book" is associated with the THEME instead of the entire span "the book", but AIC-HM would score 1 because "book" corresponds to the semantic head of the argument. Unlike previous approaches that apply independent models for the FP and AIC tasks, the `GrUT` approach proposes these as side effects of a single, monolithic rewriting process. In this approach, a single Transformer model is fine-tuned jointly on both tasks, sharing weights across them. This integration allows for a full interpretation of input commands in real-world scenarios.

To test the feasibility of this approach, simple evaluations were designed to examine the effect of adding different types of input information on the output of the model. The input to the Transformer consisted of the user command and additional information, including a natural language description of the map (as in Algorithm 1) and lexical unit (LU) descriptions triggered by the input text. The goal was to measure the contribution of these features and determine the optimal configuration. The models trained for evaluation were as follows:

- $BART_{base}$: Only the linguistic command was provided as input, serving as the baseline to compare other features.
- $GrUT$: The linguistic map description was appended to the user command.
- $GrUT_{LU}$: In addition to the map description, descriptions of each frame and lexical unit (LU) evoked by the input were appended to the command.

In Table 4.1, we provide examples of input and output for the models trained under this setup. The $BART_{base}$ model uses only linguistic information to produce SRL outputs, making it the simplest model. For instance, for the sentence "*Bring the book on the table*", $BART_{base}$ has to choose between the frames BRINGING and TAKING without any prior knowledge, relying solely on information derived from the training data. In contrast, $GrUT_{LU}$ derives this knowledge directly from its input and focuses on disambiguation. Additionally, all models must decide the role of the argument "table" (i.e., GOAL, SOURCE, or THEME), which depends on the proximity of the "book" to the "table" in the environment. Unlike $BART_{base}$, both $GrUT$ and $GrUT_{LU}$ are aware of the map, given the existential (EC) and proximity (PC) evidence incorporated into the input text. Table C.1 in the Appendix details the parameter values used for training the models, determined through a grid search policy.

Accuracy measures are reported in Table 4.2. We compare against Lu4R [266], the current state of the art for SRL on the HuRIC dataset. In general, all BART-based models

Model	Input	Output
$BART_{base}$	*"Bring the book on the table"*	TAKING(THEME("*the book on the table*"))
GrUT	*"Bring the book on the table # b1, also known as book or tome, is an instance of the class VOLUME and t1, also known as table, is an instance of the class TABLE # b1 is near t1"*	TAKING(THEME("*the book on the table*"))
$GrUT_{LU}$	*"Bring the book on the table # take can evoke TAKING or BRINGING # b1, also known as book or tome, is an instance of the class VOLUME and t1, also known as table, is an instance of the class TABLE # b1 is near t1"*	TAKING(THEME("*the book on the table*"))

Table 4.1: Different Input and Output examples for this testing set.

are comparable to Lu4R on the FP task and perform significantly better on the AIC task. It is worth noting that FP is treated as an independent task in Lu4R, with specific supervised learning applied. In contrast, seq2seq models like GrUT consider FP as a side effect of the end-to-end task. Interestingly, errors in FP introduced by seq2seq models are not always detrimental, as they may reflect nuanced interpretations. The second row in Table 4.2 presents results for $BART_{base}$. Despite being trained solely on the textual features of commands, the model achieves high accuracy, indicating that the Transformer architecture captures most of the relationships and properties inherent in the commands. The third row shows the performance of the *GrUT* model, trained on commands and linguistic map descriptions. *GrUT* improves upon Lu4R for the end-to-end tasks, achieving a 5% error reduction in *AIC-Exact Match* and a 2% error reduction in *AIC-Head Match*. These results confirm the hypothesis that map descriptions are essential for accurate interpretation. The last row presents results for $GrUT_{LU}$, trained on commands, map descriptions, and Frame Semantics information. While $GrUT_{LU}$ shows slight improvements in the Frame Prediction task, it does not positively impact the EM or HM measures for the end-to-end task. This suggests that Frame Semantics information may add complexity without yielding substantial benefits for these metrics.

Overall, the results demonstrate that an end-to-end approach to SRL is both viable and capable of achieving state-of-the-art performance. While LU4R employs independently trained FP and AIC classifiers, the seq2seq approach of GrUT shows significant improvements. LU4R relies on a cascade of three classification steps, with AIC composed of two

Model	FP	AIC-EM	AIC-HM
LU4R [266]	**95.94%**	87.77%	93.11%
$BART_{base}$	91.29%	84.26%	91.24%
GrUT	93.28%	**88.41%**	**93.29%**
$GrUT_{LU}$	93.34%	86.61%	92.00%

Table 4.2: Comparative Evaluation on the Frame Prediction *FP*, Argument Identification and Classification *AIC* tasks of the different SRL models: Exact Match (*EM*) and Head Match (*HM*) are the different metrics for AIC.

stages: boundary detection (Argument Identification) and argument classification (AC). In contrast, GrUT and $GrUT_{LU}$ consolidate these three independent inferences into a single monolithic rewriting process that also performs logical form compilation. The superior quality of AIC tasks achieved by the GrUT model establishes a new benchmark for SRL performance.

The availability of map descriptions proves to be consistently beneficial, as evidenced by the significant improvements of GrUT and $GrUT_{LU}$ over the baseline $BART_{base}$. *GrUT* achieves an error reduction of 26%, while $GrUT_{LU}$ 15% compared to $BART_{base}$.

Furthermore, the incorporation of frame and lexical unit evidence in $GrUT_{LU}$ yields a notable improvement over $BART_{base}$, raising the accuracy from 91.29% to 93.34%, corresponding to an error reduction of 23.53%. However, $GrUT_{LU}$ shows a decline in accuracy for FP and AIC tasks compared to LU4R and in the AIC task compared to GrUT. This suggests that the additional lexical unit descriptions may over-enrich the input text, occasionally introducing misleading information. For instance, the correct SRL for the command "*go get my book from the shelf*" is:

$$\textsc{Bringing}(\textsc{Theme}(\text{“}\textit{my book}\text{”}), \textsc{Source}(\text{“}\textit{the shelf}\text{”})).$$

However, when lexical unit descriptions are appended to the input, such as "*go can evoke MOTION and get can evoke BRINGING*", the model produces the incorrect SRL:

$$\textsc{Motion}(\textsc{Goal}(\text{“}\textit{the shelf}\text{”}) \;\&\; \textsc{Bringing}(\textsc{Theme}(\text{“}\textit{my book}\text{”})).$$

In this case, the logical form differs from the intended one, yet the behaviour of the robot remains unaffected, as the implied plan is identical: moving to the shelf and then bringing the book to the user. Nevertheless, these cases negatively impact the performance scores of $GrUT_{LU}$ in FP and AIC tasks, particularly when compared to the simpler GrUT model. Despite this, the overall semantic capabilities of the neural system remain robust, and the observed drop in performance does not undermine the practical effectiveness of the model.

The role of the Map Descriptions. To study the impact of map descriptions and confirm their importance, we selected cases where spatial relationships between entities determine the identification of the appropriate frame. A subset of 34 complex sentences was chosen, where different interpretations depended explicitly on the spatial relations within the corresponding map. This focused dataset was used to compare $BART_{base}$ with $GrUT$, evaluating the effect of map descriptions on Semantic Role Labeling quality.

Table 4.3 presents examples from this subset. The first column contains the command, as transcribed from speech; the "Map Description" column includes the description compiled using Algorithm 1, and the "Output" column shows the desired (gold-standard) interpretation. This test set highlights the contribution of the proposed $GrUT$ approach. For instance, in the first row, the correct interpretation is TAKING (rather than BRINGING), which is determined by the spatial proximity (*near*) of entities $s5$ ("phone"), $s4$ ("table"), and $a1$ ("tv"). In another example, the second sentence specifies "*table*" as the filler for the GOAL argument of PLACING, rather than "*dining room*". This interpretation is justified by the spatial relationship in the map: the table $h6$ is far from the bottle $p2$. The absence of a mentioned *near* relationship in the map description implicitly negates proximity, thereby guiding the SRL interpretation.

These examples underscore the critical role of map descriptions in resolving ambiguities and ensuring accurate SRL, particularly in cases where spatial relationships are key to semantic interpretation. On the restricted dataset of 34 instances, $GrUT$ achieves an improvement of approximately 12% in F1 score, increasing from 78% with the baseline $BART_{base}$ model to 90%. This demonstrates that the Transformer effectively learns and utilises spatial constraints derived from distance relationships in the map, significantly enhancing its ability to interpret commands accurately in scenarios where spatial relations are critical.

Error Analysis. Even though Transformers introduce some errors in Frame Prediction (FP), the resulting interpretations are semantically acceptable and roughly equivalent to the gold standard. Below, we outline three main classes of errors, with examples provided in Table 4.4:

- **Frame Prediction Errors**: In the first row, the model predicts incorrect frames. However, the predicted frames often have closely related meanings that preserve the intended and useful senses of the command.

- **Span Errors**: The second row highlights cases where the model assigns the wrong span to a given argument type. For instance, it may assign "*the studio*" instead of the correct span "*the table*" for the argument GOAL. These errors are more problematic, as they risk altering the logical structure of the interpretation. However, even in these

Sentence	Map Description	Output
take the phone near the tv on the table	*s5, also known as phone or cellphone, is an instance of class PHONE and a1, also known as television, is an instance of class TELEVISION and s4, also known as table, is an instance of class TABLE # s5 is near a1 & s5 is near s4 & a1 is near s4*	TAKING(THEME("*the phone*"))
put the bottle on the table in the dining room	*p2, also known as bottle or nursing bottle, is an instance of class BOTTLE and h6, also known as table, is an instance of class TABLE*	PLACING(THEME("*the bottle*"), GOAL("*on the table*"))
please robot take the box on the table on the couch	*t3, also known as box or crate, is an instance of class BOX and d1, also known as table, is an instance of class TABLE and i4, also known as couch or sofa, is an instance of class COUCH # t3 is near d1*	BINGING(THEME("*the box*"), GOAL("*on the couch*"))
move to the bedroom and take the mouse near the laptop on the bed	*p7 also known as bedroom is an instance of class BEDROOM & s3 also known as mouse is an instance of class MOUSE & x9 also known as computer is an instance of class COMPUTER & i9 also known as bed or bunk is an instance of class BED # s3 is near i9 & x9 is near i9*	MOTION(GOAL("*the bedroom*")) & TAKING(THEME("*the mouse*"))

Table 4.3: Examples of sentences for which map descriptions are crucial to the correct identification of the evoked Frames and respective Arguments.

cases, the overall logical form often remains intact.

- **Argument Errors**: As shown in the third row, these errors occur when the model assigns incorrect argument labels, such as GOAL instead of DIRECTION, to the correct spans (e.g., "*forward*"). Despite the mismatch, the predicted command remains semantically close to the gold standard, as DIRECTION in this case aligns with the GOAL interpretation.

Given that this Transformer-based approach achieves results comparable to state-of-the-art models on SRL tasks, an intriguing extension would involve generating a Situated Interpretation of the sentence that moves beyond the linguistic level. By leveraging the Knowledge Base (KB), the fillers of the arguments (e.g., *book* or *table*) can be replaced with their corresponding identifiers (e.g., `b1` or `t1` from the KB). This linking is critical, as it ensures the robot understands that it needs to interact with a specific entity (e.g., the book

Input	Target	Prediction
leave the book in the bedroom # n8, also known as book or booklet, is an instance of class BOOK and f9, also known as bedroom, is an instance of class BEDROOM	RELEASING(THEME("*the book*"), GOAL("*the bedroom*"))	PLACING(THEME("*the book*"), GOAL("*the bedroom*"))
put the bottle on the table in the studio # d3, also known as bottle or nursing bottle, is an instance of class BOTTLE and e5, also known as table, is an instance of class TABLE and s3, also known as studio, is an instance of class STUDIO # e5 is near s3 # d3 can contain other objects	PLACING(THEME("*the bottle*"), GOAL("*the table*"))	PLACING(THEME("*the bottle*"), GOAL("*the studio*"))
could you please move forward # t2, also known as robot or you, is an instance of class ROBOT	MOTION(THEME("*you*"), DIRECTION("*forward*"))	MOTION(THEME("*you*"), GOAL("*forward*"))

Table 4.4: Error analysis table: Input is *GrUT* model input, Target is the expected SRL in logical form (the so called *gold*) and Prediction is the model output.

with identifier `b1`) rather than any generic instance of the same type. In the next section, we present the extension of `GrUT` for Situated SRL, demonstrating how this approach grounds linguistic interpretations in real-world contexts.

4.1.4 Situated Semantic Role Labeling with GrUT

In the first extension of this work, GrUT [111] was enhanced to ground language in the physical world by directly interpreting commands and mapping them to entities within the robot's environment. For example, given the command Bring the book on the table, GrUT generates a logical form grounded in the Knowledge Base (KB), such as: 'BRINGING(THEME(B1), GOAL(T1))', where `b1` represents the identified book, and `t1` represents the table. This interpretation applies when the book and table are spatially distant. Conversely, when the book is already on the table, GrUT produces the formula: 'TAKING(THEME(B1), SOURCE(T1))', identifying that the book is already on the table and must be taken, potentially for relocation. The key advancement is making the interpretation Situated by linking physical entities with their linguistic references (i.e., the labels humans use to denote them) through the use of identifiers in the interpretation.

In general, as outlined in [266], it is assumed that each entity in a map is associated with one or more labels to enable grounding. For instance, linguistic references like volume or book may correspond to the object `b1`. This approach, used in both [266] and [108], allows

the retrieval of all entities involved in a command from the map. The simplest method selects all entities whose denotation matches a word in the command, as applied in [111]. However, this assumption is unrealistic in natural language interaction, where synonyms or paraphrases are often used to refer to objects. For example, "*take the handbook ...*" or "*take the tome ...*" can be interchangeable with "*take the book ...*". To address this limitation, a more sophisticated retrieval function is required, leveraging expressive associations between linguistic labels. This involves using neural semantic similarity functions to identify objects referenced in commands, even when synonyms or paraphrases are employed. To the best of my knowledge, this enhancement expands on recent research such as [130] and [33] by introducing the first end-to-end technique for Fully Grounded Linguistic Interpretation. This approach relies on an explicit logical description of the environment, integrating the KB to achieve situated and context-aware command interpretation.

`GrUT` assumes that each entity e in the environment is enriched with a set of lexical references $LR(e) = \{w_1^e, \ldots, w_l^e\}$, linking words $(w_1, \ldots, w_{|s|})$ in the sentence s. For example, consider the volume v_1 with lexical reference $LR(v_1) = \{volume, book\}$.A robust linguistic grounding function is essential for `GrUT` to construct the map description. Algorithm 2 defines the policy for retrieving entities involved in the utterance. It retrieves all entities with significant lexical similarity to nouns in s, using three *LexicalSimilarity* functions of increasing expressiveness:

1. **Exact Match**: This simplest function performs a direct string comparison, returning 1 if the strings match and 0 otherwise. Entities with lexical references perfectly matching a word in the command are retrieved. While precise, this method fails when users employ synonyms (e.g., referring to $v1$ with *handbook*, *tome*, or *manual*). Additionally, it requires a comprehensive map construction, which is impractical for covering all possible lexical references.

2. **Levenshtein similarity**: a "soft" string matching approach that accounts for minor differences between strings. The similarity function $LevSim$ is defined as:

$$LevSim(w_i, w_j) = 1 - \frac{LevDist(w_i, w_j)}{|longest(w_i, w_j)|} \tag{4.3}$$

   where $LevDist$ is the Levenshtein distance and $|longest(w_i, w_j)$ is the length of the longer word. $LevSim$ ranges between 0 (totally different) and 1 (identical). This similarity function is more robust in linking slightly different input strings (e.g., *book* vs *handbook*) and it may capture some sort of morphological analogy between words, but it fails when the user refers to entities using synonyms. As an example,

$LevDist(\text{“book”}, \text{“handbook”}) = 4$
$|longest(\text{“book”}, \text{“handbook”})| = 8$
$LevSim(\text{“book”}, \text{“handbook”}) = 1 - \frac{1}{2} = \frac{1}{2}$.
It means that half the word is identical and the other half needs some transformation. Still, it fails when the user refers to entities using synonyms, such as *tome*. In fact, the value of $LevSim(\text{“book”}, \text{“tome”}) = 1 - \frac{3}{4} = \frac{1}{4}$.

3. **Neural Semantic similarity**: This function computes cosine similarity[3] between word embeddings [227], linking words with paradigmatic relationships, such as quasi-synonymy. Word embeddings are generated using the `Word2vec` model [184], trained on English Wikipedia. For example, the Neural Semantic similarity between the word *“book”* and the lexical reference *“tome”* is 0.96, indicating a strong semantic similarity, reflecting their interchangeable use.

To prevent over-retrieval when using smoothed measures (e.g., neural embeddings) in maps with numerous objects, a similarity threshold τ is applied. Only entities with significant similarity to a command word are retrieved. This method remains robust when multiple entities are retrieved for the same word. For instance, if there are several books in the environment, the model should select the book relevant to the user's intent (e.g., the one *on the table*). If no matching entity exists, the system should detect the inconsistency and notify the user with a message (“*The command is not executable*”) and an explanation (“*No objects can be linked to the mention book*”). Although the input text may redundantly include all retrieved entities in the map description, the output arguments should contain only relevant entities. The attention mechanism of the Transformer is expected to filter out irrelevant entities and focus on valid ones to produce the correct output. This ensures the final interpretation aligns with the user's intent and the environmental context.

Experimental Setup and Evaluation. The evaluation of the extended GrUT model was conducted on the same dataset, HuRIC, but with an additional step: transforming the interpretation from a purely linguistic level to a situated level by linking the arguments to identifiers of entities in the surrounding environment. This grounding process, described previously, links linguistic references to their corresponding entities. On average, each entity is represented by 1.37 lexical references, providing more linguistic variability compared to the earlier dataset. The same 10-fold cross-validation policy and hyperparameters (detailed in Table C.1) were applied.

To assess the quality of interpretations, the following tasks were used, as in earlier evaluations: Frame Prediction (FP) that easures the accuracy of predicting frame names evoked by

[3]To align with the other measures, cosine similarity values are clipped to the range $[0, 1]$.

Algorithm 2 Entity Retrieval Algorithm

```
1: procedure GET_CANDIDATE_ENTITIES(Word w, LexicalSimilarity ls, Threshold τ_ls)
2:     Candidate_Entities ← ∅
3:     for e ∈ KB_Entities do
4:         for lex_ref ∈ LR(e) do                    ▷ For each lexical reference
5:             if ls(w, lex_ref) > τ_ls then
6:                 Candidate_Entities ← Candidate_Entities ∪ e
7:             end if
8:         end for
9:     end for
10:    return Candidate_Entities
11: end procedure
```

the command; Argument Identification and Classification - Exact Match (AIC-ExM), which tests the ability of the system to generate arguments and link them to the correct entities, requiring exact alignment with the gold standard; Argument Identification and Classification - Head Match (AIC-HeM), that is a more relaxed version of AIC-ExM, where an argument is considered correct if it includes the semantic head of the referenced entity.

The evaluation also investigated the impact of three Lexical Similarity functions used during the entity retrieval step. Each function was tuned using a threshold τ, optimised on the validation set to maximise F1 scores for entity retrieval.In this subtask, Precision is measured as the average percentage of entities that are correctly retrieved when constructing the map description, while Recall is measured as the average percentage of entities that were expected to be retrieved as mentioned in the command. In particular, under the Exact Match policy a definition of $\tau_{\text{EM}} = 0.50$ was applied, while under the policy based on Levenshtein Similarity and the Word Embedding one, $\tau_{\text{Lev}} = 0.80$ and $\tau_{\text{WE}} = 0.55$ are respectively used.

Table 4.5 summarises the experimental results. The first rows show the performance of GrUT + *ExPostGrounding*, a strong baseline derived from the earlier GrUT model [108]. `GrUT` generates logic forms expressing the interpretation of commands at a linguistic level and it was reported to achieve 92.28% of F1 in the FP task, 88.41% in the Argument Identification and Classification Exact match (AIC-ExM) and 93.29% as the score in recovering the Semantic Head of the individual arguments.

For the baseline, GrUT predictions were reused, and grounding was applied retrospectively. For each argument, such as Goal("on the table near the window"), the first noun (table) was identified, and the entity maximising lexical similarity replaced the argument. If no suitable entity was retrieved (or the threshold was not exceeded), the argument was removed. Three Lexical Similarity functions were evaluated for this process, i.e. based on Exact Match (EM), Levenshtein Similarity (LS) and semantic similarity estimated over Word

Embeddings (WE). These results underline the trade-offs in precision and flexibility across the similarity functions, with Word Embeddings offering a strong balance between semantic understanding and robust retrieval.

Model	Retrieval Policy	FP	AIC-ExM	AIC-Enty
GrUT + *ExPostGrounding*	Exact Match	92.18%	78.62%	80.00%
	Levenshtein Sim.		79.80%	81.37%
	Word Embeddings		80.91%	82.22%
GrUT *End-to-End*	Exact Match	90.38%	83.16%	84.79%
	Levenshtein Sim.	**92.40%**	84.24%	85.66%
	Word Embeddings	91.90%	**90.03%**	**91.46%**

Table 4.5: Comparative Evaluation on the Frame Prediction FP, Argument Identification and Classification AIC tasks of the different G-SRL models: Exact Match (ExM) and Head Match ($AIC\text{-}Enty$) are the different metrics for AIC.

The application of the grounding function introduces a notable performance drop, with AIC-ExM decreasing from 88.41% to 80.91%. This decline is largely attributed to approximately 10% of entities in HuRIC having lexical references that do not match any words used in candidate arguments.

For simpler argument-level AIC tasks, the Exact Match retrieval method achieves 78.62% and 80.00%, while the Word2Vec (W2V) retrieval method raises these values to 80.91% and 82.22%, respectively. This improvement demonstrates the utility of word embeddings and vector similarity in recovering connections. However, the relatively small difference between Exact Match and W2V methods suggests that the lexical references in HuRIC closely align with the command words. Notably, the entity retrieval policy does not significantly impact the Frame Prediction subtask.

When applying the proposed GrUT-*End-to-End* model, the Transformer-based approach outperforms the baselines, achieving a 50% error reduction on AIC tasks. For example, $ExPostGrounding_{W2V}$ at 80.91% improves to $End\text{-}to\text{-}End_{W2V}$ at 90.03%. Neural representation-based lexical grounding proves robust in entity retrieval, while the Transformer's attention mechanism effectively grounds interpretations. The model also demonstrates improvements over the original GrUT, effectively mapping entities from the map descriptions associated with the input command to the correct entities in the output interpretations. The differences between Exact Match (EM), Levenshtein Similarity (LS), and Word Embeddings (WE) retrieval methods become more pronounced in the end-to-end setup. Exact Match fails when no entity is retrieved, resulting in arguments being excluded from the final interpretation. Conversely, smoother measures like WEs may retrieve a superset of correct entities, intro-

ducing noise. However, the Transformer effectively prunes irrelevant entities, maintaining high accuracy for grounded interpretations. Retrieval policies also influence Frame Prediction tasks, albeit to a lesser degree. Exact Match often overlooks entities not retrieved in the input, negatively affecting frame disambiguation. In contrast, LS improves frame disambiguation quality over WE. This suggests that while "extra" entities retrieved by WE can enhance grounding in some cases, they may hinder frame prediction, highlighting a trade-off between entity retrieval and frame disambiguation.

Retrieval Policy	**Output**	**Correct**
Exact Match	BRINGING(THEME('*the book*'), GOAL('*the bedroom*'))	NO
Levenshtein Similarity	BRINGING(THEME('*the book*'), GOAL(s_6))	NO
Word Embeddings	BRINGING(THEME(w_6)) GOAL(s_6))	YES

Table 4.6: Error analysis of the `GrUT` - *End-to-End* model (and different retrieval policies) applied to the command "*take the book that is in the kitchen*".

The error analysis, in Table 4.6, highlights that most misinterpretations stem from errors during the linguistic grounding phase. The example involves a simple map with two entities: w_6, an instance of the `Book` class with the single lexical reference *volume*, and s_6, an instance of the `Room` class with the single lexical reference *guest room*. Given the command "take the book to the bed room", the `GrUT`-*End-to-End* model using Exact Match fails to retrieve any entities. While the model infers the correct predicate and arguments, it merely rewrites the input text. Using Levenshtein Similarity, the model successfully links "guest room" to the map, allowing the second argument to be correctly grounded. When applying cosine similarity within the Neural Word Embedding space, the model generates the enriched input: "*take the book to the bed room # w_6 also known as volume is an instance of class BOOK & s_6 also known as guest room is an instance of class BEDROOM & w_6 is far from s_6*" This leads to the correct interpretation. The majority of errors occur with entities that have only one lexical reference, often uncommon nouns like *volume* for the `Book` type entity, which the grounding function struggles to retrieve.

The experimental results underscore the robustness of the proposed methods, particularly when compared with traditional architectures that sequentially handle linguistic interpretation and entity grounding. This approach offers broad applicability, requiring minimal adaptation to specific scenarios or domains. However, these results are currently limited to English commands. Expanding this work to support multiple languages, particularly Italian, presents a compelling direction for future research. The next Section explores this expansion.

4.1.5 Multilingual GrUT

The subsequent extension [113] of `GrUT` addresses the challenge of enabling grounded language understanding in multilingual contexts. One of the primary hurdles is managing diverse linguistic descriptions across different languages while maintaining accurate grounding. To overcome this, `GrUT` incorporates advanced lexical similarity measures, such as Levenshtein similarity and neural embeddings, which enhance the retrieval of entities from the Knowledge Base (KB) even amidst linguistic variations. For instance, the architecture ensures that when a user refers to a "*book*" or a "*tome*" in English, or "*libro*" in Italian, the robot recognizes all terms as referring to the same object in the environment. By estimating entity relationships through language-independent measures, `GrUT` bridges linguistic differences effectively, enabling consistent grounding across languages. This multilingual capability is particularly crucial in scenarios where users may speak different languages or use varying terminologies to describe identical objects. `GrUT` ensures that commands in any supported language are interpreted accurately, allowing robots to operate seamlessly across diverse linguistic environments. Moreover, incorporating multilingual data in the training process accelerates model convergence and enriches the final task performance. Commands in different languages expose the model to varied linguistic nuances, broadening its knowledge representation and improving its versatility. Despite the abundance of models and resources for English, a significant gap exists for other languages. Evaluating `GrUT` in a multilingual setting, especially extending its application to Italian, is a promising step. This involves leveraging the same principles to develop and evaluate models tailored for Italian and multilingual LLMs. Since Frame Semantics is inherently language-independent, the grounded predicate for an utterance remains consistent, regardless of whether the command is in Italian or English. This demonstrates the potential of `GrUT` to support multilingual interaction while maintaining robust semantic grounding across languages.

The primary challenges in Grounded Language Understanding include:

- **Multilingual understanding**: `GrUT` addresses the complexities of interpreting commands in multiple languages while maintaining grounding precision. The challenge lies in managing linguistic diversity without compromising the consistency of the grounding process. By leveraging lexical similarity measures and neural embeddings, `GrUT` provides a robust framework for handling multilingual commands, ensuring seamless interactions across diverse languages.

- **Entity disambiguation**: Accurately retrieving and interpreting the intended entity becomes especially challenging in environments with multiple similar objects. Disambiguation is achieved using detailed attributes and spatial relationships, allowing the

model to distinguish between entities. The attention mechanism within Transformers is critical for focusing on the most relevant aspects of the input, effectively resolving ambiguities.

One approach to support multiple languages is translating the Semantic Map into target languages, such as Italian, by directly converting the English version. This information is then stored in the Semantic Map. However, this translational approach can be computationally expensive, as the storage requirements grow linearly with the number of supported languages. Furthermore, any update to the Semantic Map necessitates additional translation steps, increasing the overhead.

A more dynamic method involves translating the Semantic Map into the user's command language on-the-fly, i.e., during interpretation. This approach uses a Language Recognizer module[4] to identify the language of the input utterance, such as Italian. Based on the recognized language, a corresponding Semantic Map is dynamically generated, enabling real-time interpretation tailored to the user's linguistic context.

Alternatively, a more universal approach involves adopting a Multi-Lingual process that minimizes dependency on the command's language. Two alternatives in this scenario are: *(1)* employing dedicated language-specific models, where commands are interpreted using models explicitly trained for each language (see Fig. 4.2); *(2)* adopting a multi-lingual model, where commands are interpreted using a single model capable of handling multiple languages simultaneously (see Fig. 4.3). For this specific work, we relied on training data enriched with lexical references in Italian, ensuring compatibility with multi-lingual interpretation processes. As such, solutions involving machine translation are excluded from further discussion.

Multi-lingual End-to-end Grounded Semantic Role Labeling. A multilingual setting is ideal for interpreting commands expressed in various languages, leveraging linguistic knowledge to achieve accurate understanding across diverse inputs. Figure 4.2 demonstrates a workflow involving language-specific models. When a command is issued in a specific language L (e.g., Italian or English), a `Language Recognizer` model identifies L. This information is then passed to the `Map Description Generator` module, which, along with the utterance, extracts the relevant entities mentioned in the command (e.g., BOOK, TABLE) and generates a textual description of the map in L. A language-specific model then combines the map description with the command to produce an interpretation.

For instance, consider a command issued in Italian. Entities must have appropriate lexi-

[4]For this purpose, we employed the language recognizer available at https://huggingface.co/dinalzein/xlm-roberta-base-finetuned-language-identification, achieving 99.59% accuracy on the Language Identification Dataset: https://huggingface.co/datasets/papluca/language-identification.

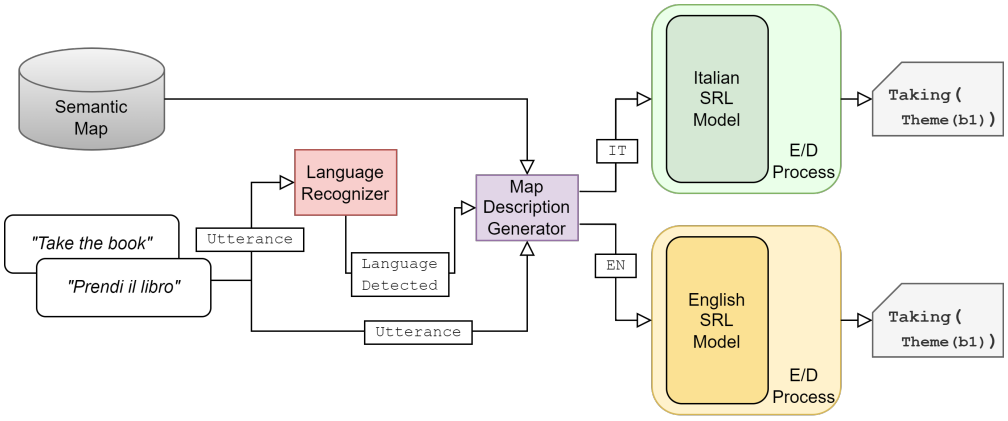


Figure 4.2: Language-specific Workflow.

cal references in Italian to be identified and processed. Using Italian lexical references (e.g., "*tomo*" or "*tavolo*"), the map descriptions are generated in Italian to take full advantage of language-specific LLMs. When the command "*Porta il volume sul tavolo vicino la finestra*" is given, the `Language Recognizer` identifies Italian as the input language. The `Map Description Generator` then produces the following description, integrating the EC, PC, and CC information (as discussed earlier):

"b_1, conosciuto anche come libro o volume, è un'istanza della classe LIBRO*, t_1, conosciuto anche come tavolo, è un'instanza della classe* TAVOLO *e w_1, conosciuto anche come finestra, è un'instanza della classe* FINESTRA *# t_1 è vicino w_1".*

Subsequently, an Italian Transformer model, such as IT5 [231] or BART-IT [142], is fine-tuned on Italian commands and lexical references to produce the grounded interpretation:

$$\textsc{Bringing}\big(\textsc{Theme}(b_1), \textsc{Goal}(t_1)\big) \tag{4.4}$$

This interpretation enables a robot to execute a plan as follows: *(1)* move where the book b_1 is, *(2)* take b_1, then *(3)* bring it to the table t_1, that is near w_1. When the same command is expressed in English, the `Language Recognizer` activates the English lexical references to retrieve the entities, and an English-specific model is triggered. Pre-trained models like T5 [217] or BART [152], fine-tuned on English commands and lexical references, are employed. Despite the linguistic differences, the output remains the same logical form, i.e., the logical form in the example from equation (4.4), as it is language-neutral.

Figure 4.3 illustrates the multi-lingual workflow for interpreting natural language commands. Similar to the language-specific approach, this workflow begins with a `Language Recognizer` that detects the language of the input command. The `Map Description Generator`

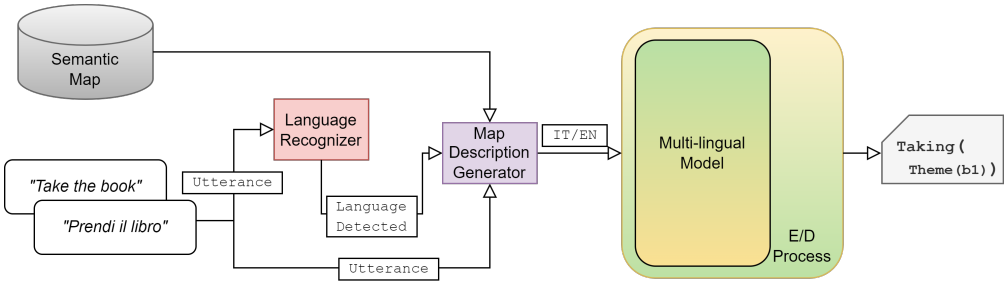


Figure 4.3: Multi-lingual Workflow.

then produces a natural language description of the environment in the same language as the command. However, in the final step, the map description and input command are sent to a multi-lingual model capable of interpreting commands across languages, eliminating the need for separate models for each language. This approach enables the training and maintenance of a single multi-lingual model for the interpretation process. A model like mT5 [282] is fine-tuned using combined examples in both Italian and English, making it adaptable to commands in either language.

It is worth noting that the assumption that language-specific solutions are more computationally complex than language-independent solutions does not always hold. Multi-lingual models like mT5 typically have significantly more parameters than language-specific models such as T5/BART or IT5/BART-IT. This is because multi-lingual models must account for a broader variety of linguistic phenomena across multiple languages. Furthermore, while multi-lingual models benefit from exposure to large-scale pre-training in multiple languages, they may require extensive fine-tuning to achieve comparable or better accuracy in generating interpretations. The added complexity of managing diverse languages can dilute the focus of the model on the nuances of any single language, potentially affecting the precision of its interpretations. Consequently, the choice between language-specific and language-independent solutions must carefully balance computational demands with the desired level of interpretative accuracy.

Experimental Setup and Evaluation. The experimental evaluation serves two primary objectives. Firstly, it demonstrates the applicability of the `GrUT` approach for interpreting robotic commands in multiple languages, specifically English and Italian. Secondly, it evaluates the effectiveness of a multilingual setting, exploring both language-specific and language-independent LLMs. The models tested were fine-tuned on their respective languages and evaluated under a uniform setup. Experiments were conducted on a T4 16GB GPU using the *Simpletransformers* and *Pytorch* packages for loading and training models

from Hugging Face[5]. The source code is openly available[6].

The evaluation utilised the HuRIC corpus, comprising 656 English and 241 Italian commands paired with interpretations in terms of predicates and arguments. The grounding process described earlier was applied, linking interpretations to entity identifiers in the environment. A 10-fold cross-validation setup was employed for its reliability, especially given the limited size of the Italian dataset.

The evaluation tasks remain consistent with those of previous sections:

- **Frame Prediction (FP)**: Measuring the ability of the models to accurately generate the names of Frames evoked by the voice command.
- **Argument Identification and Classification - Exact Match (AIC-ExM)**: Assessing the ability to correctly generate the names of Arguments, linking them to the entities evoking those Arguments.
- **Argument Identification and Classification - Head Match (AIC-HeM)**: A more relaxed metric where the model is rewarded if the Argument contains at least the Semantic Head of the correct linguistic reference.

Based on previous evaluations, the Neural Semantic Similarity (NSS) with a threshold $\tau_{En,NSS} = 0.55$ was selected for entity retrieval over the English dataset, as it yielded the best F1 score on the AIC task. For Italian, a slightly higher threshold of $\tau_{It,NSS} = 0.60$ was optimal. Moving forward, the NSS method was consistently used for all experiments across both languages.

Once the entity retrieval policy was established, a thorough evaluation of various English models was performed, including comparisons with language-independent multilingual models. Table 4.7 summarises the results for English. The first row reports the best-performing model from the previous section (`GrUT End-to-End` based on NSS, as outlined in Table 4.5). This model serves as the baseline, achieving: 91.90% F1 on Frame Prediction (FP), 90.03% F1 on Argument Identification and Classification - Exact Match (AIC-ExM), and 91.46% F1 on Argument Identification and Classification - Head Match (AIC-HeM). This baseline demonstrated competitive performance compared to state-of-the-art models for Grounded Semantic Role Labeling (G-SRL), establishing a strong reference point for subsequent evaluations.

Models such as T5 and mT5 were fine-tuned with different learning rates (lr), yielding varied performances. Among them, T5 with an lr of $5 \cdot 10^{-4}$ achieved the best performance at

[5] https://huggingface.co
[6] https://github.com/crux82/grut

Model	LR	FP	AIC-ExM	AIC-HeM
BART [111]	$5 \cdot 10^{-5}$	91.90	**90.03**	**91.46**
mT5	$5 \cdot 10^{-4}$	84.22	75.07	78.99
	$1 \cdot 10^{-4}$	86.51	80.18	83.66
	$5 \cdot 10^{-5}$	84.77	77.15	79.81
T5	$1 \cdot 10^{-3}$	91.68	82.06	84.28
	$5 \cdot 10^{-4}$	**93.06**	86.01	86.67
	$1 \cdot 10^{-4}$	89.26	83.56	85.17
	$5 \cdot 10^{-5}$	89.96	82.30	83.52

Table 4.7: Comparative Evaluation of `GrUT` in terms of F1 for the English models on the Frame Prediction FP, Argument Identification and Classification AIC tasks of the different G-SRL models: Exact Match (ExM) and Head Match (HeM) are the different metrics for AIC. In **bold** the best performance for each task.

93.06% on the FP task, with a drop of 4% compared to the baseline on the AIC task. While other T5 models with different learning rates also performed well, they failed to surpass state-of-the-art models in overall performance. In contrast, mT5 performed the lowest on both the FP and AIC tasks. The best mT5 with $1 \cdot 10^{-4}$ lr achieved 86.51% on the FP task, 80.18% as Exact Match, and 83.66% as Head Match on the AIC task. This difference in performance on the same test set between BART-based and T5-based models could be mainly attributed to the difference in the size of the pre-training data. BART appears to be more robust in the interpretation process, while T5 and mT5 appear to be more sensitive to unrepresented phenomena in the training dataset. For example, frames like Releasing or Giving, which constitute only 1.5% of the data (10 examples out of 656), posed challenges for all models. BART correctly interpreted 12 out of 19 such examples, outperforming T5 (9 correct examples) and mT5 (7 correct examples). This indicates that the pre-training phase of BART significantly enhanced its capacity to manage data sparsity, whereas larger, multilingual models like mT5 are more susceptible to underrepresented phenomena.

Table 4.8 presents results for the Italian dataset. The first row includes LU4R [266] as a soft baseline. While LU4R is the only model performing Semantic Role Labeling on this specific Italian dataset, its results are not directly comparable to those evaluated here, as LU4R operates solely at a linguistic level without performing the grounding step. The `GrUT` models are thus solving a much more complex task. This comparison highlights the limitations of LU4R in grounded scenarios, as its interpretations do not extend beyond linguistic constructs, underscoring the significance of the grounding process in enhancing interpretative capabilities for multilingual contexts.

Model	LR	FP	AIC-ExM	AIC-HeM
LU4R [266]	-	*95.32*	*77.67*	*86.35*
mT5	$1 \cdot 10^{-3}$	47.00	29.40	36.04
	$1 \cdot 10^{-4}$	41.99	37.95	41.33
IT5	$1 \cdot 10^{-3}$	55.74	43.26	45.95
	$1 \cdot 10^{-4}$	45.92	37.26	39.38
BART-IT	$1 \cdot 10^{-4}$	**81.38**	65.63	67.82
	$7 \cdot 10^{-5}$	80.07	**67.12**	**68.85**
	$5 \cdot 10^{-5}$	77.7	62.97	64.82

Table 4.8: Comparative Evaluation of `GrUT` in terms of F1 for the Italian models on the Frame Prediction FP, Argument Identification and Classification AIC tasks of the different G-SRL models. In **bold** the best performance for each task. In *italic* the performance of LU4R model as it is not directly comparable, given it carries out no Grounding.

Models such as mT5, IT5 and BART-IT were fine-tuned with different learning rates (lr). Among them, BART-IT with an lr of $1 \cdot 10^{-4}$ achieved the best performance, reaching 81.38% on the FP task, and with an lr of $7 \cdot 10^{-5}$ a 67.12% as Exact Match and 68.85% as Head Match on the AIC task. The other T5-based models performed poorly on both tasks, with IT5 (lr $1 \cdot 10^{-3}$) achieving a maximum of 55.74% for the FP task and 43.26% and 45.95% as Exact and Head Match respectively on the AIC task.

Augmenting the Italian dataset. To address the limitations of the small Italian dataset, consisting of only 241 utterances, methods for augmenting the dataset with additional data were explored. The underrepresentation of certain Frames, such as Inspecting, Giving, and Being_in_category, each accounting for only 2% of the total samples, highlights the need for augmentation. Fine-tuning large language models (LLMs) with such limited samples is challenging and can result in suboptimal performance. One straightforward method involves annotating new utterances manually by creating new Semantic Maps from scratch. However, this approach is resource-intensive. Alternatively, external datasets could be adapted, with preprocessing to align input commands, Semantic Maps, and interpretation outputs with the format of the original data.

A more efficient and scalable approach was adopted: translating the English HuRIC dataset into Italian while retaining the original contexts and interpretations. This approach leverages the language-independent nature of Frame Semantics, enabling the reuse of logical forms. By translating the 656 English utterances into Italian and pairing them with their original (language-independent) interpretations, the dataset was expanded significantly. The augmented dataset was used for training, while the original Italian dataset was split into

training, evaluation, and testing sets in an 80/10/10 ratio, with a 10-fold cross-validation method applied. DeepL[7] was used for automatic translation due to its high accuracy in contextually translating longer texts. To mitigate translation ambiguities, the English utterances were paired with their corresponding lexical references from the Semantic Map, providing additional context. For example, the English noun "*glass*" was correctly translated into Italian as "*bicchiere*" or "*calice*" depending on the context, such as in "*Bring me a glass of water*".

While no direct evaluation of the translation method was performed, its quality was indirectly assessed through the performance of the models trained on the augmented dataset. High-quality translations were expected to improve model performance, while poor translations would negatively affect results. The augmented dataset increased the representativeness of the training material, particularly benefiting models that rely on large amounts of data for fine-tuning. Furthermore, this augmentation rebalanced the Italian-to-English data ratio from approximately 1:2.7 to nearly 1:1 for the multilingual model (mT5), potentially enhancing its performance.

Model	**LR**	**FP**	**AIC-ExM**	**AIC-HeM**
LU4R [266]	-	*95.32*	*77.67*	*86.35*
mT5	$1 \cdot 10^{-3}$	82.26	59.36	66.43
	$1 \cdot 10^{-4}$	90.61	73,21	82.89
IT5	$1 \cdot 10^{-4}$	72.00	64.44	65.97
	$5 \cdot 10^{-5}$	65.48	60.02	61.85
BART-IT	$1 \cdot 10^{-4}$	94.45	76.61	79.50
	$5 \cdot 10^{-5}$	**96.86**	**82.30**	**85.19**
	$2 \cdot 10^{-5}$	95.17	79.47	82.89

Table 4.9: Comparative Evaluation of `GrUT` in terms of F1 for the Italian models fine-tuned on the augmented dataset, i.e. Italian one and the English translated. In **bold** the best performance for each task. In *italic* the performance of the LU4R model as it is not directly comparable.

The experiments in Table 4.8 were repeated with the augmented training material, while the test set remained unchanged. The results are reported in Table 4.9, with LU4R included as a soft baseline in the first row for reference. However, LU4R remains incomparable to the other models, as it does not perform grounded interpretation.

The IT5 model demonstrated a slight improvement across both tasks but remains less effective compared to other models. Conversely, BART-IT with a learning rate of $5 \cdot$

[7]Translation performed in February 2023 via https://deepl.com.

10^{-5}achieved state-of-the-art performance in both tasks, with an impressive 96.86% F1 on the Frame Prediction (FP) task, surpassing the baseline. It also scored 82.10% for Exact Match (AIC-ExM) and 85.19% for Head Match (AIC-HeM) on the Argument Identification and Classification (AIC) task. It is important to highlight the methodological difference: while LU4R addresses the overall SRL task through a cascade of token-wise classification steps, the Transformer-based models evaluated here generate the full text for the interpretation task. Consequently, the FP and AIC subtasks emerge naturally as side effects of the holistic generation process, rather than being explicitly modelled as independent steps. This distinction underscores the efficiency and versatility of the Transformer-based approach in handling SRL tasks.

Comparison between Mono and Multi-lingual models. The experimental evaluation aimed to compare the performance of the `GrUT` architecture in multilingual settings, specifically contrasting language-specific and multilingual Large Language Models. The results from Tables 4.7 and 4.8 indicate that language-independent models, like mT5, often underperform compared to language-specific models. Multilingual models appear to be more sensitive to the scale of phenomena in this task, requiring a significantly larger dataset to match the performance of language-specific models. This disparity is particularly evident in cases like Italian, where only about 250 examples span 15 distinct linguistic predicates. In such cases, injecting additional data through automatic machine translation proved highly effective in bridging the gap.

Another consideration is the computational cost of the workflows depicted in Figures 4.2 and 4.3. While both workflows use the same `Language Recognizer` and `Map Description Generator` modules, the key difference lies in the interpretation models. Language-specific models, such as BART and BART-IT, with 140 million parameters each, collectively have a parameter count of 280 million, which is less than half of mT5's 580 million. Moreover, BART-based models required less training time on the same GPU configuration as mT5, further emphasizing the efficiency of the language-specific workflow depicted in Figure 4.2. This approach not only delivers better performance but also optimizes model size, making it a more scalable solution for introducing additional languages based on computational resources.

An intriguing direction involves enhancing the interaction between models and users when the model fails to ground an entity from its KB. Instead of outright rejecting a command, the model could engage in a dialogue by asking clarifying questions, such as, "*I see no books on the table. What do you mean?*". Furthermore, the model could suggest potential alternatives when the retrieval process yields low confidence. For example, if the input command is "*Find me my suns*" and the closest KB match is *sunglasses*, the model could

respond, "*Did you mean your sunglasses?*". Such interactions would transform the model from a passive interpreter to an active collaborator, enhancing its role in achieving shared goals. This concept will be explored further in the subsequent section.

4.2 Interacting in a collaborative environment

In collaborative environments, interaction is central to achieving shared goals between participants [168, 257], whether they are humans or autonomous agents. An environment, in its broadest sense, refers to the setting in which these interactions take place, encompassing not only the physical or virtual space but also the rules, constraints, and resources available [257, 243, 118, 188, 187]. In collaborative settings, multiple actors, typically a human and an intelligent system, work together, leveraging their respective strengths to reach a common objective. For this to be effective, both actors must actively contribute, sharing information and making decisions that benefit the overall success of the group [257].

In such scenarios, the interaction cannot rely solely on passive responses. Instead, both parties, especially the robotic actor, must have a clear understanding of the goal and the steps necessary to achieve it [243]. This requires constant communication and clarification. For example, if the human says "*Take the book on the table*", the robot must first identify the book, clarify any ambiguity if there are multiple books, and ensure it can grab it correctly as requested. If there are no books on tables, it should be able to ask more information through clarification questions: "*I don't see books on tables, what did you mean?*" [2, 256, 3]. This active participation, where the agent takes the initiative to ask clarifying questions, is key to ensuring that both parties are aligned in their efforts and that the task can be executed correctly.

Unlike traditional chatbot systems, which merely provide reactive answers, collaborative agents must be proactive, constantly assessing the current state of the environment and how it affects task completion [256]. They must dynamically engage in problem-solving by verifying that the commands issued by the user are feasible and, if not, prompt the user for further instructions or propose alternative solutions. For example, suppose the robot is required to move an object that is too heavy for its capabilities. In that case, it should be able to identify this aspect and promptly inform the user, such as "*This object is too heavy for me, what else can I do for you?*". Sometimes, this agents learn to solve problems through Reinforcement Learning [22] or Human Preferences [53], techinques very popular in the last decades. While it remains an interesting and fascinating problema, in this thesis we are more interested in the usage of Natural Language to interact with such agents, and so we will not go into more details on the Reinforcement Learning topic.

In order to understand the commands and check any inconsistency with the world, the agents need a way to receive such information: Natural Language. Usually, humans like to speak, verbally, but, for simplicity, suppose a text-based interaction, where the agent receives the text the humans write and can interpret it. For assessing the state of the environment, the simplest and intuitive way is to use cameras to capture pictures, in the same way we, humans, use our eyes to observe the world. Moreover, agents that operate in these environments must possess the ability to ground language in both the task at hand and the current state of the environment, utilizing information from both visual and textual data [132, 188, 187]. In doing so, they can better interpret and execute commands while maintaining a constant dialogue with the human participant, ensuring that progress towards the goal is smooth and efficient. In the end, this way of interacting resembles the way humans communicate, which we always seek even with robotic entities. As we move towards more complex, interactive environments, such as those found in robotic or virtual task-based scenarios [168, 256], this collaboration must be tightly integrated. Models designed for multi-modal interaction, such as those combining language and visual understanding [160, 284, 274, 105, 106], allow these agents to perform tasks with higher precision and a better understanding of the surroundings. They shift the focus from simple command execution to true collaboration, where human and robotic actors work together, sharing information and actively resolving issues as they arise. For this purpose, we will explore the interaction in a collaborative environment through Natural Language.

4.2.1 Modern Collaborative Interaction Platforms

In collaborative environments, natural language (NL) plays a pivotal role in enabling effective interaction between humans and robots. A successful collaborative agent must go beyond passively following instructions: it must actively engage in dialogue to interpret commands, resolve ambiguities, and provide meaningful feedback. The objective is to create a system where the robot not only understands NL commands but also integrates contextual information about its environment to make intelligent decisions. By leveraging these capabilities, the robot can dynamically adjust its actions and collaborate proactively with the human user. This interaction mirrors human-human communication and forms the basis for exploring platforms and datasets tailored to such collaborative scenarios.

While using a real robot for studying collaborative interaction may seem ideal, it is often impractical due to several challenges. Real robotic systems are expensive to acquire and maintain, require significant physical space for experimentation, and are subject to unpredictable real-world variables that complicate controlled testing. Additionally, iterating

on experiments with real robots is time-intensive and costly, making them less suitable for large-scale research. These limitations underscore the value of simulated environments, which provide a more accessible, scalable, and repeatable alternative for exploring natural language-driven collaboration.

Simulation platforms offer a practical and efficient alternative for studying collaborative interaction. They provide a controlled environment where experiments can be repeated under consistent conditions, allowing researchers to focus on developing and testing models without the unpredictability of the real world. These platforms are cost-effective, eliminating the need for expensive hardware and maintenance, and they enable faster iteration cycles by allowing immediate adjustments and testing. Although simulated environments may not fully replicate the complexities of the real world, they are particularly well-suited for tasks involving natural language interaction. By prioritising verbal exchanges between agents and users, researchers can explore the dynamics of collaboration in scenarios where physical actions are secondary to dialogue, making simulation platforms an invaluable tool for advancing collaborative interaction studies.

Several simulation platforms have been developed to support research in collaborative environments, each with distinct strengths and limitations. AI2Thor [135] provides photo-realistic indoor scenes and is particularly effective for interactive AI tasks, though it can be computationally demanding. iGibson [153] is well-suited for high-fidelity physics simulations, making it ideal for robotics applications, although its steep learning curve and hardware requirements may pose challenges. Habitat [232, 251] offers an efficient simulation framework for embodied AI research, focusing on navigation and exploration, though its interaction capabilities are limited. ThreeDWorld [86] excels in multi-agent interaction and realistic physics for both indoor and outdoor scenarios, enabling complex collaborative tasks, but it lacks comprehensive vision capabilities. Finally, SAPIEN [279] provides high-quality rendering and physics for object manipulation tasks, making it a strong contender for robotics research, though its emphasis on manipulation may limit its versatility for broader collaboration studies. These platforms offer diverse capabilities, allowing researchers to choose the one best aligned with their specific goals.

Interactive data is equally crucial for studying collaborative environments, as it captures the richness of human-like communication driving natural language interactions. This type of data allows researchers to examine how robots interpret ambiguous commands, ask clarifying questions, and adapt their responses based on user feedback. By prioritising human-like interaction over task-oriented dialogues, researchers can develop agents capable of engaging in meaningful and contextually aware conversations. While task-oriented datasets are useful for specific applications, they often lack the fluidity and realism of human-human commu-

nication. Interactive data bridges this gap, providing a foundation for systems that mimic natural dialogue while incorporating simple task-driven interactions.

Several datasets have been developed to support research in interactive and task-oriented dialogues, each offering unique advantages and limitations. Interactive Task Learning [151] focuses on enabling robots to acquire new tasks through natural dialogues with humans. While it provides valuable insights into task acquisition, its scope is primarily task-oriented rather than promoting rich, collaborative exchanges. Similarly, the Collaborative Dialogue in Minecraft [193] dataset offers structured interactions in a simulated world, allowing researchers to explore task execution through NL commands, but it is less focused on natural, free-flowing dialogue. The ConvAI3 ClariQ [5] dataset centres on generating clarifying questions for open-domain dialogue systems, but its artificial and narrowly focused approach limits its applicability to human-like communication scenarios. DialoGLUE [175], while a robust benchmark for task-oriented dialogues, lacks the realism needed to simulate collaborative human-robot interactions effectively. These datasets provide valuable resources for understanding specific aspects of dialogue systems but fall short in supporting broader studies on collaboration and ambiguity resolution.

Among the available options, the IGLU 2022 [133, 132, 187, 188] dataset stands out as an ideal choice for studying collaborative interaction. It features a Minecraft-like simulated world, offering a simple and flexible environment composed of coloured blocks where robots can process natural language commands and clarify ambiguities through dialogue. Unlike other datasets, IGLU emphasises the robot as an active collaborator, not merely a reactive agent. Another key advantage of IGLU is its inclusion of clarification questions, enabling the exploration of ambiguity resolution in natural language commands. Its recent launch at NeurIPS 2022 highlights its relevance and alignment with cutting-edge research. By focusing on collaboration within a dynamic and interactive simulated world, IGLU provides a comprehensive foundation for examining the nuances of natural language interaction, making it the most suitable platform for this thesis.

4.2.2 Interactive Grounded Language Understanding

The Interactive Grounded Language Understanding (IGLU) task stems from the broader goal of improving communication between humans and artificial agents, particularly in collaborative scenarios where natural language facilitates interaction. This task was a key focus of the NeurIPS 2022 IGLU challenge [133], a competition aimed at advancing the state of natural language understanding and interaction in human-robot collaboration. The challenge united researchers to develop agents capable of understanding, reasoning, and acting

on human commands within a controlled environment.

The IGLU task takes place in a simulated environment featuring two agents: the (human) Architect and the (robotic) Builder. Together, they collaborate to construct complex structures using coloured blocks. The Builder's objective is to interpret and execute the Architect's instructions to recreate a specified target design. Success in this task requires the AI agent not only to process language but also to engage in active communication when uncertainties arise. The interaction between the Architect and Builder is bi-directional. While the Architect provides construction commands, the Builder must evaluate whether the instructions can be executed within the given context or whether further clarification is necessary. This dynamic, interactive process of clarifying instructions is central to the complexity of the IGLU task. It mirrors real-world collaboration, where humans and robots must adapt to each other's responses to achieve shared goals effectively. A defining aspect of the IGLU task is its handling of role inversion. When the Builder encounters ambiguous or incomplete instructions, it must shift roles and ask clarifying questions, effectively becoming an active participant, while the human assumes the role of a "passive answerer". This role inversion demands that the Builder not only parse and comprehend the provided language but also generate contextually appropriate questions that elicit useful responses from the human collaborator.

For instance, consider the scenario depicted in Figure 4.4. In the upper part, the Architect issues a clear instruction, such as, "*At each end of the orange row, place one red block*". Since this command is unambiguous given the current state of the simulated *3D environment*, the Builder can execute it successfully, producing the expected result, as shown on the right. In contrast, the bottom part illustrates a more complex scenario. The Architect provides the command, "*Place two blue blocks on the leftmost red block*". Here, the phrase "*the leftmost red block*" is ambiguous because it could refer to different blocks depending on the Builder's orientation within the *3D environment*. To resolve this ambiguity, the Builder must ask a relevant question, such as "*Which direction is the leftmost facing: East or West?*". This example highlights the AI agent's essential capacity for adaptive interaction and deeper comprehension, moving beyond simple command execution to collaborative problem-solving.

The NeurIPS 2022 IGLU challenge was designed to benchmark and encourage advancements in Natural Language Processing, Machine Learning, and Human-Robot Interaction. Participating teams were tasked with developing AI agents capable of completing the IGLU task with high efficiency and accuracy. The structure of the challenge included several key aspects:

- **Task Diversity**: Commands in the challenge varied in complexity, ranging from straightforward, single-step instructions to multi-step, context-dependent directions.

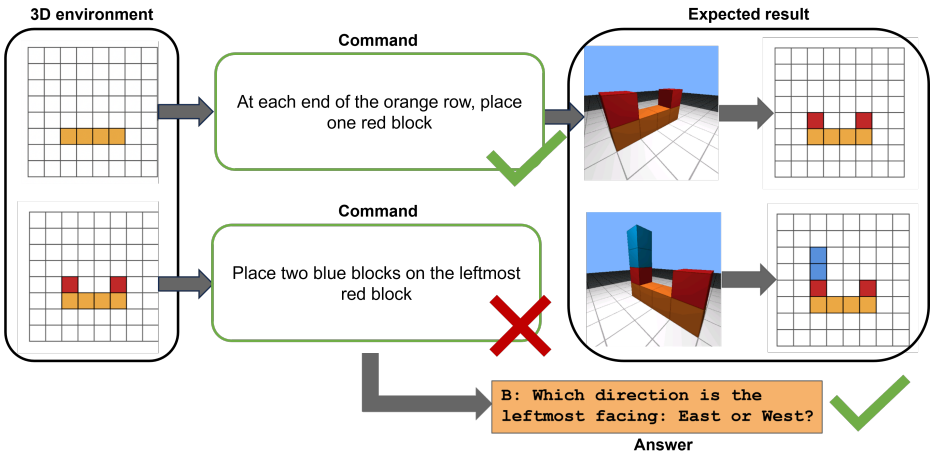


Figure 4.4: Taken from IGLU challenge description. *Top*: The architect's command was clear and no questions were needed, thus the Builder can execute it. *Bottom*: The word '*leftmost*' in the Command is ambiguous, so the Builder asks a clarifying question.

These commands tested the limits of both language understanding and strategic interaction. For instance, some commands required the Builder to infer information from the combination of the command and the current state of the environment. In such cases, clarification questions were unnecessary, as the Builder was expected to deduce the correct action and confirm its ability to execute the command. This exemplifies the demand for an AI agent to seamlessly integrate linguistic input with environmental knowledge.

- **Tasks and Evaluation Metrics**: The IGLU challenge comprised two primary tasks: *When to Ask* and *What to Ask*.
  - The *When to Ask* task assessed the ability to classify a command as executable or non-executable. This binary classification task was evaluated using the F1 score, which measured the capability of the model to determine whether sufficient information was available to proceed or whether further clarification was necessary.
  - For commands classified as non-executable, the *What to Ask* task evaluated the ability to produce the most relevant question. Initially, this task was framed as a retrieval problem, where the model ranked a closed set of candidate questions, selecting the highest-ranked option as its output. The Mean Reciprocal Rank (MRR) metric was used to evaluate the retrieval task, capturing the likelihood that the model identified a clarifying question that effectively advanced the interaction.

While these metrics offered a structured evaluation within the controlled challenge environment, they present limitations when applied to real-world scenarios. In practice, AI systems often generate responses rather than selecting from predefined sets, allowing for greater flexibility, linguistic diversity, and context-specific nuance. This generative approach enables AI agents to craft tailored responses, fostering natural and dynamic interactions. Thus, although the F1 and MRR metrics were effective for the IGLU challenge constraints, they do not fully encapsulate the complexity and fluidity of open-ended, real-world interactions.

- **Realistic Interaction Modeling**: The challenge emphasized the importance of simulating realistic human-agent interactions, requiring robust systems capable of managing linguistic ambiguities, incomplete information, and the unpredictability inherent in human communication. Additionally, some commands were intentionally ill-structured, comprising random yet thematically relevant words, to test the robustness of the models. For example, consider the command "*Facing north, blue and purple blocks will be arranged one by one*". While the command includes plausible vocabulary for the scenario, it lacks executable meaning, irrespective of the environment state. Such commands invariably necessitated a clarification question, challenging the AI system to identify and respond appropriately to ill-formed inputs.

One of the primary challenges in IGLU is that language, as used by humans, is inherently flexible and context-sensitive. Commands such as "*Place a block here*" or "*Make the left side higher*" are often vague unless supplemented with contextual knowledge or additional input. This ambiguity requires an AI system to not only parse the language but also infer meaning based on the current environment. The need for models to manage this complexity highlighted several **research questions**:

- How can AI agents dynamically decide **when required information is missing** (and thus to seek clarification) versus when to proceed based on partial information?
- What strategies can be used to train models to **generate contextually relevant questions** that mimic human-like requests?
- How can we evaluate **the quality of the generated answers**, given that free-text responses pose challenges for comparison with the ground truth?
- How can multi-modal input (e.g., visual and textual data) be leveraged to **improve decision-making** in collaborative environments?

- Can we extend 'one-turn-interaction' to a **dialogue**, enabling the model to ask multiple questions to retrieve all relevant information for a complete request?

These questions set the stage for the development of more sophisticated models capable of engaging in interactive dialogue, with performance measured not just by task completion but also by the quality of interaction. In the following sections, we will explore the IGLU challenge and address these research questions where possible.

An End-to-end Transformer-based Model for IGLU. As an initial solution, the BART-IGLU [117] model was developed to address the IGLU task, leveraging the foundational capabilities of the Bidirectional and Auto-Regressive Transformers (BART [152]) architecture. Known for its robust language generation and comprehension abilities, BART operates as a sequence-to-sequence model pre-trained as a denoising autoencoder. This pre-training enables the model to reconstruct the original form of an input sentence that has been corrupted, making it particularly well-suited for tasks requiring both understanding and generating coherent language. For the IGLU task, the BART model was adapted to process instructions, determine executability, and generate clarifying dialogue as necessary.

Typically, participants in this challenge model the process using an initial classification step to determine whether a given command can be executed within the current environment. This is followed by a retrieval process to select the most suitable clarifying question when the command is ambiguous [133]. Notably, none of the approaches employed in the challenge are fully end-to-end systems, as they do not directly produce affirmative answers or seek assistance when presented with a question as input. Instead, they rely on a two-step process.

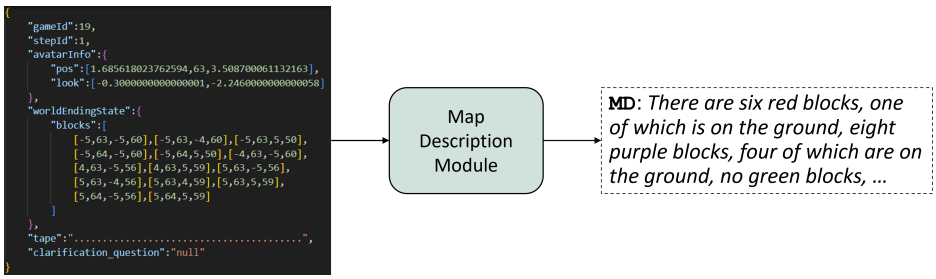


Figure 4.5: The flow for converting the 3 dimensional information of the world in natural language.

Central to the adaptation of BART for the IGLU environment was the implementation of a *Textification* strategy, which mirrors the approach used in the GrUT [108] model, as discussed in the previous section. *Textification* involves converting environmental observations into structured textual representations that the model can interpret, transforming environmental information into text to leverage the language modeling capabilities of BART. This

step was essential because BART, in its original form, is designed to handle text-only input. We adopted a similar approach by combining textual descriptions of maps with natural language commands. To construct a textual description of a map, we provide information about the blocks, their positions, and their colours. This process is illustrated in Figure 4.5, which references the description of the environment in Figure 4.6. This structured description forms the foundation of this approach and provides context for the BART model. Additionally, details such as the colours of the blocks on the ground can be utilized in ambiguous situations. Subsequently, we combine this textual Map Description (MD) with a natural language command, such as "*Break the green blocks.*". Together, these components form the input for the BART-based system. The model interprets the textual map description and the command, generating a corresponding question when more information is needed. For example, the output question might be: "*There are no green blocks, which blocks should I break?*". A critical aspect to emphasize is the context-awareness of BART through the MD, which allows it to generate questions about missing information or ambiguities in the given command. As a result, the MD functions as both a descriptor of the Minecraft-like environment and a surrogate for visual information.

The application is thus straightforward: BART takes as input the MD and the natural language command, producing either an affirmative answer such as "*I can execute it*" when the command is clear and coherent with the environment or a clarification question in free text form when ambiguities or inconsistencies arise.

Experimental Setup and Evaluation. In this section, we assess the proficiency of the trained model in generating contextually grounded questions, which serves as evidence of its ability to comprehend instructions and identify missing information that can be reformulated into a question. This evaluation provides valuable insights into the cognitive understanding of the model and its competence in generating contextually relevant questions. First, we scrutinize the quality of the answers produced by BART. Then, we explore whether an end-to-end system can successfully generate valid responses. It is important to note that while two sentences, denoted as A and A', may exhibit variations in their BLEU scores, they can still be semantically coherent with respect to the task. Therefore, we conduct a thorough manual analysis of the generated questions. Since the answers of the model are in free text form, evaluating them presents certain challenges [6]. Inspired by works such as [144], we introduce two new human-judged evaluation metrics: Utility and Fluency.

The model is based on BART-base, implemented using the Huggingface framework[8]. The training process involves providing as input the concatenation of the environment description and the user utterance, while the output is the expected response from the robot. This

[8] https://huggingface.co/facebook/bart-base

response is the artificial string "*I can execute it.*" if no additional information is required or the appropriate clarifying question if more information is needed. The hyperparameters are detailed in Table D.1 in the Appendix. For the execution of commands in the environment, other works, such as [126], have explored this topic; however, it is beyond the scope of this work.

We evaluated the quality of the generated text by measuring the BLEU score between the correct and generated sentences. The results are as follows: BLEU1 = 0.255, BLEU2 = 0.147, BLEU3 = 0.083, and BLEU4 = 0.061. It is evident that as the expected n-grams increase, the BLEU score decreases, with BLEU4 reaching a low value of 0.061.

However, BLEU, originally designed for tasks like machine translation, can be overly restrictive. For instance, in response to a command such as "*Destroy all the red blocks*", a system answering "*The map contains no red blocks*" may share no common terms with a response like "*I don't see any elements of the requested color*", resulting in a BLEU score of 0.

To address these limitations, we conducted a qualitative assessment using a test set comprising 683 examples. From this set, we selected a subset of 47 examples where the system generated a request. We manually examined whether the generated sentences, although differing from the expected ones, contained questions that were useful in resolving the ambiguity or limitations of the user's request. Generated sentences meeting this criterion were considered correct; otherwise, they were considered incorrect. This allowed us to compute a Relaxed-Accuracy, defined as the percentage of examples manually considered correct.

Additionally, for the error analysis, we identified eight categories of "missing" information in the commands that the gold-standard (GS) annotated question addresses. These categories include aspects such as the NUMBER or COLOR of blocks to be placed or removed, the DIRECTION in which a line of blocks must be placed, or BLOCK MISSING, where the command refers to a specific block that does not exist in the environment. Table 4.10 describes all the identified categories, along with an example question for each.

As shown in Table 4.10, the reported Relaxed-Accuracy is quite low in most categories, with DIRECTION achieving 22.23%. This is primarily due to two phenomena: *i)* the majority of the commands in the test set are complete, and the BART system presented in this work correctly generates the sentence "*I can execute it.*", indicating that no additional questions are needed (the COMPLETE category); *ii)* BART fails to generate the exact question annotated by the Gold Standard. Despite low performance in certain specific categories, the OVERALL Relaxed-Accuracy reaches 92.54%, which is a noteworthy result given the complexity of the task and the limitations of the dataset. A detailed error analysis of this model is provided in Appendix E.

Category	Description with example	Relaxed-Acc
Block	"*Which specific block do you mean ?*"	38.46%
Vertical-Horizontal	"*How are they arranged? Vertical or horizontal?*"	50.00%
Number	"*How many blocks? Or how long?*"	57.14%
Square	"*Where should I place the blocks?*"	77.14%
Color	"*Which color should the block be?*"	50.00%
Direction	"*In which direction? What is the orientation?*"	22.23%
Block Missing	"*There is no red block*"	58.34%
Complete	"*I can execute it.*"	97.81%
Overall	-	92.54%

Table 4.10: The categories of "missing" information in the command identified in this work. Each category is described by a question example. A "Relaxed" Accuracy is computed for each category on the test set.

As previously discussed, exact match comparisons between two sentences often lack the informativeness needed to fully assess the capabilities of Generative Models. To address this, we introduced two additional metrics involving human judgment: individuals were asked to determine which response they preferred and which exhibited superior English syntax and semantics. This assessment offers a holistic perspective on both the linguistic quality and the contextual accuracy of the generated questions.

In the evaluation process, we assessed the test dataset using two key metrics: *Utility* and *Fluency*. These metrics were instrumental in evaluating the performance of the BART model. *Utility*, the first metric, evaluates the ability to generate a coherent question that directly addresses the missing information in the given command. In essence, it measures the proficiency in generating relevant questions to elicit the required information. *Fluency*, the second metric, evaluates the grammatical correctness and natural flow of the generated questions in English, irrespective of the correctness of the response itself. Table 4.11 provides the scores for these two metrics, along with brief descriptions of each score.

Score	Utility	Fluency
1	*Completely inconsistent*	*Not English*
2	*Incorrect question*	*Random English words*
3	*Awareness of the task, but the question is not relevant*	*Understandable but critical errors*
4	*Asks at least one missing information (color, blocks, etc.)*	*Light grammatical errors*
5	*Perfect*	*Perfect*

Table 4.11: Scores for the Utility and Fluency metrics from 1 to 5, where both need to be maximized.

To ensure objectivity and impartiality in this evaluation process, we enlisted the assis-

tance of an evaluator who was not part of the development team. To avoid any bias, it was essential that this external evaluator remained unaware of whether the sentences they were assessing originated from the Gold Standard (GS) or BART. This unbiased assessment was conducted on the test dataset comprising 230 sentences, where the question provided by the GS differed from the question generated by BART. These examples were evenly split between BART and the GS and were paired with the visual representation of the environment (as shown in Figure 4.6). This process enabled us to derive meaningful insights into the semantic and syntactic capabilities of the model, highlighting both its strengths and areas for improvement.

The Gold Standard annotation achieves a Utility score of 3.10, indicating that the annotated data of the IGLU competition is not consistently accurate, occasionally lacking comprehensive information and sometimes posing misleading questions. In contrast, the BART model achieves a higher Utility score of 3.95, reflecting its ability to generate more relevant questions that address critical missing information in the command, despite occasional inaccuracies. Regarding Fluency scores, both models perform exceptionally well, with no significant disparities observed: the Gold Standard annotation attains a score of 4.84, while the BART model achieves a slightly higher score of 4.97.

The strengths of BART-IGLU lay in its effective text processing and adaptability for HRI. Its ability to maintain high fluency throughout interactions made it an appealing tool for scenarios requiring seamless communication. However, the reliance on the textification introduced certain limitations. Since visual information had to be translated into text, the model often missed the nuances of spatial relationships that could be better captured through direct visual input. This gap became particularly evident in tasks that involved complex spatial reasoning or multi-step commands, where text alone could not fully convey the necessary context.

The shortcomings of BART-IGLU underscored the need for an enhanced approach that could integrate multi-modal data, leading to the development of the MM-IGLU model. This transition aimed to overcome the limitations of text-only input by generating the visual representation of the environments and subsequently incorporating such information along with the text of the command, offering a richer context for understanding and interaction.

4.3 Multi-Modal Interaction in HRI

The approach of the BART-IGLU model to interpreting environmental context was constrained by its text-only input. Without direct access to visual cues, the model often generated redundant or suboptimal questions, particularly when handling commands involving

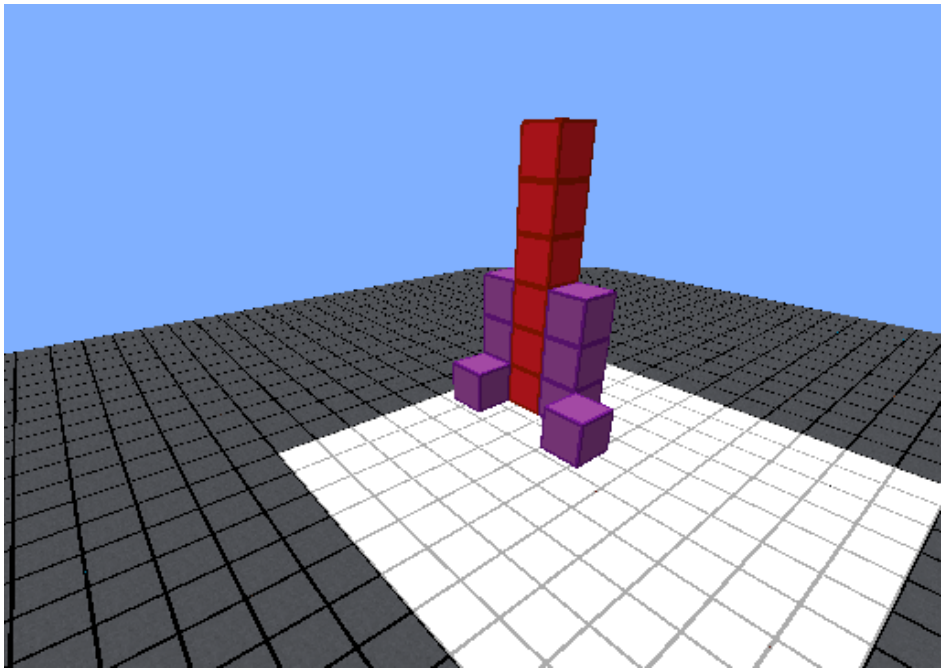


Figure 4.6: An example of visual rendering of the environment, where the Instruction given by the Human is "*Break the green blocks*" and the expected answer is "*There are no green blocks, which blocks should I break?*".

spatial relationships or specific positioning that could be inferred by visually observing the environment. Relying solely on text descriptions increased the potential for misinterpretation and unnecessary clarification questions. To address these limitations, the resource *Multi-Modal IGLU* (`MM-IGLU`) [118] was developed by generating images of each environment coupled with the commands from the original challenge. Furthermore, a multi-modal architecture was introduced to integrate visual data directly into the interaction process. This enhancement enabled the model to interpret spatial relationships and environmental cues more accurately, reducing ambiguity and improving the quality of interactions.

One of the critical steps in transitioning to `MM-IGLU` was the expansion of the IGLU dataset to include visual representations of the environments associated with each instruction. Originally, the dataset comprised only textual commands and JSON-like environment descriptions, which limited the ability to understand the spatial aspects of the task. To enable `MM-IGLU` to utilize visual information, 3D images of various construction environments were generated, capturing the layouts and configurations relevant to each task. This refined dataset now offers not only textual information but also multi-modal data, encompassing both images depicting the world for each command and a meticulously constructed textified environment description from the previous Section. These descriptions ensure a controlled syntax, free from hallucinations, thereby providing a more robust dataset for a

comprehensive range of methodologies.

The architecture of the `MM-IGLU` model was based on the Large Language and Vision Assistant (LLaVA) [160] framework, which provided to process and understand complex instructions that rely on spatial and visual cues in addition to language. By combining a vision encoder with a language model, LLaVA provides a structure that allows `MM-IGLU` to generate responses grounded in both textual and visual contexts, which is critical for the nuanced interactions required in the IGLU task.

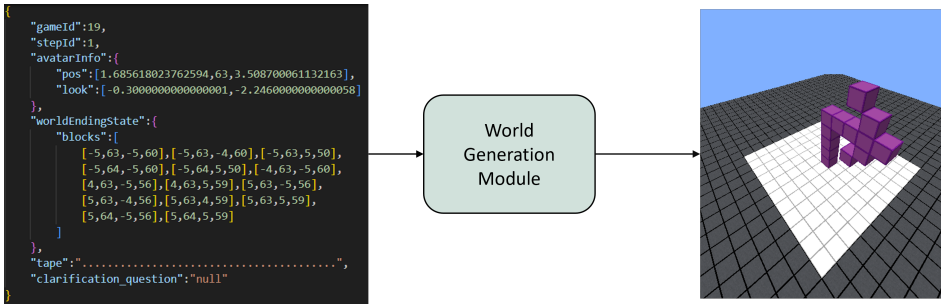


Figure 4.7: The image generation process, where each JSON-like environment description corresponds to a 3D colored image.

4.3.1 Building the MM-IGLU Resource

To enrich the IGLU dataset with multi-modal evidence, we introduced two additional dimensions for each command-question pair: *i*) images depicting the block configurations in the environment, and *ii*) environment descriptions in natural language.

Images are generated by extending the `gridworld` tool provided in the competition to initialize an empty environment from the JSON-like description (Figure 4.7) provided by IGLU and automatically placing one block at a time in the correct position. A single viewpoint was carefully selected to represent the agent's perspective within the simulated world, positioning it at a slightly elevated angle from the ground to maximize the visibility of the blocks present, as illustrated in Figure 4.6. All images were generated at a resolution of 256×256 pixels. The description process corresponds to Figure 4.5, as discussed in the previous section on BART-IGLU.

We provide a new split of the overall dataset into Training, Validation, and Testing sets (the latter not available during or after the competition), as detailed in Table 4.12. Out of the $5,530$ training commands, 717 (12.97%) were annotated as "Ambiguous" (i.e., they require at least one question to advance in the task), while $4,813$ (87.03%) were considered "Clear" instructions (i.e., executable without further clarification). The average length of clarifying questions is approximately 12 words, indicating that these questions tend to be

Section	Instructions			Avg Len	
	#Exs	#Clear	#Amb	C	Q
Train	5,530	4,813	717	18.60	12.25
Val	615	531	84	17.42	11.46
Test	683	594	89	18.76	11.69

Table 4.12: Statistics of the datasets for total examples ("#Exs"), clear commands ("#Clear"), ambiguous commands ("#Amb"), and average word length for commands ("C") and questions ("Q").

quite specific. The same trend is observed in the Validation set with 615 total commands and the Testing set with 683 total commands. Additional details and analysis about the resource are provided in Appendix D.

4.3.2 LLMs for Multi-modal IGLU

We utilize advanced Transformer-based models and Large Language Models (LLMs), specifically tailored for sequence-to-sequence tasks. Whether processing an image or textual description, the objective of the model remains consistent: to take an input (either text or text paired with an image) and produce a precise output in natural language. Within the IGLU framework, two tasks are integrated: *classification* and *generation*. The classification task prompts the model to decide whether the given command is executable based on the provided context. If executable, the model confirms with a "*Yes*"; otherwise, it signals a "*No*". The generation task is employed when the command lacks clarity, requiring the model to formulate a pertinent question to gain the necessary information. Inference for these tasks can follow two approaches. The first is a *two-step method*, where the model initially determines the executability of a command and, if incomplete, subsequently generates a clarifying question. The alternative is a *monolithic* strategy, where the model either confirms "*I can execute it*" or directly poses the required question.

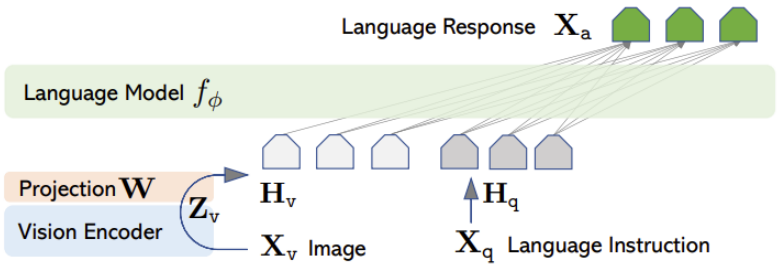


Figure 4.8: The LLaVA network architecture, taken from [160]

We adopted the Multi-modal approach based on the Large Language and Vision Assis-

tant framework (LLaVA) introduced by [160]. LLaVA integrates foundational visual models with linguistic models, employing a single-layer neural network, known as the Projector, to align the output representation from the visual model with the input representation from the language model. The architecture of LLaVA is depicted in Figure 4.8. In this figure, X_v and X_q represent the image and input text, respectively, while H_v and H_q denote their embedding representations, aligned with the language model. The input text X_q undergoes tokenization, while the image X_v is processed through the Vision Encoder and the Projection layer W to align it with the Language Model vector space. This alignment ensures effective communication between the vision and language components, enabling it to leverage both modalities. The LLaVA architecture in `MM-IGLU` begins by processing visual data through the vision encoder. This encoder, based on convolutional neural networks (specifically CLIP [215]), extracts essential features from the images, such as shapes, colours, spatial arrangements, and object relationships, into an embedding vector. These features are then passed through a projection layer, transforming the vector into a space interpretable by the language model. This transformation ensures compatibility between visual information and the text-based components of the model. Concurrently, the textual instruction provided by the Architect is fed into the language model. The language model, typically a large Transformer-based architecture, interprets the instruction in light of the current task context. When the projected visual features and textual representations converge in the language model, they produce a response reflecting both the immediate visual environment and the intent of the command. This alignment allows `MM-IGLU` to generate context-aware responses grounded in both modalities.

The model is fine-tuned[9] by taking as input the tuple:

$$\langle \texttt{Introduction}, \texttt{Prompt}, \texttt{Image}, \texttt{Command} \rangle$$

The `Introduction` provides contextual background for the task and reads:

"*In this virtual world reminiscent of Minecraft, you are a robotic entity equipped with the ability to move freely, place or remove blocks within the environment. Imagine you are situated in the environment depicted in the image provided. Your task is to determine whether you can execute a given command based on the current configuration of the world. If you require additional information to carry out the command effectively, you should respond by asking relevant clarifying questions, such as inquiring about block colors, quantities, directions, or any other necessary details.*"

[9]Initially, this model was tested in a zero-shot manner, but it resulted in unstable outcomes, often leading to hallucinated answers. While most sentences generated were sensible, they typically failed to demonstrate an understanding of the need to perform actions within the environment, often miscounting blocks.

The `Prompt` specifies the subtask. For the classification task, it states: "*Respond with 'Yes' if you can execute the command, or 'No' if additional information is required.*" For the generation task, the prompt is: "*Answer with 'I can execute it' if the command is executable, or pose a pertinent clarifying question if further details are needed.*".

The `Image` token serves as a placeholder, replaced by X_v through the vision encoder, while the `Command` represents the input natural language command. Thus, X_q is the concatenation of `Introduction`, `Prompt`, and `Command`. The output X_a of the model follows a "*Yes/No*" structure for classification tasks or generates a direct question or affirmative response, "*I can execute it*", for generation tasks.

Inspired by the approach of ExtremITA [109], which demonstrated effective data fusion from multiple tasks to guide LLM prompting, we introduced the capability for multi-modal models to train a single LLaVA model by combining data from both classification and generation task prompts. This multi-task learning approach is promising, as it allows tasks to mutually benefit each other. Specifically, the generation task may improve as the model implicitly specializes in the classification task. From a practical perspective, this is achieved by merging training datasets generated from both modalities with task-specific instructions.

4.3.3 Experimental Setup and Evaluation

In this section, we assess the proficiency of the Multi-Modal model in generating contextually grounded questions, which serves as evidence of its ability to comprehend instructions and identify missing information that can be reformulated into a question, similarly to BART-IGLU. Additionally, we apply the same human-judged evaluation metrics: Utility and Fluency.

In the LLaVA architecture, two essential components must be selected: the LLM and the Vision Module for encoding images. Following initial experimentation, we adopted the combination of CLIP [215] for the Vision Module and LLaMA2 [261] for the LLM, as they produced superior results on the development set. While other contenders such as Vicuna [289] and Guanaco [69] were considered, their results are omitted as they were considerably low. LLaMA2, implemented using the HuggingFace framework, is available in different sizes ($7b$ vs $13b$) and variations (`base` or `chat` versions). The Vision Module, based on CLIP, remained frozen throughout all experiments[10]. The projector, a single-layer Feedforward Neural Network, was initially based on LLaVA's release but later re-tuned from scratch,

[10]The fine-tuning process adhered to the one described in [160]. Their rationale was that CLIP performed well on their images without further tuning. Although the images are in a "Minecraft-style", zero-shot descriptions requested from the model were accurate, indicating understanding to some extent and that no additional fine-tuning of CLIP was necessary.

resulting in slightly improved convergence. The hyperparameters were optimized on the development set and are summarized in Table D.1 in the Appendix.

Evaluating the Question Generation Process. We present the results from various Language Models applied to Classification (deciding whether to ask or not) and Generation (determining what to ask) tasks. All models fine-tuned using the LLAVA framework employ the same CLIP visual encoder, which remained "frozen" during fine-tuning. Evaluation was conducted based on Macro-F1 scores, treating the phrase "*I can execute it.*" as the positive response to the command "*Can you execute this command?*". For Classification (as shown in Table 4.13), the Text-Only BART-IGLU model, reliant on the command and environment description, achieved a Macro-F1 score of 74.36%. This performance was primarily hindered when determining whether to pose questions (negative F1). The model, initially designed for question generation as detailed in the previous section, exhibited classification as a side effect of its training. Fine-tuning BART specifically for classification and multitask modes yielded similar results to its generation mode. Notably, the model equates "*Yes*" with "*I can execute it.*", highlighting limited multitasking generalization. A direct comparison with the systems participating in the IGLU competition is infeasible due to the unavailability of test data. However, the top three competition systems scored Macro-F1 scores of 76.6%, 76.1%, and 75.4% [133]. Despite differences in the test sets, this local test results are comparable to the top-performing systems, emphasizing the end-to-end nature of these models.

Model name	**Type**	**Tr.task**	**F1 Pos**	**F1 Neg**	**M-F1**
BART-IGLU	TO	Gen	93.76%	54.55%	74.36%
LLaMA2-7b	MM	Class	96.04%	62.05%	79.05%
LLaMA2Chat-7b	MM	Class	96.42%	67.16%	81.79%
LLaMA2-13b	MM	Class	96.19%	64.12%	80.16%
LLaMA2Chat-13b	MM	Class	**96.43%**	**67.16%**	**81.80%**
LLaMA2Chat-13b	MM	MT	96.35%	66.17%	81.26%

Table 4.13: The Classification performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two. The Type TO stands for Textual-Only and MM stands for Multi-Modal. MT here stands for Multi-Task training, i.e. the union of the Classification dataset and the Generation one, using ad hoc instructions.

In Multi-Modal (MM) solutions using LLaMA2 checkpoints from the LLAVA framework, initial results were suboptimal, likely due to LLaVA's training on real images rather than Minecraft-style environments. Subsequently, we focused on fine-tuning Meta-LLaMA2 models for command Classification. These models exhibited improved performance, especially the Chat variants, which were pre-trained on instruction data. The 7*b* variant improved from

79.05% to 81.79%, while the 13*b* variant rose from 80.16% to 81.80%. Interestingly, no significant difference was observed between the two sizes. We further trained the LLaMA2Chat-13b version using Multi-Task training, alternating between Classification and Generation tasks. While this approach resulted in a slight decline in Classification performance, reaching 81.26% Macro-F1, it later demonstrated improved performance in the Generation task, confirming the initial hypothesis of mutual benefits in multi-task learning [109].

Model name	**Type**	**Tr. task**	**F1 Pos**	**F1 Neg**	**M-F1**
BART-IGLU	TO	Gen	93.76%	7.14%	50.45%
LLaMA2Chat-13b	MM	Gen	93.90%	45.26%	69.58%
LLaMA2Chat-13b	MM	MT	**93.95%**	**47.89%**	**70.92%**

Table 4.14: The Generation performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two. The Type TO stands for Textual-Only and MM stands for Multi-Modal. MT here stands for Multi-Task training, i.e. the union of the Classification dataset and the Generation one, using ad hoc instructions.

4.3.4 Utility and Fluency Evaluation

We evaluated the Utility and Fluency aspects of the generated answers once again. As usual, we enlisted two external annotators unfamiliar with the specifics of the project. They were provided with both system-generated examples and gold standard examples requiring clarifications, without any indication of their source. This approach was designed to minimize bias. Each annotator was instructed to rate the clarifications along two dimensions. First, they assessed *Utility*, assigning a score between 1 and 3 based on the guidelines in the second column of Table 4.15. This measure captured the effectiveness of the clarification concerning the task in a nuanced manner. Second, they evaluated *Fluency*, assigning a score between 1 and 3 based on the criteria in the third column of Table 4.15, which focused on the quality of the English writing, including grammatical and syntactical correctness. Compared to BART-IGLU, the scoring range was reduced to 1-3 to mitigate excessive variability observed in previous results.

The results are presented in Table 4.16. The MM-model achieved the highest Utility score of 2.73 (out of 3), reflecting its ability to generate relevant questions and address important missing information, though occasional inaccuracies persist. In terms of Fluency scores, all models performed exceptionally well: 2.91 for the Gold Standard annotation, 2.98 for the BART-IGLU model, and 2.99 for the MM-model. The Pearson correlation between the two annotators was 0.64 for Utility and 0.81 for Fluency. The results indicate that the model

Score	Utility	Fluency
1	*Incorrect classification*	*Not English or random English words*
2	*The clarification suggests awareness of the task but misses some key aspects*	*English with grammatical errors*
3	*Perfect*	*Perfect*

Table 4.15: Scores for the Utility and Fluency metrics from 1 to 3, where both need to be maximized.

generates simple, effective, and linguistically correct sentences, leveraging the strengths of the LLM, and appears even more useful than the clarifications suggested by the original annotators.

Dataset	Utility	Fluency
Gold standard	2.16	2.91
BART-IGLU	2.37	2.98
MM-model	2.73	2.99

Table 4.16: Utility and Fluency results for the Gold Standard (GS), the linguistic-only BART-IGLU, and the Multi-Modal model (MM-model).

The integration of vision and language in `MM-IGLU`, particularly through the use of a recent and larger LLMs based on LLaMA, allows the model to generate responses that are more diverse and linguistically nuanced. Traditional retrieval-based systems are constrained to selecting responses from a fixed set, reducing flexibility. However, with the generative capabilities provided by the LLaVA architecture, `MM-IGLU` has the freedom to craft responses that reflect subtle variations in task context, enabling more dynamic and natural interactions. This approach allows `MM-IGLU` to extend beyond the constraints of the original challenge, paving the way for future advancements in tasks and datasets.

The inclusion of visual data provided the Builder with direct visual context, allowing it to infer information that would otherwise require verbal clarification. For example, in tasks where block arrangements or orientations were unique, the Builder could directly observe the necessary configurations from the image, avoiding redundant queries. This dataset expansion offered a richer and more realistic context for training and evaluating the model, enabling a more efficient interaction process. The integration of visual data fundamentally shifted the perspective on how this task can be approached.

While English remains a resource-rich language with extensive experimentation, Italian is significantly less represented. Testing such an architecture on Italian is both a compelling challenge and an area of potential development. This exploration is addressed in the next section.

4.3.5 MM-IGLU-IT: Expanding Multi-Modal IGLU for Italian

Following the development of MM-IGLU for English-language interaction, the limitations of language exclusivity became increasingly evident, particularly for Human-Robot Interaction (HRI) in diverse linguistic contexts. While MM-IGLU enabled the model to handle multi-modal instructions with spatial and linguistic grounding, it lacked adaptability to Italian, a language underrepresented in large language models. This gap posed a significant challenge in achieving inclusive and linguistically accurate interaction capabilities, as Italian language nuances introduce complexities not fully addressed by existing English-trained systems, as demonstrated in [115]. These differences became more pronounced, highlighting the need for an Italian-specific dataset that could support nuanced clarifications and context-aware dialogue generation without relying solely on translated English data. The goal of `MM-IGLU-IT` [37] was therefore to create a dedicated multi-modal Italian dataset to address these challenges, enabling the model to engage naturally and accurately in Italian. By developing a resource tailored to the Italian language structure and usage, `MM-IGLU-IT` enhances the ability to generate linguistically appropriate responses, perform grounded interactions, and achieve high utility and fluency in Italian-language HRI tasks.

Inspired by the approach of [65], which translated the Visual Question Answering dataset into Italian, we utilized DeepL for the initial translation[11]. However, unlike [65], which validated only the test set, this process involved manual validation of the entire dataset, including training, validation, and test sets, by two annotators. This comprehensive validation ensured the integrity of the data, reducing the exposure of models to synthetic data and enhancing the training process. Before completing validation, we evaluated the translation quality using BLEU [206] scores on the test set for both user commands and clarification questions, compared with the manual validation. The results are presented in Table 4.17. High BLEU-1 scores indicate good overall translation quality, but the decline in scores with higher n-grams highlighted the need for further corrections. This prompted a manual validation of all translations to ensure accuracy. Consequently, `MM-IGLU-IT` encompasses the same $6,800$ examples, as detailed in Table 4.12. Each example includes an image depicting the arrangement and colors of blocks in the environment, accompanied by a command. If

[11]Source accessed in March 2024 at https://www.deepl.com/it/translator.

-	**Commands**	**Questions**
BLEU-1	0.88	0.95
BLEU-2	0.83	0.92
BLEU-3	0.78	0.48
BLEU-4	0.73	0.39

Table 4.17: The different values of BLEU score computed between the automatic translation in Italian and the manually validated version, both for the input commands and the output clarification questions, on the test set only.

the command is not executable given the configuration, an expected clarification statement from the robot is included. Similar to IGLU and MM-IGLU, 13% of the examples require clarification.

The statistics of this dataset, e.g., the number of commands, questions, ambiguities, and so on, are perfectly aligned with the MM-IGLU dataset from the previous section. The main contribution here is the translation and validation in Italian. Furthermore, each Italian example is aligned with the original image and the corresponding command in English, supporting future cross-lingual research. Additional details about the validation process can be found in Appendix D.2.

Experimental Setup and Evaluation. In this setup, the model is based on LLaVA [160] and is fine-tuned[12] by taking as input the tuple:

$$\langle \texttt{Introduction}, \texttt{Prompt}, \texttt{Image}, \texttt{Command} \rangle$$

This experimental setup mirrors the English counterpart, with the key distinction that all textual elements are in Italian: the `Introduction`, `Prompt`, and `Command`.

Once again, we evaluated the Classification and Generation capabilities of the Italian models in solving the task. In the classification modality, the agent determines whether a command can be executed, responding with either "*Yes*" or "*No*". In the generation task, the agent generates a textual response, either affirming the executability of the issued command or producing a clarification question if it is ambiguous. The same LLaMA2 Chat-13b[13] language model is used, as Italian is included in its pre-training stage, albeit minimally (see Appendix A for further details).

Given the multilingual nature of the LLaMA model, we tested different language com-

[12]Initially, this model was tested in a zero-shot manner, but it resulted in unstable outcomes, often leading to hallucinated answers. While most sentences generated were sensible, they typically failed to demonstrate an understanding of the need to perform actions within the environment, often miscounting blocks.

[13]https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

binations between English and Italian. Specifically, we compared the multi-modal model introduced in the previous section, trained on the English dataset (`LLaMA2Chat-13b-EN`), with the model trained on the Italian dataset (`LLaMA2Chat-13b-IT`). This comparison assesses the impact of language-specific training on model performance. Following [118], the linear projector, initially derived from LLaVA's release, was later completely re-tuned to achieve slight improvements in convergence, with all models using the same frozen version of CLIP. Hyperparameters remained unchanged, as detailed in the previous Section.

Model name	**Tr. Lan**	**Test Lan**	**F1 Pos**	**F1 Neg**	**M-F1**
`LLaMA2Chat-13b-EN`	EN	EN	96.43%	67.16%	81.80%
`LLaMA2Chat-13b-EN`	EN	IT	70.07%	24.29%	47.18%
`LLaMA2Chat-13b-IT`	IT	IT	97.81%	66.67%	82.24%

Table 4.18: The Classification performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two. The evaluation is divided into the Language of Training (Tr. Lan) and the Language of Testing (Test Lan).

Observing the results in Table 4.18, the `LLaMA2Chat-13b-EN` model serves as the reference point. It achieves an F1 Positive score of 96.43% when asserting its ability to recognize commands consistent with the environment. However, performance declines for ambiguous commands requiring clarification, resulting in an overall Macro-F1 of 81.80%. This is plausible given the dataset imbalance, with only 13% of cases requiring a "*No*". Testing this model on Italian data without further tuning led to a significant drop in performance (Macro-F1 47.18%), particularly in identifying ambiguous commands (F1 Neg 24.29%). Despite this, the model occasionally responded correctly, likely due to similarities between English and Italian terms (e.g., "*destroy*"/"*distruggere*") and the shared vision model. A more accurate evaluation was conducted by assessing the `LLaMA2Chat-13b-IT` model, trained on Italian-translated data, on the Italian test set. This model achieved comparable performance to its English counterpart in correctly identifying executable or ambiguous commands, with a slightly higher overall Macro-F1. Notably, its ability to identify ambiguous commands in Italian doubled compared to the English-trained model (F1 Neg 66.67% vs. 24.29%).

Finally, we evaluated the models in a generation setup where the task is to produce a complete phrase, either affirming the command or generating a clarification for ambiguous commands. Results are presented in Table 4.19. The `LLaMA2Chat-13b-EN` model, trained and tested in English, achieved an F1 Positive score of 93.95%, an F1 Negative score of 47.89%, and a Macro-F1 of 70.92%. When tested on Italian data without further tuning, performance dropped significantly (F1 Positive 70.01%, F1 Negative 0.00%, Macro-F1

Model name	Tr. Lan	Test Lan	F1 Pos	F1 Neg	M-F1
`LLaMA2Chat-13b-EN`	EN	EN	93.95%	47.89%	70.92%
`LLaMA2Chat-13b-EN`	EN	IT	70.01%	0.00%	35.00%
`LLaMA2Chat-13b-IT`	IT	IT	93.62%	44.16%	68.89%

Table 4.19: The Generation performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two.

35.00%), highlighting its struggles with ambiguous commands in Italian. In contrast, the `LLaMA2Chat-13b-IT` model, fine-tuned on the Italian dataset, performed comparably to the English baseline on Italian test data, with an F1 Positive score of 93.62%, an F1 Negative score of 44.16%, and a Macro-F1 of 68.89%. These results suggest that a model trained solely on Yes/No responses is more effective for this specific task when a command is indeed executable. As in [118], a multi-step approach remains advantageous, i.e., first identifying ambiguity and then generating clarifications.

Dataset	Language	Utility	Fluency
Gold standard	EN	2.16	2.91
`LLaMA2Chat-13b-EN`	EN	2.73	2.99
Gold standard	IT	2.69	2.98
`LLaMA2chat-13b-IT`	IT	2.79	2.99

Table 4.20: Utility and Fluency results for the Gold Standard and the Multi-Modal model (LLaMA2chat-13b-IT).

Lastly, we evaluated Utility and Fluency for the Italian model. Two external annotators assessed shuffled predictions from the Italian-adapted model and the Italian Gold Standard, scoring Utility and Fluency from 1 (worst) to 3 (best) as per Table 4.15.

Results in Table 4.20 show that the `LLaMA2Chat-13b-IT` model achieved the highest Utility score of 2.79 and a Fluency score of 2.99. Inter-annotator agreement was strong, with Pearson correlations of 0.81 for Utility and 0.83 for Fluency. For example, for the command "*Distruggi 1 blocco e mettine altri 3 in fila*"[14], the expected output was "*Distruggere quale blocco?*"[15]. However, the `LLaMA2Chat-13b-IT` model produced a more comprehensive question addressing all missing information: "*Quale specifico blocco devo distruggere e quale colore/posizione/direzione deve avere la fila di 3 blocchi?*"[16]. This generalization shows that

[14]In English: "*Destroy 1 block and build another 3 in a row*"
[15]In English: "*Destroy which one block?*"
[16]In English: "*Which specific block should I destroy, and what color/direction/position should the three-*

the model, trained on the manually validated dataset, effectively understands the task at hand: when the Architect issues multiple request in a single command, the Builder needs different information for the different requests.

These results underscore the importance of language-specific training. The Italian model achieved significantly higher Utility and Fluency scores than models relying solely on English data, highlighting the necessity and effectiveness of the MM-IGLU-IT dataset.

Of course, these results have been obtained in the early 2024, and the field of LLMs is rapidly evolving. The results presented here are based on the state of the art at that time, and it is likely that new models will be released in the future that will outperform these results. However, the goal of this work was to demonstrate the feasibility of extending multimodal LLMs to handle situated dialogues in a Minecraft-like world, and the results obtained so far are promising. A more recent evaluation is presented in Section 5.2, where we will see how the results of the previous sections can be improved by using a more recent (and smaller) LLM, such as Phi4 [183].

4.3.6 Extending Multimodal LLMs to handle Situated Dialogues

All the aforementioned work in collaborative environments, such as the experiments conducted in the above sections, has been carried out under the constraint of single-turn interactions. Specifically, the Architect issues a command, and if the Builder encounters an issue in executing it, it has a single opportunity to request clarifications. While this approach allows for basic interaction, it fails to capture the richness of real-world human dialogues, where interactions are multi-turn and often involve incremental exchanges to progressively refine understanding. In this Section, we will explore an idea about how to extend Multimodal LLMs to handle Situated Dialogues in the Minecraft-like world made of blocks. This work has been currently submitted [107] for review. The goal is to extend the previous experiments to a more realistic scenario, where the robot can ask multiple clarifying questions before executing the command. This is a crucial step towards achieving more natural and effective human-robot interactions in real-world scenarios.

The objective of this Section is to shed light on the multi-turn interaction, so in the rest I will focus on the following problems:

- **Single-turn interaction**: The interaction model only allows a single clarification step, which does not reflect realistic communication patterns. In real-world scenarios, interlocutors take turns alternately, asking follow-up questions and providing answers until mutual understanding is achieved.

block row be?"

- **Scarcity of incomplete commands**: Only 13% of the commands in the dataset are incomplete, limiting the ability of the model to generalise to more complex situations where critical information is missing.
- **Limited language diversity**: The dataset includes silver-standard Italian data generated through machine translation, resulting in reduced linguistic specificity and fluency. Additionally, responses in the dataset are drawn from a closed set of answers, limiting the variability and complexity of linguistic output.
- **Simplistic task design**: The tasks are designed with binary outcomes (either the goal is satisfied, or it is not) without modelling intermediate steps or sequences of actions that reflect realistic problem-solving processes.
- **Lack of reasoning over sequences of actions**: The previous experiments do not capture the incremental reasoning necessary to solve complex tasks. In realistic scenarios, the robot should engage in a structured sequence of clarifying exchanges, rather than relying on a single exhaustive clarification attempt. It should be able to plan ahead the sequence of clarifications it should ask, based on the information it has already gathered and the user's initial command.
- **Unrealistic interaction patterns**: The interactions in prior experiments were overly simplistic, focusing solely on whether the goal was satisfied or not, and limiting the complexity of the interaction. In a realistic multi-turn dialogue, the robot needs to make strategic decisions about what information is essential and how to proceed based on partial understanding.

These limitations reveal the need for a more advanced model of interaction, one that goes beyond single-turn clarification and embraces the complexity of multi-turn dialogues. In particular, they motivate the introduction of a structured planning mechanism that enables the robot to formulate and follow a sequence of clarification steps, tailored to the specific ambiguities of each command. This planning-based approach serves as the foundation for the methodology we introduce in the following paragraphs.

In real-world communication, interaction is rarely resolved in a single exchange. Instead, human dialogues are characterised by a **multi-turn process**, where interlocutors take turns alternately, progressively refining their understanding through incremental clarification. This stands in stark contrast to the single-turn interaction model used in prior work, which simplifies the interaction to a single question-answer pair when ambiguities arise.

An example of such a multi-turn interaction is shown in Figure 4.9. In this dialogue, the user issues an ambiguous instruction: "*Add some blocks above the easternmost purple*

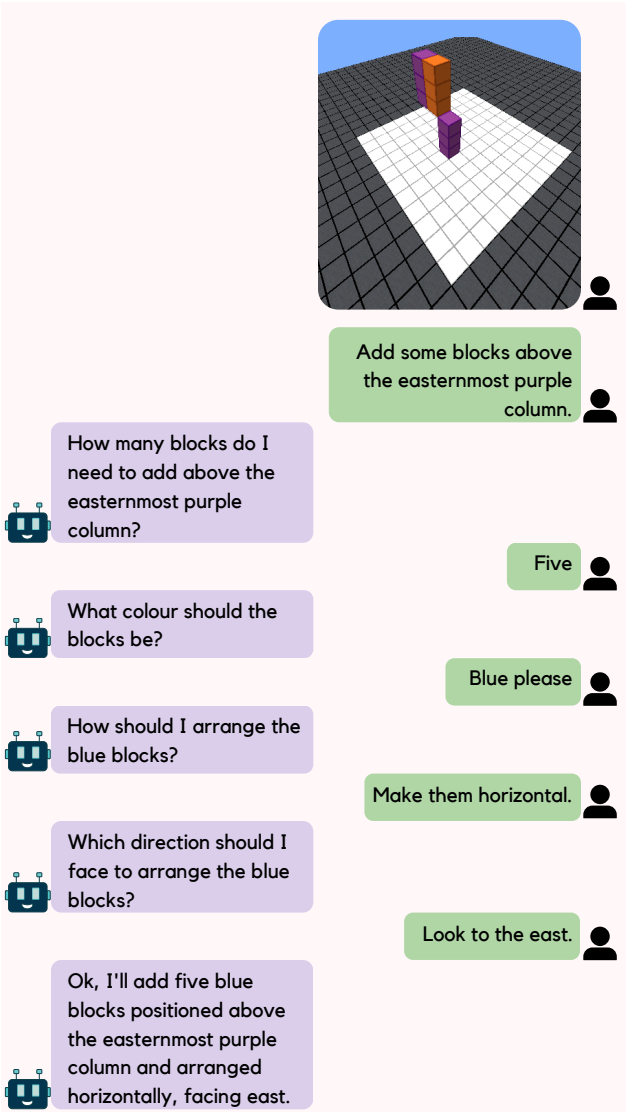


Figure 4.9: An example of a multi-turn interaction between the Architect and the Builder. The user issues an ambiguous command, and the robot engages in a sequence of clarifying questions to resolve ambiguities before executing the command.

column". The robot is unable to execute the command immediately and therefore initiates a sequence of clarifying questions. It first asks for the number of blocks, then their colour, their arrangement, and the orientation required to complete the task. Only after all ambiguities are resolved does it proceed to summarise and execute the command. This simple example illustrates several of the key challenges addressed in this work: the need for incremental clarification, the role of reasoning over spatial relations and attributes, and the necessity to track dialogue context across turns. It also highlights the strategic role of dialogue planning, where each question serves a specific communicative goal and contributes to progressively narrowing down the space of possible interpretations of the user's intention.

An important aspect is to explore **different aspects of incompleteness** in user commands. In some cases, essential information such as colour, quantity, or spatial positioning may be missing, as in the above example. In others, the ambiguity may stem from incomplete specifications of actions or optional information. Determining what to ask and when to ask it is a critical problem, as the robot needs to distinguish between necessary and non-essential clarifications to optimise the interaction process. Moreover, the robot must have a clear understanding of its capabilities and limitations. For example, while the robot may not have the ability to perform actions such as dancing, it should be able to construct a figure resembling a dancer when instructed to do so.

To address these challenges, we propose a novel dialogue framework in which the robot explicitly generates a **dialogue plan** before producing its natural language response. This plan consists of a sequence of communicative goals the model intends to pursue in order to disambiguate the user's instruction. Each goal in the plan corresponds to a specific clarification the robot wants to obtain (e.g., ask about quantity, colour, position, etc.). At each turn, given the user command and the dialogue history, the model first generates this dialogue plan, then selects the first unresolved communicative goal and generates a question accordingly. When the user provides a new answer, the model updates its internal memory and re-generates the dialogue plan based on the accumulated context. If the original plan is still coherent with the updated context, the model retains it (removing the already-resolved goal). Otherwise, it dynamically constructs a new plan that better fits the revised state of the interaction. This approach allows the model to exhibit consistent behaviour across turns, as it does not treat each utterance in isolation but rather grounds each decision in a structured and interpretable process. By decoupling the planning phase from the response generation phase, the system becomes more transparent and robust: the plan serves as an intermediate representation of the intentions of the model, which can be inspected and analysed independently from the surface form of the question. Furthermore, the planning mechanism enables the model to anticipate future clarification needs and structure its interaction accordingly,

leading to more efficient and coherent dialogues.

By structuring the interaction as a multi-turn process, we aim to create a more realistic and flexible **dialogue system**. In our approach, the core idea is that the robot does not reactively generate clarification questions on a turn-by-turn basis. Instead, it first generates an explicit **plan of clarification steps** that it will follow to resolve ambiguities in the user's command. This plan is a high-level, interpretable representation of the information-gathering strategy of the robot, and it guides the subsequent generation of clarification questions during the dialogue, effectively acting as the *dialogue plan*. Each step in the plan corresponds to a specific piece of missing or ambiguous information, such as colour, shape, position, or number of blocks. This explicit planning mechanism makes the model more robust, as it forces the system to consider the full scope of uncertainties before acting. Moreover, it allows for better generalisation, as similar plans may apply to structurally different commands with equivalent ambiguity patterns. Each turn in the dialogue is thus grounded in a precise element of the plan, enabling the model to track progress and adapt its behaviour accordingly. Once the plan has been exhausted or deemed unfeasible, the model can decide whether the command is now executable or if residual ambiguities persist. In this way, the dialogue structure becomes a direct instantiation of the plan, and the behaviour of the model is explainable and controllable through the planning step.

To extend the current multimodal architecture to support multi-turn dialogues, we propose a structured approach where the textual input is transformed into a sequence representing the history of interactions. This history consists of alternating turns between the user and the robot, allowing the model to accumulate context over time and progressively refine its understanding of the user's intention. Crucially, the dialogue is not managed in a reactive fashion. Instead, each user command C, together with an image I of the environment, is first processed by the model to generate a structured plan $P = [s_1, s_2, \ldots, s_n]$, where each step s_i corresponds to a clarification strategy targeting a specific ambiguity or missing piece of information in C. Each step is represented in natural language and guides the model in selecting the type of clarification to perform. At every turn of the dialogue, the model uses this plan P as a high-level roadmap. Specifically, it generates a clarification question based on the first unresolved step s_1 of the current plan. Once the user provides an answer, the history $H_t = [(q_1, a_1), \ldots, (q_t, a_t)]$ is updated, and the model is prompted again to regenerate the plan, now conditioned on the full dialogue history and the image. The model has been trained to reproduce the original plan if it is still consistent with the updated context. In that case, the next question will target the subsequent step s_2. If the context has changed significantly (for example, due to unexpected user input), the model may generate a revised plan. In both cases, the model always proceeds by executing the

first unresolved step of the current plan. This iterative loop continues until the command is fully disambiguated or the dialogue reaches a natural termination. This *planning-then-dialogue* paradigm introduces modularity and transparency into the behaviour of the model. By decoupling high-level reasoning (dialogue planning) from low-level generation (question asking), it enables more strategic interactions and ensures that each exchange is grounded in a coherent and dynamically updated communicative goal.

The integration of dialogues into LLMs requires addressing several conceptual challenges:

- **Contextual memory**: As the dialogue progresses, the model must maintain a coherent memory of past exchanges. This involves tracking previously gathered information and using it to inform future responses. For example, if the user specifies a colour in an earlier turn, the model should avoid asking about it again in subsequent turns.

- **Goal tracking**: Beyond maintaining context, the model must continuously evaluate the overall goal of the interaction. Each turn should be assessed in terms of whether it brings the dialogue closer to satisfying the user's initial request. This requires the model to balance local objectives (clarifying specific ambiguities) with the global objective (executing the user's command).

- **Handling diverse dialogue patterns**: Unlike rigid single-turn interactions, multi-turn dialogues can vary widely in structure. Some interactions may involve straightforward clarifications, while others may require complex, multi-step reasoning. The model must be capable of adapting to different dialogue flows without losing track of the user's intent.

- **Decision-making during interaction**: In each turn, the robot must decide the most appropriate action, whether to request additional information, provide feedback, or execute part of the command. This decision-making process is critical for achieving an efficient and successful interaction.

The integration of multi-turn dialogues into the multimodal architecture naturally enhances the role of multimodality by emphasising the interplay between textual and visual inputs over time. In this extended framework, the input image represents the perception of the robot about the environment, which may change dynamically as the interaction progresses. The goal is to ensure that **visual information remains relevant** across multiple dialogue turns, aiding in the resolution of ambiguities and guiding the actions of the robot. The system can exploit visual cues more effectively to resolve ambiguities without requiring explicit clarification. For instance, consider a scenario where the user commands, "*Remove the tallest block from the map*". If there is a single block that stands out as the tallest,

the robot can infer the missing information directly from the image without needing to ask follow-up questions. Conversely, if multiple blocks appear similar in height, the robot must initiate a dialogue to gather additional information from the user.

Moreover, multi-turn dialogues allow to introduce greater **semantic diversity** compared to single-turn interactions. Commands can vary in their degree of ambiguity, incompleteness, and required reasoning steps. Furthermore, the same visual scene may lead to entirely different dialogues depending on the user's goal. For example:

- Two users observing the same scene may issue commands with distinct goals, leading to divergent interactions despite identical visual inputs. Such interactions are fundamental for a robot as they show the variability of a dialogue.
- Different scenes may still result in similar dialogues if the underlying tasks share a common structure or purpose, and so it will help the Builder understand differences and similarities between dialogues.

The adoption of an explicit dialogue planning mechanism offers several advantages that go beyond the immediate execution of the user's command. First, it facilitates **generalisation** by allowing the model to abstract from specific surface forms and focus instead on the underlying communicative intentions. Since each plan is expressed as a structured list of goals, the same high-level plan can be realised through different linguistic strategies, depending on the language, context, or user preferences. Second, this architecture opens the possibility of **transfer across domains**. For example, while the dialogue in MM-IGLU takes place in a Minecraft-like environment, the dialogue planning procedure is domain-agnostic and could be reused in different situated dialogue settings, such as human-robot collaboration in household tasks or industrial scenarios. In such cases, the model could be fine-tuned to produce domain-specific responses while maintaining the same structure of dialogue plans, leading to improved adaptability and reduced training costs. Of course, the categories of incompleteness and the types of clarifications may vary across domains, but the underlying principles of dialogue planning and multi-turn interaction remain applicable. Finally, the dialogue plan acts as a **transparent control interface** for the interaction. Since each generated plan explicitly lists the next communicative steps, it becomes possible to inspect and intervene in the dialogue process. This is particularly valuable in safety-critical scenarios, where the ability to predict and control an action of the robot is essential.

Building upon these extensions, we define new **research questions** aimed at advancing multimodal dialogue systems beyond single-turn interactions.

- How can we design an LLM-based architecture capable of learning not only to respond

but to **engage in goal-oriented multi-turn dialogues** where each turn incrementally refines the understanding of incomplete commands?

- What **types of dialogue strategies** emerge when the model is trained on tasks with varying levels of incompleteness, and how do these strategies adapt to **different sources of ambiguity**, such as missing attributes or complex spatial relations?
- How does the integration of **dynamic visual context** influence the effectiveness of multi-turn dialogues, and to what extent can visual information **reduce the need for explicit clarifications** by the robot?
- How can we **evaluate the overall success** of these multi-turn interactions, ensuring that the robot not only completes tasks efficiently but also demonstrates adaptability in real-world collaborative environments?
- How can **explicit dialogue plan** generation guide the decision-making of the model over multiple turns, and how does it **impact on the coherence and task success**?

Empirical Investigation. The previously introduced architecture based on explicit planning has been tested on the newly collected dataset of situated dialogues introduced in the earlier section (4.3.2) of this Thesis. The goal of this experimental evaluation is to assess the ability of multimodal LLMs to engage in coherent, goal-oriented interactions that span multiple turns, and to verify whether explicit planning enhances the clarity, control, and robustness of such interactions. We formulate this as a generation task where the model receives a dialogue history in natural language, together with an image of the environment, and must produce a coherent and grounded next utterance as a response. In our proposed setting, the model is trained to first generate a *dialogue plan*, i.e., a sequence of abstract intentions that determine the strategy to resolve the ambiguity, and then to produce the textual utterance corresponding to the first step in that plan. At the next turn, the model receives the updated history, is asked to generate a plan again (either confirming the previous one or adapting it), and proceeds with the next step. To evaluate this setup, we compare different versions of multimodal instruction-following models on the MM-IGLU-Dialogues dataset. The models we consider are based on LLaVA and MiniCPM, both of which are capable of handling visual and textual inputs. The dialogues in the dataset are rich in linguistic and visual diversity, and are annotated in both English and Italian, thus allowing us to assess performance in both monolingual and cross-lingual scenarios.

Several **architectures** are of interest for this task. For instance, the multimodal neural models such as LLaVA (explored in Section 4.3) is appealing due to its straightforward integration of language and vision, while MiniCPM (explored in Section 3.3) offers an excellent

trade-off between the number of parameters and performance. The final training approach for this task might involve fine-tuning all the parameters of a model, as demonstrated in Section 3.3, or alternatively applying LoRA, as discussed in Section 3.2, or Q-LoRA, as shown in Section 3.1. These models, which process both the dialogue history and the image simultaneously, fully exploit the attention mechanism of the underlying language model, to decide which are the relevant parts of the history. At run-time, the entire dialogue history will be used to inform decision-making, ensuring that each response is based on the accumulated context of previous exchanges.

During the Thesis, the dataset for experimentation that will be used for this experimentation has already been developed in a full dialogic form, featuring interactions between an Architect and a Builder, similar to the setup used in MM-IGLU. The key difference here is that the Builder is allowed multiple turns to ask questions and resolve ambiguities in the given command. Each dialogue should remain truthful and consistent with respect to the environment shown in the images we previously generated (as discussed in Section 4.3.1). Moreover, each dialogue must have a defined end: either the command becomes executable or it remains inconsistent to some degree.

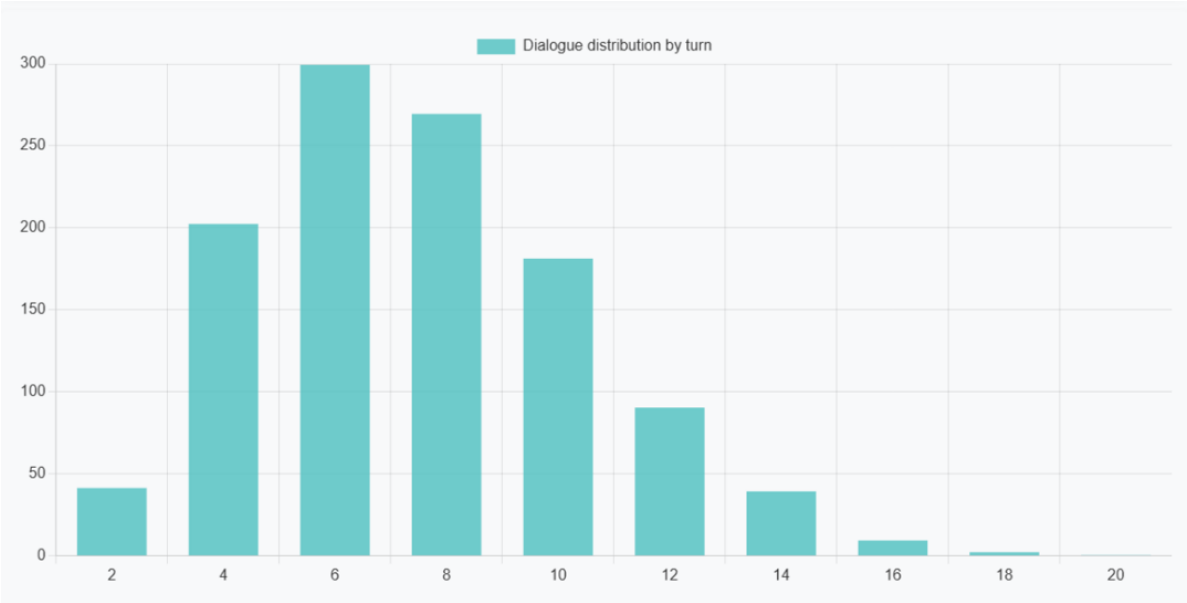


Figure 4.10: The turns distribution of the collected dialogues. On the X axys the number of turns and on the Y axis the number of dialogues ending with in that specific turn.

The **interaction** occurs in natural language, specifically in English and Italian, and must consistently reference the environment presented in the images. The Architect has a specific objective, such as constructing a square, removing all blocks from the map, creating figures representing people, geometric shapes, simple objects or flags, or adding and removing parts

of a structure. The Architect should follow its objective until completion. While the Builder must decide if it has all the relevant information to execute the command or, as usual, ask some clarification questions to gather them.

We involved five human annotators to generate these **dialogues**, starting from the images produced in MM-IGLU. Annotators were given complete freedom to write whatever they wanted and for as many turns as they deemed necessary. The dataset contains a variety of commands: some are intentionally incorrect, others include misleading answers, and some even contain nonsensical words, all designed to evaluate the robustness of future models. In total, we collected approximately $1,136$ dialogues in both Italian and English, with an average of about 6 turns per dialogue. Figure 4.10 shows the turns distribution of all the collected dialogues: a small number of dialogues are straightforward and they end in two turns only. On the other hand, the vast majority need more clarification questions to disambiguate inconsistencies or collect all the information, with a few of them culminating in 20 turns.

Category	**Description**
Color	Clarifies the colour of blocks when unspecified in the command.
Number	Asks for the number of blocks required to fulfil the task.
Orientation	Seeks the orientation of the builder (e.g., North, South, East, West).
Direction	Determines the direction for placing a sequence of blocks.
Block Missing	Identifies cases where a specified block is absent on the map and requests clarification.
Confirmation	Confirms that the command can be executed; includes a Confirmation With Recap subcategory that reiterates the command with a summary of the information gathered.
Disposition	Inquires about the arrangement of blocks, with a subcategory Precise Disposition for detailed alignment questions.
Position	Clarifies the location where blocks should be placed or removed from, with subcategories Precise Position and Precise Block for detailed positioning or block identification.
Not Executable	Indicates when the command cannot be executed in cases of inconsistencies, with subcategories such as Command, Action, and Color Not Found for specific reasons.

Table 4.21: Dialogue clarification categories and their descriptions.

Table 4.21 presents the taxonomy of clarification strategies employed during multi-turn dialogues. Each category reflects a specific type of ambiguity or missing information the model may need to address before task execution. These include attributes such as colour or size, spatial relations, object references, structural properties, and confirmations of understanding. The structured plan generated by the model at each turn includes a ranked list

of these categories, guiding the generation of the actual response. This abstraction enables the model to maintain a consistent reasoning path and reuse the same plan if still valid in subsequent turns.

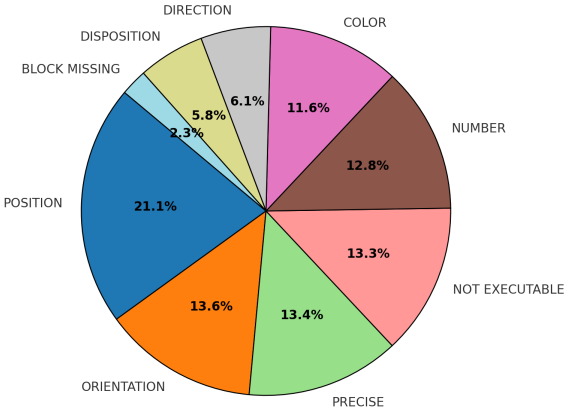


Figure 4.11: Category Distribution in the Annotated Dataset.

The distribution of these adopted missing information types in the plans of our dialogue dataset is reported in Figure 4.11. It can be observed that they are almost evenly distributed, with a few examples of Block Missing. Notice that the Confirmation is not reported, since all our dialogues end with a confirmation from the Builder, since our goal was to provide dialogues where, at some point, all ambiguities are resolved!

By leveraging these multimodal architectures and evaluating them on the constructed dialogic dataset, we can determine whether the models are capable of effectively engaging in goal-oriented multi-turn dialogues, identify emerging dialogue strategies, and analyse the various sources of ambiguity encountered during interactions. Additionally, this evaluation allows us to assess how well the models integrate dynamic visual context and measure their overall success in achieving the Architect's intended goals.

Experimental Setup and Evaluation Objectives. To evaluate the effectiveness of structured dialogue planning in multimodal situated dialogues, we conduct a comprehensive empirical study addressing three main objectives: (*i*) assessing whether fine-tuning on the collected bilingual dataset leads to improvements over zero-shot performance, (*ii*) determining the impact of planning on dialogue efficiency at both the individual turn and full-dialogue level, and (*iii*) evaluating the extent to which structured planning supports cross-lingual generalisation.

Table 4.22 reports statistics on the annotated dataset used in our experiments. It includes the number of dialogues per split, the average number of dialogue turns, and the average length of both individual turns and final recaps. These values are presented separately for English and Italian to reflect language-specific variation. The dataset contains 908 training,

113 development, and 115 test dialogues, with a consistent average of approximately 7.5 turns per dialogue across all splits. Turn and recap lengths are slightly longer in English than in Italian, but comparable across splits, confirming the internal consistency of the dataset. These figures reflect the richness and diversity of the interactions, supporting a robust evaluation of both monolingual and cross-lingual performance.

	Lang	Train	Dev	Test
Dialogues Number	-	908	113	115
AVG Turns Number	-	7,52	7,40	7,60
AVG Turn Lenght	EN	10,71	9,77	10,37
	IT	9,77	8,91	9,56
AVG Recap Lenght	EN	23,71	21,30	23,09
	IT	22,08	19,87	22,19

Table 4.22: Dataset statistics: number of dialogues, average turns, and turn/recap lengths in English and Italian, split for training, development, and testing.

Models and Training Strategy. The models used in this study, LLaVA [160] and MiniCPM [284], are trained using a unified approach. Both undergo fine-tuning on the MM-IGLU-Dialogues dataset, using LoRA with rank $R = 64$ and scaling factor $\alpha = 64$, applied to the attention and output layers of the language decoding module. Additionally, the vision encoder is jointly fine-tuned, following the setup described in the MiniCPM paper. Training is conducted on 4 A100 GPUs with a total batch size of 8 and a learning rate of $5 \cdot 10^{-5}$. Full prompt templates and additional experimental details are reported in Appendix **??**. During training, the model is conditioned on both the dialogue history and the visual context. It is trained to first produce a structured clarification plan in the form of a comma-separated list of strategy categories, and then to generate a natural language response, conditioned exclusively on the first unresolved category in the plan. At each turn, the model either preserves the remaining plan if consistent with the updated context, or regenerates a new plan before responding. We test LLaVA and MiniCPM under three configurations: (a) zero-shot generation using only system prompts, (b) fine-tuning on dialogue history without planning, and (c) fine-tuning with structured planning. The models receive the dialogue history up to the current turn, the user's command, and the environment image. For each turn, the model must either generate a clarification question (Q) if it lacks sufficient information, or a recap (RC) summarising the current understanding of the task. More details about the used prompts can be found in Appendix F.1.

Evaluation Metrics. The quality of clarification questions and recaps is evaluated through cosine similarity between their sentence embeddings and gold-standard outputs, using the

Sentence-BERT model `all-MiniLM-L6-v2`. For planning, we compute Precision, Recall and F1-score over the predicted plan categories. Evaluation is performed at two levels: a *turn-driven evaluation*, focusing on individual steps of the interaction, and a *dialogue-level evaluation*, measuring the ability to sustain full, coherent, and efficient multi-turn conversations.

Results and Discussion. Table 4.23 compares LLaVA and MiniCPM across both evaluation settings, highlighting the impact of fine-tuning with structured planning. The results show that fine-tuned models effectively classify planning categories, achieving over 90% F1-score in plan generation. In contrast, baseline zero-shot models struggle to generate contextually appropriate responses, with LLaVA scoring only 0.37 in the similarity between its clarification questions and expected ones. Fine-tuning significantly improves performance, with LLaVA reaching 0.82 when incorporating planning. Comparing the two models, fine-tuned MiniCPM outperforms LLaVA in question and recap generation, while both models achieve similar performance in planning.

Setting	**Model**	**Plan**			**Generation**	
		P	**R**	**F1**	**Question**	**Recap**
zero-shot	LLaVA	-	-	-	0.37	0.39
	MiniCPM	-	-	-	0.55	0.71
Fine-tuned	LLaVA	0.93	0.90	0.92	0.82	0.91
(with Plan)	MiniCPM	0.90	0.91	0.91	**.84**	**.96**

Table 4.23: Comparison of LLaVA and MiniCPM in zero-shot and fine-tuned settings for plan accuracy and response generation.

To further assess the impact of explicit planning beyond fine-tuning alone, we conduct a detailed evaluation comparing MiniCPM in settings with and without structured planning. Given its consistently superior performance across previous experiments, we focus exclusively on MiniCPM for this analysis. Table 4.24 presents results across three configurations: zero-shot, fine-tuning on dialogue history alone, and fine-tuning with explicit planning. In the monolingual EN→EN scenario, fine-tuning without planning significantly improves generation quality compared to the zero-shot model (e.g., Question Generation rises from 0.548 to 0.768, while Recap Generation increases from 0.706 to 0.921). However, the addition of planning further boosts performance, with Question Generation reaching 0.843 and Recap Generation improving to 0.955. These findings confirm that fine-tuning alone enhances response quality, but planning refines the interaction by making clarification requests more structured and summaries more accurate. We also examine cross-lingual generalization, where the model is trained in one language and tested in another. The results reveal that

Setting	Language		Generation	
	Train	Test	Question	Recap
zero-shot	-	EN	0.548	0.706
	-	IT	0.595	0.727
Fine-tuned (no Plan)	EN	EN	0.768	0.921
	EN	IT	0.659	0.861
	IT	IT	0.794	0.925
	IT	EN	0.638	0.908
Fine-tuned (with Plan)	EN	EN	**0.843**	**0.955**
	EN	IT	**0.735**	**0.930**
	IT	IT	**0.867**	**0.947**
	IT	EN	**0.738**	**0.927**

Table 4.24: MiniCPM performance in zero-shot, fine-tuned, and planning-enhanced settings across monolingual and cross-lingual evaluations.

models fine-tuned only on dialogue history experience a notable drop in Question Generation accuracy when applied across languages (EN→IT: 0.659, IT→EN: 0.638). This degradation suggests that, without structured planning, the model struggles to transfer the reasoning process necessary for effective question formulation. In contrast, incorporating planning significantly mitigates this issue: the EN→IT model achieves 0.735, and the IT→EN model reaches 0.738, both exceeding the performance of the history-only model in monolingual settings. Interestingly, Recap Generation remains more stable across settings, with smaller differences between monolingual and cross-lingual results. This indicates that summarization is a more transferable skill, whereas effective clarification question generation benefits more directly from structured planning. These findings underscore the role of explicit planning not only in improving dialogue quality but also in strengthening the ability to generalize across languages, reinforcing the importance of structured Multi-Modal training for robust cross-lingual HRI applications.

Dialogue Evaluation. In the previous evaluation, we analyzed the performance of the model at the individual response level, leveraging the gold-standard dialogue history to assess its ability to generate appropriate clarification questions and recaps. However, in this phase, we shift our focus to evaluating the overall quality of full dialogues rather than isolated responses. Specifically, we aim to answer the question: *How well can the model sustain a complete interaction that dynamically adapts to user input while effectively gathering the necessary information?* To achieve this, we deploy the fine-tuned MiniCPM model in an interactive setting, allowing it to generate dialogues in real-time. Two independent annotators, who had not participated in dataset annotation, conducted this evaluation. Prior to

the study, they were given only a brief introduction: they reviewed ten training examples and read descriptions of the dialogue categories to familiarize themselves with the interaction structure. Each annotator randomly selected a test case, viewed the corresponding world image, and initialized the system by providing the user's first request, after reading the gold-standard recap to understand the intended task outcome. From that point, they interacted with the model by answering its clarification questions, ensuring they provided the necessary details for the system to achieve the expected goal. This process was repeated across 40 dialogues, enabling a comprehensive evaluation of how effectively the model conducts a full conversation and adapts to evolving user input.

Setting	**AVG Length**	**Dialogue Efficiency**	**Relevance**	**Fluency**
no Plan	12.5	1.05	3.13	4.88
with Plan	11.6	0.98	**4.25**	**4.90**

Table 4.25: Dialogue evaluation results

To assess how efficiently the model conducts interactions, we measure the Dialogue Efficiency Cost, defined as the ratio between the actual number of turns and the ideal number needed to complete the task:

$$\text{Dialogue Efficiency Cost} = \frac{\text{Number of Turns}}{\text{Ideal Number of Turns}} \tag{4.5}$$

A lower score indicates that the model gathers necessary information with fewer exchanges, highlighting the impact of planning on dialogue effectiveness.

At the end of each interaction, annotators completed a brief questionnaire to assess the overall quality of the dialogue. Specifically, they provided ratings on two key dimensions: Relevance and Fluency. Relevance measures the ability to seek and provide only necessary information, ensuring that each turn contributes meaningfully to task completion without unnecessary or redundant exchanges. A higher relevance score indicates that the dialogue remains focused and aligned with user intent. Fluency assesses how naturally and coherently the assistant formulates questions and responses, prioritizing clarity, grammatical correctness, and ease of comprehension. Poor fluency may result in unnatural phrasing or disjointed interactions, while high fluency ensures smooth and user-friendly communication. Annotators rated these dimensions on a five-point scale, from 1 (Extremely Poor) to 5 (Excellent). The full questionnaire and scoring details are provided in Appendix F.3. Table 4.25 presents a comparative analysis of model performance with and without planning across four key metrics: dialogue efficiency, relevance, fluency, and average dialogue

length. The results show that incorporating a planning phase leads to notable improvements in dialogue quality. In particular, the planned model achieves a significantly higher relevance score (4.25 vs. 3.13), suggesting that it generates more contextually appropriate and purposeful exchanges, avoiding unnecessary, off-topic questions. Similarly, fluency improves slightly (4.90 vs. 4.88), indicating that while both settings produce grammatically well-formed responses, the planned model better maintains conversational coherence. From an efficiency perspective, the model with planning achieves a dialogue efficiency score closer to the optimal value of 1 (0.98 vs. 1.05), suggesting that it requires fewer redundant turns to reach the intended task completion. This is further reflected in the average dialogue length, where the planned model produces slightly shorter interactions (11.6 vs. 12.5), potentially indicating a more structured and goal-driven exchange. These results indicate that explicit planning improves dialogue relevance while offering a moderate gain in efficiency by keeping interactions more focused and reducing unnecessary exchanges. An error analysis is shown in Appendix F.2.

Chapter 5

Conclusions and Discussion

In this Chapter, the discoveries and the best results achieved in this Thesis are summarised. The progression started from traditional fine-tuning, moved towards applying modern adaptation techniques such as LoRA and Q-LoRA to Large Language Models, scaled further by extending knowledge and capabilities to solve multiple tasks concurrently, and culminated in the integration of Vision and Language for Visual Question Answering tasks. These methodologies were subsequently applied to fine-tune other models on various heterogeneous downstream tasks.

This concluding chapter revisits the overarching objective set forth in the introduction: enabling intelligent systems, specifically robots, to perform situated language understanding in dynamic, real-world environments. Throughout the thesis, key challenges that hinder current AI systems from achieving this level of interaction were addressed. By focusing on efficient model adaptation, multilingual capabilities, and multimodal integration, the methodologies and contributions presented align directly with the goal of creating adaptable, context-aware systems. From fine-tuning approaches that reduce resource requirements to introducing models capable of asking clarification questions, each contribution advances the thesis objective of fostering interactive, collaborative human-robot dialogue in diverse and unpredictable scenarios.

Central to this thesis is the Situated Understanding of Natural Language, with the aim of showing how intelligent robots can understand, process, and reason in the real world. Moreover, it is shown that intelligent robots should ask questions when they lack understanding or sufficient information. This shifts the perspective from passive answerers to active collaborators in interaction with humans. For this purpose, the MM-IGLU resource was developed, evaluated, and subsequently released to the broader public.

5.1 Summary of the Contributions

This thesis has introduced several significant contributions in the fields of Large Language Models (LLMs), multimodal learning, and situated language understanding. The work presented addresses both foundational advancements and practical applications, with a specific focus on enabling intelligent robots to interact meaningfully in real-world environments. Below is a summary of the key contributions:

- **Efficient fine-tuning techniques for LLMs:** This thesis explored modern adaptation techniques, including LoRA and Q-LoRA, demonstrating how large models can be fine-tuned efficiently using limited computational resources, such as a single 16GB GPU. These methods significantly reduce the hardware requirements for fine-tuning large models, enabling broader accessibility to LLM research.

- **First LLM for multilingual dependency parsing:** textttU-DepPLLaMA, the first LLM capable of performing end-to-end dependency parsing across multiple languages, was developed. This capability can be exploited by robots, as it enables syntactic parsing that can be leveraged in the interpretation of complex linguistic commands, facilitating more accurate and nuanced interactions.

- **First LLM to solve multiple linguistic tasks in Italian:** `ExtremITA` is the first LLM designed to handle a wide range of linguistic tasks in Italian. By demonstrating that a single model can solve numerous tasks concurrently, this contribution helps robots mimic human-like multitasking capabilities, enhancing their linguistic competence in low-resource languages. It's worth noting that this system participated in the EVALITA 2023 challenge: it achieved first place in 41% of the subtasks (9 out of a total of 22) and showcased top-three performance in 64% (14 out of 22) of them. These subtasks encompass various semantic dimensions, including Affect Detection, Authorship Analysis, Computational Ethics, Named Entity Recognition, Information Extraction, and Discourse Coherence.

- **Multimodal learning for Visual Question Answering (VQA):** This thesis showed an application of extended LLMs with multimodal capabilities `MM-VQA-it`, enabling them to process and reason over visual inputs. The proposed approach achieved state-of-the-art performance on Visual Question Answering tasks, demonstrating that combining vision and language is essential for real-world problem-solving.

- **Situated language understanding for robotics:** The `GrUT` approach was introduced as a framework for interpreting situated language commands in real-world envi-

ronments. GrUT achieves state-of-the-art performance in interpreting commands in English, Italian, and even in multilingual scenarios involving a mix of English and Italian. Unlike traditional models that provide only linguistic interpretations, GrUT grounds the interpretation by producing object identifiers, enabling robots to directly map the interpretation into executable robotic primitives. This allows robots equipped with GrUT to generate ready-to-execute commands and seamlessly perform the requested tasks.

- **An LLM for asking questions in an end-to-end manner:** This thesis introduced BART-IGLU, a Language Model capable of actively asking clarification questions in a Minecraft-like environment. This marks a paradigm shift from passive answerers to active collaborators in human-robot interaction. BART-IGLU enhances interaction robustness by prompting the user for additional information when commands are ambiguous or incomplete.

- **Development of the MM-IGLU resource:** A significant outcome of this thesis is the creation of MM-IGLU, a complex and robust resource designed for evaluating models in multimodal situated dialogues. MM-IGLU includes commands of varying complexity and supports cross-modal evaluation, allowing researchers to assess the robustness and performance of models across different input modalities. The resource was rigorously evaluated and subsequently released to the public, fostering further research in this domain.

- **Extension of MM-IGLU to Italian:** In addition to the original MM-IGLU resource, this thesis presents MM-IGLU-it, an extension that introduces support for the Italian language, making the resource truly multilingual. This extension enables models to be trained and evaluated in a multilingual setting, broadening the scope of situated language understanding to include low-resource languages.

Together, these contributions advance the fields of efficient model adaptation, multimodal interaction, and situated dialogue systems, providing a solid foundation for future research in human-robot collaboration.

5.2 Too Many Models, Too Little Time: The Recent Proliferation of Modern LLMs

The pace at which new Large Language Models (LLMs) are being released has become increasingly difficult to follow. Barely a week goes by without the announcement of a new

model, sometimes open-source, sometimes commercial, each claiming incremental or even breakthrough advancements in capabilities, efficiency, or modality integration. This phenomenon, often referred to as the "LLM arms race", reflects both the intense competition between major AI research labs and the growing demand for models tailored to specific domains or applications.

In early 2025 alone, numerous models have emerged, ranging from purely text-based architectures to highly capable multi-modal systems. On the textual side, we have seen the release of `GPT-4.5` by OpenAI [202], `Claude 3.7 Sonnet` by Anthropic [13], `Grok-3` by xAI [278], `LLaMA 3.1` [178], `LLaMA 3.2` [179], `LLaMA 3.3` [180] and `LLaMA 4` [181] by Meta, and `DeepSeek R1` [238], each contributing to the growing diversity of LLMs with different training philosophies, sizes, and intended use-cases.

In parallel, the multi-modal landscape has evolved rapidly, with models such as `GPT-4.1` (OpenAI) [201], `Gemini 2.5 Pro` [90] and `Gemini 2.5 Flash` [90] (Google DeepMind), `Qwen2.5-Omni` [281] (Alibaba), and `OmniMamba` [291] (HUST) pushing the boundaries of what it means for an AI system to process and reason over text, images, audio, and even video inputs. These models are not merely extensions of their textual predecessors; they often feature unified tokenisation pipelines and sophisticated alignment strategies to enable seamless cross-modal understanding and generation.

In such an overwhelming and dynamic context, it becomes essential to focus on individual models that stand out for specific technical or practical reasons. For the scope of this thesis, I have selected `Phi-4` [183], a compact yet powerful multi-modal model developed by Microsoft. Phi-4 is particularly notable for its native support of multi-modal inputs, its efficient architecture, its size (it's relatively small with its 5.6 billion paremeters), and its promising performance across a variety of tasks. It represents an ideal candidate for investigating grounded language understanding in situated settings, such as Human-Robot Interaction.

In the following subsection, a dedicated discussion of the `Phi-4` model, covering its architecture, training, and experimental results on the `MM-IGLU` and `MM-IGLU-it` datasets is provided.

5.2.1 The Phi-4 Model: A Compact and Multimodal Transformer

`Phi-4` [183] is a transformer-based language model developed by Microsoft, designed from the ground up to support multi-modal reasoning. Unlike earlier families of models such as `LLaMA` and `LLaMA 2`, which were primarily optimised for textual tasks and require additional components or adapters to handle other modalities, `Phi-4` natively integrates visual and

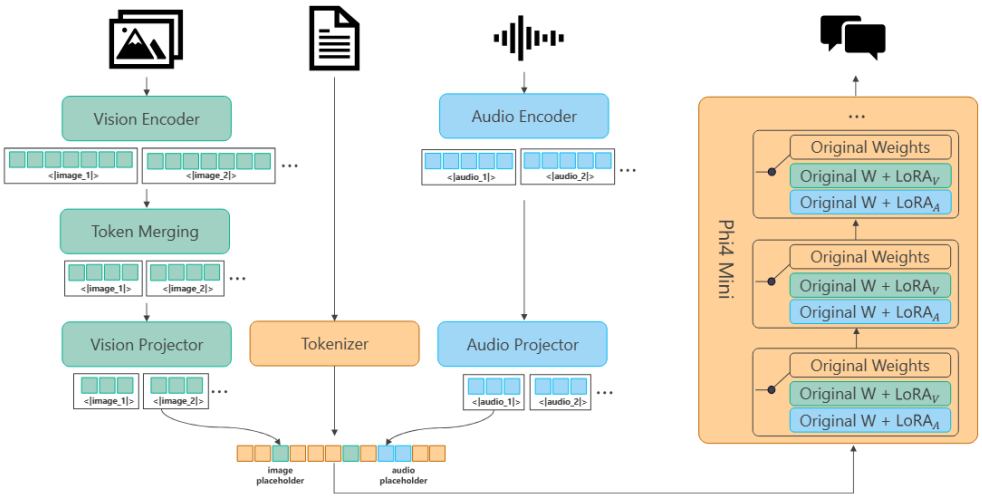


Figure 5.1: The architecture of the Phi-4 model, taken from [183].

textual information within a single architecture.

This design enables the model to perform grounded understanding and reasoning directly over interleaved sequences of text and image tokens. Moreover, `Phi-4` is significantly smaller than most contemporary LLMs, with 5.6 billion parameters, making it a computationally efficient alternative without sacrificing performance on complex tasks.

The architecture of `Phi-4` follows a modular yet unified design that allows the model to process text, images, and audio within a single sequence of tokens. As shown in Figure 5.1, input data from different modalities is first encoded and projected into a shared token space. Specifically, visual data passes through a *Vision Encoder*, followed by a *Token Merging* module and a *Vision Projector*, which compresses the visual embeddings and formats them as token-like inputs. A similar process applies to audio inputs, which are handled by an *Audio Encoder* and an *Audio Projector*.

These projected image and audio tokens are inserted into the token stream via special placeholders and merged with the output of a standard text tokenizer. The resulting multimodal sequence is then processed by a decoder-only transformer core `Phi-4 Mini`, which retains the original pretrained weights and integrates additional adapters using modality-specific LoRA layers ($\texttt{LoRA}_\texttt{V}$ and $\texttt{LoRA}_\texttt{A}$). This architecture allows the model to extend its capabilities to vision and audio without retraining the full network from scratch.

Training is performed in two main phases: first, the base model is trained or adapted on high-quality text-only data using synthetic instruction tuning, following the "textbook-style" data curation approach typical of the Phi series [93, 125]. In a second stage, multi-modal capabilities are introduced by aligning visual and audio representations with their textual

counterparts through supervised learning on interleaved multimodal datasets. LoRA layers are trained separately for each modality, enabling efficient fine-tuning and preserving the compact size of the core model ($5.6B$ parameters).

This architecture makes `Phi-4` particularly suited to grounded and situated tasks where context must be extracted from multiple modalities and interpreted jointly in a coherent dialogue.

5.2.2 Evaluating Phi4 on the MM-IGLU and MM-IGLU-it Benchmarks

To evaluate the capabilities of `Phi-4` in grounded instruction understanding, we rely on the `MM-IGLU` benchmark, and its Italian extension `MM-IGLU-it`. These datasets are designed to test the ability of a model to interpret ambiguous natural language commands in a situated and multimodal context. Each example includes a user command, an image of the environment, and the target interpretation, as already discussed in Chapter 4.3.1. Models are evaluated on two main tasks: a binary classification task to determine whether a command is clear or ambiguous, and a generation task where the model must produce a clarifying question in case of ambiguity.

The `MM-IGLU-it` dataset follows the same structure as the original benchmark, but all commands and annotations are first translated into Italian and then manually validated and curated, as introduced in Chapter 4.3.5. This allows us to assess not only multimodal reasoning, but also the ability of the model to generalise across languages, including low-resource ones such as Italian.

Training Procedure. We fine-tuned the `Phi-4 Instruct (5.6B)` model on both the English and Italian versions of the MM-IGLU dataset. During training, each input consisted of an interleaved sequence of the environment image and the user command, while the output varied depending on the task: either a *Yes*/*No* answer (for the classification task) or a natural language clarification question (for the generation task). All training was performed using LoRA adaptation on the modality-specific components only, keeping the original weights of the base model frozen. This approach ensured efficient training on a single $40GB$ GPU without sacrificing any ability of the model to integrate visual and linguistic cues.

Evaluation. The results are presented in Table 5.1 for the classification task and Table 5.2 for the generation task. For each model, we report performance in different training and testing configurations: models trained and tested in English, models trained in English and tested in Italian (cross-lingual generalisation), and models trained and tested in Italian. Each table includes F1 scores for both the positive class (clear command), the negative class

(ambiguous command), and the macro average. Moreover, we also report the performance of the `LLaMA2Chat` model, as already discussed in Chapter 4.3.3, for a direct comparison. Finally, we also include the performance of the `Phi-4` model in zero-shot settings, where it was not fine-tuned on the MM-IGLU dataset.

Model name	**Tr. Lan**	**Test Lan**	**F1 Pos**	**F1 Neg**	**M-F1**
`LLaMA2Chat-13b-EN`	EN	EN	96.43%	67.16%	81.80%
`Phi4-5.6B-0shot`	-	EN	43.33%	24.57%	33.95%
`Phi4-5.6B-EN`	EN	EN	86.44%	61.92%	74.18%
`LLaMA2Chat-13b-EN`	EN	IT	70.07%	24.29%	47.18%
`Phi4-5.6B-EN`	EN	IT	93.19%	43.54%	68.36%
`LLaMA2Chat-13b-IT`	IT	IT	97.81%	66.67%	82.24%
`Phi4-5.6B-0shot`	-	IT	75.98%	21.81%	48.89%
`Phi4-5.6B-IT`	IT	IT	94.22%	68.18%	81.20%

Table 5.1: The Classification performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two. The evaluation is divided into the Language of Training (Tr. Lan) and the Language of Testing (Test Lan).

Classification Performance. In the classification task (Table 5.1), `Phi4-5.6B` demonstrates a substantial improvement when fine-tuned on English data, reaching a macro F1 of 74.18%, which narrows the gap with the larger `LLaMA2Chat-13b` model. Despite its smaller size and the use of LoRA adapters only, `Phi4` shows solid results, particularly on the ambiguous class (F1 Neg: 61.92%). In contrast, its zero-shot performance is limited (33.95% M-F1), highlighting the importance of domain-specific adaptation for accurate ambiguity detection.

On the Italian test set, the zero-shot `Phi4` reaches 48.89% M-F1, outperforming the English-only `LLaMA2Chat` in the same setting, which suggests that `Phi4` possesses a degree of cross-lingual robustness, albeit with weak recall on the negative class. Notably, the Italian fine-tuned version of `Phi4` achieves 81.20% macro F1, with an F1 of 68.18% on the ambiguous class, on par with `LLaMA2Chat-13b-IT`. This result confirms that `Phi4`, despite its smaller scale, can match the performance of much larger models when properly adapted, while remaining computationally lightweight and easily deployable.

Generation Performance. In the generation task (Table 5.2), the fine-tuned English version of `Phi4` achieves a remarkably high macro F1 score of 87.39%, with strong performance across both positive and negative classes. This suggests that the model is particularly well-suited to generating clarification questions when properly tuned, outperforming older

Model name	Tr. Lan	Test Lan	F1 Pos	F1 Neg	M-F1
LLaMA2Chat-13b-EN	EN	EN	93.95%	47.89%	70.92%
Phi4-5.6B-0shot	-	EN	26.15%	23.28%	24.72%
Phi4-5.6B-EN	EN	EN	98.35%	76.42%	87.39%
LLaMA2Chat-13b-EN	EN	IT	70.01%	0.00%	35.00%
Phi4-5.6B-EN	EN	IT	84.14%	0.00%	42.07%
LLaMA2Chat-13b-IT	IT	IT	93.62%	44.16%	68.89%
Phi4-5.6B-0shot	-	IT	68.88%	0.00%	34.44%
Phi4-5.6B-IT	IT	IT	90.53%	46.61%	68.57%

Table 5.2: The Generation performance is divided into F1 of the positive class (the command is clear), F1 of the negative class (the command is ambiguous), and the Macro F1 of the two.

models despite being significantly smaller.

In the zero-shot setting, however, performance drops sharply to 24.72%, confirming that clarification generation requires substantial task-specific adaptation. When tested on Italian, the zero-shot Phi4 attains 34.44% macro F1comparable to LLaMA2Chat in the same setting, although with no recall on the ambiguous class.

Crucially, the Italian fine-tuned version of Phi4 reaches 68.57% macro F1, with a clear gain in F1 on the ambiguous class (46.61%), matching the overall performance of LLaMA2Chat-13b-IT. These results reinforce the idea that Phi4 is a viable and efficient alternative for multilingual generation in grounded settings, provided that moderate fine-tuning is applied.

5.2.3 Reflections on the Current Wave of Models

The experimental analysis presented in this section confirms both the opportunities and challenges emerging from the recent wave of LLMs. As discussed in the introductory chapter, one of the core ambitions of this thesis is to explore how grounded language understanding can be enabled within the context of interactive, situated environments, particularly in Human-Robot Interaction (HRI) scenarios. In this light, the proliferation of new LLMs, while difficult to track, offers a growing and appealing pool of candidates for such applications.

Among these, Phi-4 emerges as a particularly promising model. Its lightweight architecture (5.6B parameters), native support for multi-modal inputs, and strong performance make it well-suited for deployment on robotic platforms with limited computational resources. Unlike larger models that require offloading or distributed inference, Phi-4 can be fine-tuned efficiently and operated in constrained environments without sacrificing responsiveness or contextual grounding. Moreover, its ability to integrate visual and textual information na-

tively is crucial for robots acting in dynamic physical environments where understanding user instructions often depends on the surrounding context. The results on the MM-IGLU and MM-IGLU-it datasets show that, with targeted fine-tuning, such models can contribute meaningfully to the development of intelligent agents capable of proactive and cooperative behaviours.

In particular, when compared to the 13B-parameter `LLaMA2Chat`, `Phi-4` achieves competitive or superior results on all tasks and languages evaluated. Despite having less than half the number of parameters, `Phi-4` matches the performance of `LLaMA2Chat` on Italian classification and generation tasks, and even outperforms it on the English generation task. This efficiency-to-performance ratio makes `Phi-4` an excellent candidate for on-board deployment in real robots, especially for in-situ testing of language understanding and clarification mechanisms in realistic environments.

Of course, this experimentation is not exhaustive and further work is needed to fully explore the capabilities of `Phi-4` in real-world scenarios. The results presented here are promising, but they also highlight the need for more extensive evaluation across diverse tasks, modalities, and deployment settings.

5.3 Open Issues and Future Work

Generalisation to New Environments. One of the primary challenges in deploying language models for real-world applications lies in their ability to generalise effectively across diverse and dynamic environments. Current models, including those developed and evaluated in this thesis, have demonstrated strong performance in controlled and predefined settings. However, real-world environments are inherently unpredictable, with varying conditions, noise, and unforeseen complexities that can significantly impact the performance. For instance, a robot operating in a household must be capable of understanding commands in contexts that were not explicitly encountered during training, such as cluttered rooms, varying lighting conditions, or background noise. Achieving robust generalisation requires models to move beyond static, dataset-specific learning paradigms and incorporate continual learning techniques. Future work should focus on enabling models to adapt dynamically to changing environments, learning incrementally from new experiences without catastrophic forgetting. Furthermore, testing in a wider variety of real-world scenarios, including outdoor environments and industrial settings, is necessary to ensure that models maintain reliable performance under diverse conditions.

Multilingual and Low-Resource Language Support. Although the models presented in this thesis have shown promising results in multilingual settings, the primary focus has

been on high-resource languages or those that are well-represented in pre-existing datasets. Multilingual models inherently struggle with low-resource languages, where the available training data is sparse or of low quality. This limitation can hinder the deployment of intelligent systems in regions where such languages are predominantly spoken, thereby reducing their accessibility and usability. Future research should explore advanced transfer learning approaches and semi-supervised methods to extend the language coverage of these models. Techniques such as zero-shot and few-shot learning could be employed to enable models to infer knowledge from high-resource languages and apply it to low-resource counterparts. Additionally, creating new datasets that include underrepresented languages, possibly through community-driven initiatives, would significantly improve model performance in these languages. Expanding `MM-IGLU-it` to support additional low-resource languages is a logical next step in this direction.

Real-Time Performance and Efficiency. One of the fundamental requirements for deploying models on physical robotic platforms is ensuring real-time performance. While significant advancements have been made in parameter-efficient fine-tuning methods, such as LoRA and Q-LoRA, large multimodal models still present substantial computational demands that may exceed the capabilities of embedded systems. This becomes particularly problematic when multiple modalities, such as vision, language, speech, sensors and retrieval, need to be processed concurrently. Future efforts should focus on further optimising resource usage without compromising model accuracy. This could involve the exploration of hardware-software co-design approaches, where both the model architecture and the underlying hardware are jointly optimised to achieve low-latency performance. Additionally, investigating pruning and quantisation techniques for multimodal models, combined with efficient runtime inference algorithms, could help bridge the gap between research-grade models and real-time deployment in embedded systems.

Interactive Learning and Clarification. The introduction and adaptation of `BART-IGLU` in Section 4.2, `LLaVA` and `MiniCPM` in Section 4.3 and `Phi4` in Section 5.2.1 in this thesis represents a significant step toward enabling models to act as active collaborators in human-robot interaction. They enable a collaborative setting for the dialogical interaction with the user. Unlike traditional models that passively respond to user commands, these are capable of asking clarification questions when faced with ambiguous or incomplete information, going back and forth with the user until a mutual satisfaction is reached. Moreover, `Phi4` demonstrates the potential for real-time interaction in dynamic environments, where the model can adapt its responses based on user feedback and contextual cues. Future research should focus on developing models that can learn continuously from user feedback, improving their understanding and adaptability over time. This could be done by the integration of reinforce-

ment learning from human feedback (RLHF) to fine-tune the models in real-world scenarios. Moreover, designing a robust framework for lifelong learning, where the models retain past knowledge while acquiring new information, will be essential for their long-term usability. Such improvements would enable more natural and effective interactions, particularly in complex environments where ambiguity is frequent.

Ethical and Interpretability Concerns. As language models become more integrated into real-world applications, ethical considerations and interpretability become critical. Multilingual and multimodal models may exhibit biases inherited from the data on which they were trained, potentially leading to unfair or discriminatory outcomes when deployed. For example, a robot providing assistance in a multilingual household might prioritise commands in certain languages over others, reflecting underlying biases in the training corpus. Addressing these biases requires not only more balanced datasets but also the development of bias mitigation strategies at both the data and model levels. Interpretability, on the other hand, is vital for building trust in human-robot interactions. Users must be able to understand the reasoning behind a decisions, particularly in high-stakes scenarios such as healthcare or industrial automation. Future work should focus on creating explainability tools that allow users to trace the decision-making process of the model.

5.3.1 A Cognitive Architecture for Multi-Modal Situated Dialogues

The development of a unified cognitive architecture for multi-modal situated dialogues represents a significant step forward in the design of intelligent systems. The motivation behind this proposition stems from the increasing need for models capable of understanding and reasoning across multiple modalities, handling multiple tasks concurrently, and interacting dynamically in real-world environments. Such an architecture could enable future robots to perform complex tasks requiring situational awareness, making them effective collaborators in daily human activities.

A central component of this cognitive architecture is the integration of **Multimodal Large Language Models**, which serve as the foundation for processing and reasoning over diverse inputs such as text, images, and audio. The work conducted in this thesis, particularly in the domain of Visual Question Answering (VQA) and the development of `MM-IGLU`, demonstrates the feasibility of applying Multimodal-LLMs to real-world scenarios. These models have already shown significant capability in interpreting and generating language grounded in visual contexts, making them ideal candidates for inclusion in the proposed framework.

Another critical aspect of the architecture is its **multitasking capability**. As evidenced by the `ExtremITA` model, multitasking allows a single model to perform multiple linguistic tasks simultaneously, improving generalisation and efficiency. In a real-world deployment, a robot equipped with such a multitasking system could interpret various types of instructions, manage context-switching effectively, and handle diverse queries without requiring multiple specialised models.

To further enhance the adaptability and knowledge of the system, the architecture should incorporate **Retrieval-Augmented Generation** (RAG) mechanisms. RAG enables the model to retrieve external information dynamically during interactions, ensuring that the robot remains contextually aware even in situations where pre-existing knowledge is insufficient. For example, in a household scenario, the robot might need to retrieve real-time information about household inventory or schedules to provide accurate assistance. This integration of retrieval capabilities makes the system more robust and versatile, especially in open-ended and rapidly changing environments.

The workflow of the proposed cognitive architecture involves several key stages. First, the system processes the input, which could be a combination of visual, textual, and auditory data. Next, the Multimodal-LLM performs initial reasoning based on the input, identifying the required task and generating a preliminary response. If additional information is necessary, the RAG module retrieves relevant external knowledge, which is then integrated into the final response. The output is subsequently mapped to executable robotic primitives, enabling the robot to perform physical actions or provide informative feedback to the user. Moreover, if more information is required, the system should be able to ask questions in natural language and interacti actively with the user.

Deploying this Cognitive Architecture on robotic platforms opens up numerous possibilities for real-world applications. Potential scenarios include household assistance, where robots could help with daily chores, provide reminders, and offer general support. In industrial settings, the architecture could facilitate human-robot collaboration in complex tasks requiring precision and contextual understanding. Similarly, in healthcare, robots equipped with this system could assist elderly or disabled individuals by understanding their needs and responding appropriately.

Despite its potential, deploying such a comprehensive cognitive architecture on real-world robotic platforms poses **several challenges**. One significant issue is the computational demand of large multimodal models, which may exceed the capacity of current embedded systems. Ensuring real-time performance while maintaining accuracy will require further optimisation of both hardware and software components. Moreover, robustness in dynamic environments remains a key concern, as the system must be able to handle unexpected

changes and noisy inputs effectively. Phi4, explored in 5.2, represents a promising step in this direction, but further research is needed to ensure that the architecture can adapt to a wide range of real-world scenarios. Finally, ethical considerations must be addressed, particularly regarding bias and interpretability. As the architecture relies on large language models trained on diverse datasets, ensuring fairness and transparency in decision-making processes is crucial. Developing tools for bias detection and mitigation, as well as providing users with insights into the reasoning capabilities, will be essential for building trust in human-robot interactions.

The proposed cognitive architecture for multi-modal situated dialogues represents a significant advancement in human-robot collaboration. By integrating Multimodal-LLMs, multitasking capabilities, and RAG mechanisms, the architecture promises to transform robots into proactive, intelligent collaborators capable of assisting humans in a wide range of tasks. The work presented in this thesis lays a solid foundation for this future direction, opening up new possibilities for further research in the future.

Bibliography

[1] Zahra Abbasiantaeb, Yifei Yuan, Evangelos Kanoulas, and Mohammad Aliannejadi. Let the llms talk: Simulating human-to-human conversational qa via zero-shot llm-to-llm interactions. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 8–17, 2024.

[2] Josh Abramson, Arun Ahuja, Arthur Brussee, Federico Carnevale, Mary Cassin, Stephen Clark, Andrew Dudzik, Petko Georgiev, Aurelia Guy, Tim Harley, Felix Hill, Alden Hung, Zachary Kenton, Jessica Landon, Timothy P. Lillicrap, Kory W. Mathewson, Alistair Muldal, Adam Santoro, Nikolay Savinov, Vikrant Varma, Greg Wayne, Nathaniel Wong, Chen Yan, and Rui Zhu. Imitating interactive intelligence. *CoRR*, abs/2012.05672, 2020.

[3] Josh Abramson, Arun Ahuja, Federico Carnevale, Petko Georgiev, Alex Goldin, Alden Hung, Jessica Landon, Timothy Lillicrap, Alistair Muldal, Blake Richards, Adam Santoro, Tamara von Glehn, Greg Wayne, Nathaniel Wong, and Chen Yan. Evaluating multimodal interactive agents, 2022.

[4] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022.

[5] Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. Convai3: Generating clarifying questions for open-domain dialogue systems (clariq). *CoRR*, abs/2009.11352, 2020.

[6] Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. Building and evaluating open-domain dialogue corpora with clarifying ques-

tions. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4473–4484, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[7] James Allen, Elisabeth André, Philip R. Cohen, Dilek Hakkani-Tür, Ronald Kaplan, Oliver Lemon, and David Traum. *Challenge discussion: advancing multimodal dialogue*, page 191217. Association for Computing Machinery and Morgan & Claypool, 2019.

[8] James F. Allen and Mark G. Core. Damsl: Dialog act markup in several layers (draft 2.1). Technical report, Multiparty Discourse Group, Discourse Research Initiative, September 1997.

[9] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. The falcon series of open language models, 2023.

[10] Begoña Altuna, Goutham Karunakaran, Alberto Lavelli, Bernardo Magnini, Manuela Speranza, and Roberto Zanoli. Clinkart at evalita 2023: Overview of the task on linking a lab result to its test event in the clinical domain. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[11] Chiara Alzetta, Dominique Brunato, Felice Delll'Orletta, Alessio Miaschi, Kenji Sagae, Claudia H. Sánchez-Gutiérrez, and Giulia Venturi. Langlearn at evalita 2023: Overview of the language learning development task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[12] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086, 2018.

[13] Anthropic. Claude 3.7 sonnet system card. https://www.anthropic.com/claude-3-7-sonnet-system-card, February 2025.

[14] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

[15] Oscar Araque, Simona Frenda, Rachele Sprugnoli, Debora Nozza, and Viviana Patti. Emit at EVALITA 2023: Overview of the categorical emotion detection in italian social media task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[16] Mauricio Fadel Argerich and Marta Patiño-Martínez. Measuring and improving the energy efficiency of large language models inference. *IEEE Access*, 12:80194–80207, 2024.

[17] Gemini Team at Google. Gemini: A family of highly capable multimodal models, 2024.

[18] Jesse Atuhurra. Large language models for human-robot interaction: Opportunities and risks, 2024.

[19] J.L. Austin. *How to Do Things with Words*. Clarendon Press, Oxford, 1962.

[20] Andrea Bacciu, Giovanni Trappolini, Andrea Santilli, Emanuele Rodolà, and Fabrizio Silvestri. Fauno: The italian large language model that will leave you senza parole! *arXiv preprint arXiv:2306.14457*, 2023.

[21] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[22] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. Training a helpful and harmless assistant with reinforcement learning from human feedback, 2022.

[23] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90, Montreal, Quebec, Canada, August 1998. Association for Computational Linguistics.

[24] Charles Bally and Léopold Gautier, editors. *Recueil des publications scientifiques de F. de Saussure*. Librairie Payot & C^{ie}, Lausanne and Geneva, 1922.

[25] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443, 2018.

[26] Ermanno Bartoli, Fethiye Irmak Dogan, and Iolanda Leite. Streaming network for continual learning of object relocations under household context drifts, 2024.

[27] Pierpaolo Basile, Pierluigi Cassotti, Marco Polignano, Lucia Siciliani, and Giovanni Semeraro. On the impact of language adaptation for large language models: A case study for the italian language using only open resources. In *Proceedings of CLiC-it 2023*, 2023.

[28] Pierpaolo Basile, Elio Musacchio, Marco Polignano, Lucia Siciliani, Giuseppe Fiameni, and Giovanni Semeraro. Llamantino: Llama 2 models for effective text generation in italian language, 2023.

[29] Valerio Basile and Malvina Nissim. Sentiment analysis on Italian tweets. In Alexandra Balahur, Erik van der Goot, and Andres Montoyo, editors, *Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 100–107, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[30] Emanuele Bastianelli, Danilo Croce, Roberto Basili, and Daniele Nardi. UNITOR-HMM-TK: Structured kernel-based learning for spatial role labeling. In Suresh Manandhar and Deniz Yuret, editors, *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 573–579, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics.

[31] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[32] Yonatan Bisk, Ari Holtzman, Jesse Thomason, et al. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

[33] Rexhina Blloshmi, Simone Conia, Rocco Tripodi, and Roberto Navigli. Generating senses and roles: An end-to-end model for dependency- and span-based semantic role labeling. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 3786–3793. ijcai.org, 2021.

[34] Reinhard Blutner. Some aspects of optimality in natural language interpretation. *Journal of semantics*, 17(3):189–216, 2000.

[35] Richard A Bolt. "put-that-there" voice and gesture at the graphics interface. In *Proceedings of the 7th annual conference on Computer graphics and interactive techniques*, pages 262–270, 1980.

[36] Alessandro Bondielli, Pietro Dell'Oglio, Alessandro Lenci, Francesco Marcelloni, Lucia C. Passaro, and Marco Sabbatini. Multi-fake-detective at evalita 2023: Overview of the multimodal fake news detection and verification task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[37] Federico Borazio, Claudiu Daniel Hromei, Daniele Margiotta, Danilo Croce, and Roberto Basili. MM-IGLU-IT: Multi-modal interactive grounded language understanding in italian. In *AIxIA 2024 - Advances in Artificial Intelligence - XXIIIrd International Conference of the Italian Association for Artificial Intelligence, AIxIA 2024, Bolzano, Italy, November 25-28, 2024, Proceedings*, Bolzano, Italia, November 2024.

[38] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1):14–23, 1986.

[39] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[40] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[41] Dominique Brunato, Davide Colla, Felice Dell'Orletta, Irene Dini, Daniele Paolo Radicioni, and Andrea Amelio Ravelli. Discotex at evalita 2023: Overview of the assessing discourse coherence in italian texts task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[42] Deng Cai and Yansong He. A full end-to-end semantic role labeler, syntax-agnostic over syntax-aware? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 275–286, 2018.

[43] Erik Cambria. *Understanding Natural Language Understanding.* Springer, 2024.

[44] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL)*, pages 152–164. Association for Computational Linguistics, 2005.

[45] Pierluigi Cassotti, Lucia Siciliani, Lucia Passaro, Maristella Gatto, and Pierpaolo Basile. Wic-ita at evalita2023: Overview of the evalita2023 word-in-context for italian task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[46] Giuseppe Castellucci, Simone Filice, Danilo Croce, and Roberto Basili. UNITOR: Aspect based sentiment analysis with structured learning. In Preslav Nakov and Torsten Zesch, editors, *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 761–767, Dublin, Ireland, August 2014. Association for Computational Linguistics.

[47] Wanxiang Che and Ting Liu. Jointly modeling wsd and srl with markov logic. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 161–169, 2010.

[48] Danqi Chen and Christopher Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October 2014. Association for Computational Linguistics.

[49] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. *arXiv preprint arXiv:2312.14238*, 2023.

[50] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023.

[51] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches, 2014.

[52] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

[53] Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307, 2017.

[54] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Y. Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416, 2022.

[55] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models, 2022.

[56] Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. *arXiv preprint arXiv:2002.05867*, 2020.

[57] Trevor Cohn and Phil Blunsom. Semantic role labelling with tree conditional random fields. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 169–172, 2005.

[58] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics.

[59] Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single $&!#* vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[60] Arthur L Corbin. Interpretation of words and the parol evidence rule. *Cornell LQ*, 50:161, 1964.

[61] Mark G. Core and James F. Allen. Coding dialogues with the damsl annotation scheme. In David Traum, editor, *Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35, Menlo Park, CA, 1997. American Association for Artificial Intelligence.

[62] Corinna Cortes and Vladimir Vapnik. Support-vector networks. In *Machine Learning*, volume 20, pages 273–297. Springer, 1995.

[63] Lorenzo Cristofori, Claudiu Daniel Hromei, Francesco Scotto di Luzio, Christian Tamantini, Francesca Cordella, Danilo Croce, Loredana Zollo, and Roberto Basili. Heal9000: an intelligent rehabilitation robot. In Peter Lucas and Fabio Stella, editors, *Proceedings of the Workshop on Towards Smarter Health Care: Can Artificial Intelligence Help? co-located with 20th International Conference of the Italian Association for Artificial Intelligence (AIxIA2021), Anywhere, November 29th, 2021*, volume 3060 of *CEUR Workshop Proceedings*, pages 29–41. CEUR-WS.org, 2021.

[64] Danilo Croce, Alessandro Moschitti, and Roberto Basili. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1034–1046, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[65] Danilo Croce, Lucia C. Passaro, Alessandro Lenci, and Roberto Basili. Gqa-it: Italian question answering on image scene graphs. In Elisabetta Fersini, Marco Passarotti, and Viviana Patti, editors, *Proceedings of the Eighth Italian Conference on Computational Linguistics, CLiC-it 2021, Milan, Italy, January 26-28, 2022*, volume 3033 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

[66] Gergely Csibra and György Gergely. Natural pedagogy. *Trends in Cognitive Sciences*, 13(4):148–153, 2009.

[67] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 423–429, Barcelona, Spain, July 2004.

[68] Marie-Catherine de Marneffe, Timothy Dozat, Natalia Silveira, Katri Haverinen, Filip Ginter, Joakim Nivre, and Christopher D. Manning. Universal Stanford dependencies: A cross-linguistic typology. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA).

[69] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*, 2023.

[70] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the NAACL 2019*, pages 4171–4186, 2019.

[71] Francesco Scotto di Luzio, Christian Tamantini, Francesca Cordella, Claudiu Daniel Hromei, Lorenzo Cristofori, Danilo Croce, Roberto Basili, and Loredana Zollo. Heal9000: an intelligent rehabilitation robot for physical and cognitive interaction, 2021.

[72] Yuan Ding and Martha Palmer. Improving statistical machine translation using syntax. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 636–643, 2005.

[73] Guozhu Dong and Huan Liu. *Feature engineering for machine learning and data analytics*. CRC press, 2018.

[74] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Jingyuan Ma, Rui Li, Heming Xia, Jingjing Xu, Zhiyong Wu, Tianyu Liu, et al. A survey on in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

[75] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[76] Timothy Dozat and Christopher D. Manning. Deep biaffine attention for neural dependency parsing. *CoRR*, abs/1611.01734, 2016.

[77] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[78] Myroslava O. Dzikovska, James F. Allen, and Mary D. Swift. Linking semantic and knowledge representations in a multi-domain dialogue system. *Journal of Logic and Computation*, 18(3):405–430, 12 2007.

[79] Paul Ekman and Wallace V Friesen. Constants across cultures in the face and emotion. *Journal of Personality and Social Psychology*, 17(2):124–129, 1971.

[80] Nicholas Evans and Stephen C. Levinson. The myth of language universals: Language diversity and its importance for cognitive science. *Behavioral and Brain Sciences*, 32(5):429448, 2009.

[81] Zhihao Fan, Yeyun Gong, Zhongyu Wei, Siyuan Wang, Yameng Huang, Jian Jiao, Xuanjing Huang, Nan Duan, and Ruofei Zhang. An enhanced knowledge injection model for commonsense generation. *arXiv preprint arXiv:2012.00366*, 2020.

[82] Charles J. Fillmore. The case for case. *Universals in Linguistic Theory*, pages 1–88, 1968.

[83] Charles J. Fillmore. Frames and the semantics of understanding. *Quaderni di Semantica*, 6(2):222–254, 1985.

[84] Lyn Frazier. *On sentence interpretation*, volume 22. Springer Science & Business Media, 1999.

[85] Giovanni Gafà, Francesco Cutugno, and Marco Venuti. Emotivita at EVALITA2023: Overview of the dimensional and multidimensional emotion analysis task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[86] Chuang Gan, Jeremy Schwartz, Seth Alter, Damian Mrowca, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, et al. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*, 2020.

[87] Michael Garman. *Psycholinguistics*. Cambridge University Press, 1990.

[88] Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational linguistics*, 28(3):245–288, 2002.

[89] Michael A. Goodrich and Alan C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in HumanComputer Interaction*, 1(3):203–275, 2008.

[90] Google DeepMind. Gemini 2.5: Our most intelligent ai model. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/, March 2025.

[91] Louis A. Gottschalk. Quantification and psychological indicators of emotions: The content analysis of speech and other objective measures of psychological states. *The International Journal of Psychiatry in Medicine*, 5(4):587–596, 1974.

[92] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou,

Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan,

Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models, 2024.

[93] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need, 2023.

[94] Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lofti A Zadeh. *Feature extraction: foundations and applications*, volume 207. Springer, 2008.

[95] Aria Haghighi, Kristina Toutanova, and Christopher D Manning. A joint model for semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 173–176, 2005.

[96] Sherzod Hakimov, Soufian Jebbara, and Philipp Cimiano. Amuse: Multilingual semantic parsing for question answering over linked data, 2018.

[97] Denis Hamad, Ludovic Macaire, and Philippe Biela. Constraint scores for semi-supervised feature selection: A comparative study. *Pattern Recognition Letters*, 32(5):656–665, 2011.

[98] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.

[99] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[100] Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. Deep semantic role labeling: What works and whats next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 473–483, 2017.

[101] Jeff Heaton. An empirical analysis of feature engineering for predictive modeling. In *SoutheastCon 2016*, pages 1–6. IEEE, 2016.

[102] Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

[103] Felix Hill, Stephen Clark, Karl Moritz Hermann, and Phil Blunsom. Understanding grounded language learning agents. *arXiv preprint arXiv:1605.06572*, 2016.

[104] S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[105] Wenyi Hong, Weihan Wang, Ming Ding, Wenmeng Yu, Qingsong Lv, Yan Wang, Yean Cheng, Shiyu Huang, Junhui Ji, Zhao Xue, et al. Cogvlm2: Visual language models for image and video understanding, 2024.

[106] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxuan Zhang, Juanzi Li, Bin Xu, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2023.

[107] Claudiu Daniel Hromei, Federico Borazio, Andrea Sensi, Elisa Passone, Danilo Croce, and Roberto Basili. Training multi-modal llms through dialogue planning for hri. In *SUBMITTED AT: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, Vienna, Austria, August 2025. Association for Computational Linguistics.

[108] Claudiu Daniel Hromei, Lorenzo Cristofori, Danilo Croce, and Roberto Basili. Embedding contextual information in seq2seq models for grounded semantic role labeling. In Agostino Dovier, Angelo Montanari, and Andrea Orlandini, editors, *AIxIA 2022 - Advances in Artificial Intelligence - XXIst International Conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, November 28 - December 2, 2022, Proceedings*, volume 13796 of *Lecture Notes in Computer Science*, pages 472–485. Springer, 2022.

[109] Claudiu Daniel Hromei, Danilo Croce, Valerio Basile, and Roberto Basili. ExtremITA at EVALITA 2023: Multi-Task Sustainable Scaling to Large Language Models at its Extreme. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[110] Claudiu Daniel Hromei, Danilo Croce, Valerio Basile, and Roberto Basili. Scaling large language models to the extreme: Neural semantic processing of multiple tasks in italian. In Roberto Basili, Domenico Lembo, Carla Limongelli, and Andrea Orlandini, editors, *AIxIA 2023 - Advances in Artificial Intelligence - XXIInd International Conference of the Italian Association for Artificial Intelligence, AIxIA 2023, Rome, Italy, November 6-9, 2023, Proceedings*, volume 14318 of *Lecture Notes in Computer Science*, pages 172–186. Springer, 2023.

[111] Claudiu Daniel Hromei, Danilo Croce, and Roberto Basili. Grounding end-to-end architectures for semantic role labeling in human robot interaction. In Debora Nozza, Lucia C. Passaro, and Marco Polignano, editors, *Proceedings of the Sixth Workshop on Natural Language for Artificial Intelligence (NL4AI 2022) co-located with 21th International Conference of the Italian Association for Artificial Intelligence (AI*IA 2022), Udine, November 30th, 2022*, volume 3287 of *CEUR Workshop Proceedings*, pages 24–38. CEUR-WS.org, 2022.

[112] Claudiu Daniel Hromei, Danilo Croce, and Roberto Basili. End-to-end dependency parsing via auto-regressive large language model. In Federico Boschetti, Gianluca E. Lebani, Bernardo Magnini, and Nicole Novielli, editors, *Proceedings of the 9th Italian Conference on Computational Linguistics, Venice, Italy, November 30 - December 2, 2023*, volume 3596 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

[113] Claudiu Daniel Hromei, Danilo Croce, and Roberto Basili. Grounding end-to-end pre-trained architectures for semantic role labeling in multiple languages. *Intelligenza Artificiale*, 17(2):173–191, 2023.

[114] Claudiu Daniel Hromei, Danilo Croce, and Roberto Basili. U-DepPLLaMA: Universal dependency parsing via auto-regressive large language models. *International Journal of Computational Linguistics*, 2024.

[115] Claudiu Daniel Hromei, Danilo Croce, Rodolfo Delmonte, and Roberto Basili. La non canonica l'hai studiata? exploring llms and sentence canonicity in italian. In Felice Dell'Orletta, Alessandro Lenci, Simonetta Montemagni, and Rachele Sprugnoli, editors, *Proceedings of the 10th Italian Conference on Computational Linguistics, Pisa, Italy, December 4 - 6, 2024*, volume 3878 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2024.

[116] Claudiu Daniel Hromei, Alessia Forciniti, Daniele Margiotta, and Stefano Locati. Automatic emotion analysis in movies: Matteo garrone's dogman as a case study. In *International Conference on Deep Learning Theory and Applications*, pages 76–94. Springer, 2024.

[117] Claudiu Daniel Hromei, Daniele Margiotta, Danilo Croce, and Roberto Basili. An end-to-end transformer-based model for interactive grounded language understanding. In Elisa Bassignana, Dominique Brunato, Marco Polignano, and Alan Ramponi, editors, *Proceedings of the Seventh Workshop on Natural Language for Artificial Intelligence (NL4AI 2023) co-located with 22th International Conference of the Italian Association*

for Artificial Intelligence (AIxIA 2023), Rome, Italy, November 6th-7th, 2023, volume 3551 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

[118] Claudiu Daniel Hromei, Daniele Margiotta, Danilo Croce, and Roberto Basili. MM-IGLU: Multi-modal interactive grounded language understanding. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 11440–11451, Torino, Italia, May 2024. ELRA and ICCL.

[119] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2021.

[120] Siyuan Hu, Mingyu Ouyang, Difei Gao, and Mike Zheng Shou. The dawn of gui agent: A preliminary case study with claude 3.5 computer use, 2024.

[121] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[122] Erik Johannes Husom, Arda Goknil, Lwin Khin Shar, and Sagar Sen. The price of prompting: Profiling energy use in large language models inference, 2024.

[123] Shima Imani, Liang Du, and Harsh Shrivastava. MathPrompter: Mathematical reasoning using large language models. In Sunayana Sitaram, Beata Beigman Klebanov, and Jason D Williams, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 37–42, Toronto, Canada, July 2023. Association for Computational Linguistics.

[124] Ray S Jackendoff. Language, consciousness, culture: Essays on mental structure. *MIT Press*, 2009.

[125] Mojan Javaheripi and Sébastien Bubeck. Phi-2: The surprising power of small language models. https://www.microsoft.com/en-us/research/blog/phi-2-the-surprising-power-of-small-language-models/, December 2023.

[126] Prashant Jayannavar, Anjali Narayan-Chen, and Julia Hockenmaier. Learning to execute instructions in a minecraft dialogue. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the*

Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020, pages 2589–2602. Association for Computational Linguistics, 2020.

[127] Jianbo Jiang, Zhenyi Wu, Yu Su, and Wenshuo Cheng. Dependency-based multi-granularity code retrieval. In *2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS)*, pages 715–720, 2023.

[128] Daniel Jurafsky and James H Martin. Speech and language processing. *Prentice Hall*, 2023.

[129] Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, 2011.

[130] Aditya Kalyanpur, Or Biran, Tom Breloff, Jennifer Chu-Carroll, Ariel Diertani, Owen Rambow, and Mark Sammons. Open-domain frame semantic parsing using transformers, 2020.

[131] Callie Y Kim, Christine P Lee, and Bilge Mutlu. Understanding large-language model (llm)-powered human-robot interaction. In *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*, pages 371–380, 2024.

[132] Julia Kiseleva, Ziming Li, Mohammad Aliannejadi, Shrestha Mohanty, Maartje ter Hoeve, Mikhail Burtsev, Alexey Skrynnik, Artem Zholus, Aleksandr Panov, Kavya Srinet, et al. Interactive grounded language understanding in a collaborative environment: Iglu 2021. In *NeurIPS 2021 Competitions and Demonstrations Track*, pages 146–161. PMLR, 2022.

[133] Julia Kiseleva, Alexey Skrynnik, Artem Zholus, Shrestha Mohanty, Negar Arabzadeh, Marc-Alexandre Côté, Mohammad Aliannejadi, Milagro Teruel, Ziming Li, Mikhail Burtsev, Maartje ter Hoeve, Zoya Volovikova, Aleksandr Panov, Yuxuan Sun, Kavya Srinet, Arthur Szlam, and Ahmed Awadallah. Iglu 2022: Interactive grounded language understanding in a collaborative environment at neurips 2022, 2022.

[134] Mark L Knapp, Judith A Hall, and Terrence G Horgan. *Nonverbal communication in human interaction*. Wadsworth Publishing, 2013.

[135] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai, 2022.

[136] Dan Kondratyuk and Milan Straka. 75 languages, 1 model: Parsing Universal Dependencies universally. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795, Hong Kong, China, November 2019. Association for Computational Linguistics.

[137] Daniel Kondratyuk. 75 languages, 1 model: Parsing universal dependencies universally. *CoRR*, abs/1904.02099, 2019.

[138] Alkis Koudounas, Flavio Giobergia, Irene Benedetto, Simone Monaco, Luca Cagliero, Daniele Apiletti, Elena Baralis, et al. baρtti at geolingit: Beyond boundaries, enhancing geolocation prediction and dialect classification on social media in italy. In *CEUR Workshop Proceedings*, 2023.

[139] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105. Curran Associates, Inc., 2012.

[140] Sandra Kübler, Ryan McDonald, and Joakim Nivre. *Dependency Parsing*. Springer Cham, 2009.

[141] Marco Kuhlmann and Joakim Nivre. Transition-based techniques for non-projective dependency parsing. *Northern European Journal of Language Technology*, 2, 10 2010.

[142] Moreno La Quatra and Luca Cagliero. Bart-it: An efficient sequence-to-sequence model for italian text summarization. *Future Internet*, 15(1), 2023.

[143] John Lafferty, Andrew McCallum, Fernando Pereira, et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Icml*, volume 1, page 3. Williamstown, MA, 2001.

[144] Huiyuan Lai, Jiali Mao, Antonio Toral, and Malvina Nissim. Human judgement as a compass to navigate automatic metrics for formality transfer. In Anya Belz, Maja Popović, Ehud Reiter, and Anastasia Shimorina, editors, *Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval)*, pages 102–115, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[145] Mirko Lai, Fabio Celli, Alan Ramponi, Sara Tonelli, Cristina Bosco, and Viviana Patti. Haspeede3 at evalita 2023: Overview of the political and religious hate speech

detection task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[146] Mirko Lai, Stefano Menini, Marco Polignano, Valentina Russo, Rachele Sprugnoli, and Giulia Venturi. EVALITA 2023: Overview of the 8th evaluation campaign of natural language processing and speech tools for italian. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[147] Mirko Lai, Stefano Menini, Marco Polignano, Valentina Russo, Rachele Sprugnoli, and Giulia Venturi, editors. *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023), Parma, Italy, September 7th-8th, 2023*, volume 3473 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

[148] Tin Lai. A review on visual-slam: Advancements from geometric modelling to learning-based semantic scene understanding using multi-modal sensor fusion. *Sensors*, 22(19):7265, 2022.

[149] Anne Lauscher, Olga Majewska, Leonardo F. R. Ribeiro, Iryna Gurevych, Nikolai Rozanov, and Goran Glavas. Common sense or world knowledge? investigating adapter-based knowledge injection into pretrained transformers. *CoRR*, abs/2005.11787, 2020.

[150] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[151] Stephen C. Levinson, Tom M. Mitchell, Simon Garrod, John E. Laird, and Kenneth R. Koedinger. *Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions*. MIT Press, 2019.

[152] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[153] Chengshu Li, Fei Xia, Roberto Martín-Martín, Michael Lingelbach, Sanjana Srivastava, Bokui Shen, Kent Elliott Vainio, Cem Gokmen, Gokul Dharan, Tanish Jain, Andrey

Kurenkov, Karen Liu, Hyowon Gweon, Jiajun Wu, Li Fei-Fei, and Silvio Savarese. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 455–465. PMLR, 08–11 Nov 2022.

[154] Hongkang Li, Meng Wang, Songtao Lu, Xiaodong Cui, and Pin-Yu Chen. How do nonlinear transformers learn and generalize in in-context learning? *arXiv preprint arXiv:2402.15607*, 2024.

[155] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation, 2022.

[156] Yunzhu Li, Toru Lin, Kexin Yi, Daniel Bear, Daniel Yamins, Jiajun Wu, Joshua Tenenbaum, and Antonio Torralba. Visual grounding of learned physical models. In *International conference on machine learning*, pages 5927–5936. PMLR, 2020.

[157] Zuchao Li, Jiaxun Cai, Shexia He, and Hai Zhao. Seq2seq dependency parsing. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3203–3214, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

[158] KyungTae Lim, Niko Partanen, and Thierry Poibeau. Multilingual dependency parsing for low-resource languages: Case studies on north saami and komi-zyrian. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[159] Zhihong Lin, Donghao Zhang, Qingyi Tao, Danli Shi, Gholamreza Haffari, Qi Wu, Mingguang He, and Zongyuan Ge. Medical visual question answering: A survey. *Artificial Intelligence in Medicine*, 143:102611, September 2023.

[160] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 34892–34916. Curran Associates, Inc., 2023.

[161] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[162] Xiaoqiang Liu and Yu Zhou. A semantic role labeling cascade framework for natural language understanding. *Expert Systems with Applications*, 136:252–265, 2019.

[163] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[164] Dylan P Losey, Craig G McDonald, Edoardo Battaglia, and Marcia K O'Malley. A review of intent detection, arbitration, and communication aspects of shared control for physical human–robot interaction. *Applied Mechanics Reviews*, 70(1):010804, 2018.

[165] Man Luo, Xin Xu, Yue Liu, Panupong Pasupat, and Mehran Kazemi. In-context learning with retrieved demonstrations for language models: A survey. *arXiv preprint arXiv:2401.11624*, 2024.

[166] Gary Lupyan. The centrality of language in human cognition. *Language Learning*, 66(3):516–553, 2016.

[167] Andréa Macario Barros, Maugan Michel, Yoann Moline, Gwenolé Corre, and Frédérick Carrel. A comprehensive survey of visual slam algorithms. *Robotics*, 11(1), 2022.

[168] Brielen Madureira and David Schlangen. Instruction clarification requests in multimodal collaborative dialogue games: Tasks, and an analysis of the codraw dataset. In Andreas Vlachos and Isabelle Augenstein, editors, *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023*, pages 2295–2311. Association for Computational Linguistics, 2023.

[169] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, and Sayak Paul. Peft: State-of-the-art parameter-efficient fine-tuning methods. https://github.com/huggingface/peft, 2022.

[170] Christopher D Manning et al. The stanford corenlp natural language processing toolkit. In *Proceedings of the ACL System Demonstrations*, pages 55–60, 2014.

[171] Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019.

[172] Daniel Marcu and William Wong. Dependency treelet translation: Syntactically informed phrasal smt. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 33–40, 2005.

[173] Ryan McDonald and Joakim Nivre. Characterizing the errors of data-driven dependency parsing models. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 122–131, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[174] Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. Universal Dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[175] Shikib Mehri, Mihail Eric, and Dilek Hakkani-Tur. Dialoglue: A natural language understanding benchmark for task-oriented dialogue, 2020.

[176] Andrew N. Meltzoff. Imitation, objects, tools, and the rudiments of language in human ontogeny. *Human Evolution*, 3(1-2):45–64, 1988.

[177] Paola Merlo and Gabriele Musillo. Semantic parsing for high-precision semantic role labelling. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 1–8, 2008.

[178] Meta AI. Introducing llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/, July 2024.

[179] Meta AI. Llama 3.2: Revolutionizing edge ai and vision with open models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/, September 2024.

[180] Meta AI. Llama 3.3 model card. https://www.llama.com/docs/model-cards-and-prompt-formats/llama3_3/, December 2024.

[181] Meta AI. The llama 4 herd: The beginning of a new era of natively multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, April 2025.

[182] D. Metzler, T. Noreault, L. Richey, and Patrick Heidorn. Dependency parsing for information retrieval. pages 313–324, 01 1984.

[183] Microsoft, :, Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, Dong Chen, Dongdong Chen, Junkun Chen, Weizhu Chen, Yen-Chun Chen, Yi ling Chen, Qi Dai, Xiyang Dai, Ruchao Fan, Mei Gao, Min Gao, Amit Garg, Abhishek Goswami, Junheng Hao, Amr Hendy, Yuxuan Hu, Xin Jin, Mahmoud Khademi, Dongwoo Kim, Young Jin Kim, Gina Lee, Jinyu Li, Yunsheng Li, Chen Liang, Xihui Lin, Zeqi Lin, Mengchen Liu, Yang Liu, Gilsinia Lopez, Chong Luo, Piyush Madan, Vadim Mazalov, Arindam Mitra, Ali Mousavi, Anh Nguyen, Jing Pan, Daniel Perez-Becker, Jacob Platin, Thomas Portet, Kai Qiu, Bo Ren, Liliang Ren, Sambuddha Roy, Ning Shang, Yelong Shen, Saksham Singhal, Subhojit Som, Xia Song, Tetyana Sych, Praneetha Vaddamanu, Shuohang Wang, Yiming Wang, Zhenghao Wang, Haibin Wu, Haoran Xu, Weijian Xu, Yifan Yang, Ziyi Yang, Donghan Yu, Ishmam Zabir, Jianwen Zhang, Li Lyna Zhang, Yunan Zhang, and Xiren Zhou. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras, 2025.

[184] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013.

[185] Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. Metaicl: Learning to learn in context. *arXiv preprint arXiv:2110.15943*, 2021.

[186] Mistral AI team. Au large. https://mistral.ai/news/mistral-large/, February 2024.

[187] Shrestha Mohanty, Negar Arabzadeh, Milagro Teruel, Yuxuan Sun, Artem Zholus, Alexey Skrynnik, Mikhail Burtsev, Kavya Srinet, Aleksandr Panov, Arthur Szlam, Marc-Alexandre Côté, and Julia Kiseleva. Collecting interactive multi-modal datasets for grounded language understanding. *CoRR*, abs/2211.06552, 2022.

[188] Shrestha Mohanty, Negar Arabzadeh, Milagro Teruel, Yuxuan Sun, Artem Zholus, Alexey Skrynnik, Mikhail Burtsev, Kavya Srinet, Aleksandr Panov, Arthur Szlam, et al. Collecting interactive multi-modal datasets for grounded language understanding. *arXiv preprint arXiv:2211.06552*, 2022.

[189] Will Monroe, Robert XD Hawkins, Noah D Goodman, and Christopher Potts. Colors in context: A pragmatic neural model for grounded language understanding. *Transactions of the Association for Computational Linguistics*, 5:325–338, 2017.

[190] Alessandro Moschitti, Ana-Maria Giuglea, Bonaventura Coppola, and Roberto Basili. Hierarchical semantic role labeling. In *Proceedings of the ninth conference on computational natural language learning (CoNLL-2005)*, pages 201–204, 2005.

[191] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[192] Anjali Narayan-Chen, Colin Graber, Mayukh Das, Md Rakibul Islam, Soham Dan, Sriraam Natarajan, Janardhan Rao Doppa, Julia Hockenmaier, Martha Palmer, and Dan Roth. Towards problem solving agents that communicate and learn. In *Proceedings of the First Workshop on Language Grounding for Robotics*, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[193] Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. Collaborative dialogue in minecraft. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5405–5415. Association for Computational Linguistics, 2019.

[194] National Research Council. *How People Learn: Brain, Mind, Experience, and School: Expanded Edition*. National Academies Press, Washington, DC, August 1999.

[195] Malvina Nissim, Rik van Noord, and Rob van der Goot. Fair is better than sensational: Man is to doctor as woman is to doctor. *Computational Linguistics*, 46(2):487–497, 2020.

[196] Joakim Nivre. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553, 2008.

[197] Debora Nozza, Alessandra Teresa Cignarella, Greta Damo, Tommaso Caselli, and Viviana Patti. Hodi at evalita 2023: Overview of the homotransphobia detection in italian task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[198] Linda Onnasch and Eileen Roesler. A taxonomy to structure and analyze humanrobot interaction. *International Journal of Social Robotics*, 13:833–849, 2021.

[199] OpenAI. Gpt-4 technical report, 2023.

[200] OpenAI. Gpt-4o system card. https://openai.com/index/gpt-4o-system-card/, 2024.

[201] OpenAI. Introducing gpt-4.1 in the api. https://openai.com/index/gpt-4-1/, April 2025.

[202] OpenAI. Introducing gpt-4.5. https://openai.com/index/introducing-gpt-4-5/, February 2025.

[203] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

[204] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

[205] Alessio Palmero Aprosio and Teresa Paccosi. Nermud at evalita 2023: Overview of the named-entities recognition on multi-domain documents task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[206] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics.

[207] Ian Perera, James Allen, Lucian Galescu, Choh Man Teng, Mark Burstein, Scott Friedman, David McDonald, and Jeffrey Rye. Natural language dialogue for building and learning models and structures. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1), Feb. 2017.

[208] Ian Perera, James Allen, Choh Man Teng, and Lucian Galescu. A situated dialogue system for learning structural concepts in blocks world. In Kazunori Komatani, Diane

Litman, Kai Yu, Alex Papangelis, Lawrence Cavedon, and Mikio Nakano, editors, *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 89–98, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[209] Robert Plutchik and Henry Kellerman. Theories of emotion. 1, 1980.

[210] Michael Polanyi. *The Tacit Dimension*. University of Chicago Press, 1966.

[211] Marco Polignano, Pierpaolo Basile, Marco de Gemmis, Giovanni Semeraro, and Valerio Basile. AlBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets. In *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*, volume 2481. CEUR, 2019.

[212] Marco Polignano, Pierpaolo Basile, and Giovanni Semeraro. Advanced natural-based interaction for the italian language: Llamantino-3-anita, 2024.

[213] Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H Martin, and Daniel Jurafsky. Support vector learning for semantic argument classification. *Machine Learning*, 60:11–39, 2005.

[214] Vasin Punyakanok, Dan Roth, Wen-tau Yih, and Dav Zimak. Semantic role labeling via integer linear programming inference. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 1346–1352, 2004.

[215] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *CoRR*, abs/2103.00020, 2021.

[216] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training, 2018.

[217] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[218] Alan Ramponi and Camilla Casula. GeoLingIt at EVALITA 2023: Overview of the geolocation of linguistic variation in Italy task. In *Proceedings of the Eighth Evaluation Campaign of Natural Language Processing and Speech Tools for Italian. Final Workshop (EVALITA 2023)*, Parma, Italy, September 2023. CEUR.org.

[219] Frank Reichartz, Hannes Korte, and Gerhard Paass. Dependency tree kernels for relation extraction from natural language text. In Wray Buntine, Marko Grobelnik, Dunja Mladenić, and John Shawe-Taylor, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 270–285, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[220] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[221] E. V. Rieu and Peter Jones, editors. *The Iliad*. Penguin Classics, London, 2008. Hephaestus' automata are mentioned in Book 18.

[222] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 817–834. Springer, 2016.

[223] Subhro Roy and Dan Roth. Solving general arithmetic word problems, 2016.

[224] Daniel Russo, Salud María Jiménez-Zafra, José Antonio García-Díaz, Tommaso Caselli, Marco Guerini, L. Alfonso Ureña-López, and Rafael Valencia-García. Overview of politicit2023@evalita: Political ideology detection in italian texts. 2023.

[225] Giuseppe Russo, Niklas Stoehr, and Manoel Horta Ribeiro. Acti at EVALITA 2023: Overview of the conspiracy theory identification task. *arXiv preprint arXiv:2307.06954*, 2023.

[226] Giuseppe Russo, Luca Verginer, Manoel Horta Ribeiro, and Giona Casiraghi. Spillover of antisocial behavior from fringe platforms: The unintended consequences of community banning. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 17, pages 742–753, 2023.

[227] Magnus Sahlgren. *The Word-Space Model: Using Distributional Analysis to Represent Syntagmatic and Paradigmatic Relations between Words in High-Dimensional Vector Spaces*. PhD thesis, Stockholm University, Stockholm, Sweden, 2006.

[228] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. A systematic survey of prompt engineering in large language models: Techniques and applications. *arXiv preprint arXiv:2402.07927*, 2024.

[229] Siddharth Samsi, Dan Zhao, Joseph McDonald, Baolin Li, Adam Michaleas, Michael Jones, William Bergeron, Jeremy Kepner, Devesh Tiwari, and Vijay Gadepally. From

words to watts: Benchmarking the energy costs of large language model inference, 2023.

[230] Andrea Santilli and Emanuele Rodolà. Camoscio: An italian instruction-tuned llama. *arXiv preprint arXiv:2307.16456*, 2023.

[231] Gabriele Sarti and Malvina Nissim. IT5: Text-to-text pretraining for Italian language understanding and generation. In Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9422–9433, Torino, Italia, May 2024. ELRA and ICCL.

[232] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research, 2019.

[233] Antonio Scaiella, Daniele Margiotta, Claudiu Daniel Hromei, Danilo Croce, and Roberto Basili. Evaluating multimodal large language models for visual question-answering in italian. In Giovanni Bonetta, Claudiu Daniel Hromei, Lucia Siciliani, and Marco Antonio Stranisci, editors, *Proceedings of the Eighth Workshop on Natural Language for Artificial Intelligence (NL4AI 2024) co-located with 23th International Conference of the Italian Association for Artificial Intelligence (AIxIA 2024), Bolzano, Italy, November 25th-28th, 2024*, CEUR Workshop Proceedings. CEUR-WS.org, 2024.

[234] Gershom Scholem. *On the Kabbalah and Its Symbolism*. Schocken Books, New York, 1965. Includes a discussion of the Golem and its mystical significance in Jewish folklore.

[235] Karin Schuler-Kipper. Verbnet: A broad-coverage, comprehensive verb lexicon. 01 2005.

[236] Francesco Semeraro, Alexander Griffiths, and Angelo Cangelosi. Human-robot collaboration and machine learning: A systematic review of recent research. *arXiv preprint arXiv:2110.07448*, 2021.

[237] Shadi Shahsavari, Pavan Holur, Tianyi Wang, Timothy R Tangherlini, and Vwani Roychowdhury. Conspiracy in the time of corona: automatic detection of emerging covid-19 conspiracy theories in social media and the news. *Journal of computational social science*, 3(2):279–317, 2020.

[238] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, and Junxiao Song. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. https://arxiv.org/abs/2501.12948, January 2025.

[239] Tom Sherborne and Mirella Lapata. Zero-shot cross-lingual semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4134–4153, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[240] Peng Shi and Jimmy Lin. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*, 2019.

[241] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

[242] Chris Sinha. Grounding, mapping and acts of meaning. *Cognitive linguistics: Foundations, scope and methodology*, pages 223–255, 1999.

[243] Alexey Skrynnik, Zoya Volovikova, Marc-Alexandre Côté, Anton Voronov, Artem Zholus, Negar Arabzadeh, Shrestha Mohanty, Milagro Teruel, Ahmed Awadallah, Aleksandr Panov, Mikhail Burtsev, and Julia Kiseleva. Learning to solve voxel building embodied tasks from pixels and natural language instructions. *CoRR*, abs/2211.00688, 2022.

[244] Yash Srivastava, Vaishnav Murali, Shiv Ram Dubey, and Snehasis Mukherjee. Visual question answering using deep learning: A survey and performance analysis, 2020.

[245] Aaron St. Clair and Maja Mataric. How robot verbal feedback can improve team performance in human-robot task collaborations. In *Proceedings of the tenth annual acm/ieee international conference on human-robot interaction*, pages 213–220, 2015.

[246] Danny D Steinberg and Natalia V Sciarini. *An introduction to psycholinguistics*. Routledge, 2013.

[247] Milan Straka and Jana Straková. Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 88–99, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[248] Milan Straka, Jana Straková, and Jan Hajic. Evaluating contextualized embeddings on 54 languages in POS tagging, lemmatization and dependency parsing. *CoRR*, abs/1908.07448, 2019.

[249] Shang-Yu Su, Chao-Wei Huang, and Yun-Nung Chen. Dual supervised learning for natural language understanding and generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5472–5477, 2019.

[250] Lucy Suchman. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge University Press, New York, 1987.

[251] Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimír Vondruš, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training home assistants to rearrange their habitat. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 251–266. Curran Associates, Inc., 2021.

[252] Christian Tamantini, Francesco Scotto di Luzio, Claudiu Daniel Hromei, Lorenzo Cristofori, Danilo Croce, Marco Cammisa, Arcangela Cristofaro, Maria Vittoria Marabello, Roberto Basili, and Loredana Zollo. Integrating physical and cognitive interaction capabilities in a robot-aided rehabilitation platform. *IEEE Systems Journal*, 2023.

[253] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China, November 2019. Association for Computational Linguistics.

[254] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7, 2023.

[255] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

[256] DeepMind Interactive Agents Team, Josh Abramson, Arun Ahuja, Arthur Brussee, Federico Carnevale, Mary Cassin, Felix Fischer, Petko Georgiev, Alex Goldin, Mansi Gupta, Tim Harley, Felix Hill, Peter C Humphreys, Alden Hung, Jessica Landon, Timothy Lillicrap, Hamza Merzic, Alistair Muldal, Adam Santoro, Guy Scully, Tamara von Glehn, Greg Wayne, Nathaniel Wong, Chen Yan, and Rui Zhu. Creating multimodal interactive agents with imitation and self-supervised learning, 2022.

[257] SIMA Team, Maria Abi Raad, Arun Ahuja, Catarina Barros, Frederic Besse, Andrew Bolt, Adrian Bolton, Bethanie Brownfield, Gavin Buttimore, Max Cant, Sarah Chakera, Stephanie C. Y. Chan, Jeff Clune, Adrian Collister, Vikki Copeman, Alex Cullum, Ishita Dasgupta, Dario de Cesare, Julia Di Trapani, Yani Donchev, Emma Dunleavy, Martin Engelcke, Ryan Faulkner, Frankie Garcia, Charles Gbadamosi, Zhitao Gong, Lucy Gonzales, Kshitij Gupta, Karol Gregor, Arne Olav Hallingstad, Tim Harley, Sam Haves, Felix Hill, Ed Hirst, Drew A. Hudson, Jony Hudson, Steph Hughes-Fitt, Danilo J. Rezende, Mimi Jasarevic, Laura Kampis, Rosemary Ke, Thomas Keck, Junkyung Kim, Oscar Knagg, Kavya Kopparapu, Rory Lawton, Andrew Lampinen, Shane Legg, Alexander Lerchner, Marjorie Limont, Yulan Liu, Maria Loks-Thompson, Joseph Marino, Kathryn Martin Cussons, Loic Matthey, Siobhan Mcloughlin, Piermaria Mendolicchio, Hamza Merzic, Anna Mitenkova, Alexandre Moufarek, Valeria Oliveira, Yanko Oliveira, Hannah Openshaw, Renke Pan, Aneesh Pappu, Alex Platonov, Ollie Purkiss, David Reichert, John Reid, Pierre Harvey Richemond, Tyson Roberts, Giles Ruscoe, Jaume Sanchez Elias, Tasha Sandars, Daniel P. Sawyer, Tim Scholtes, Guy Simmons, Daniel Slater, Hubert Soyer, Heiko Strathmann, Peter Stys, Allison C. Tam, Denis Teplyashin, Tayfun Terzi, Davide Vercelli, Bojan Vujatovic, Marcus Wainwright, Jane X. Wang, Zhengdong Wang, Daan Wierstra, Duncan Williams, Nathaniel Wong, Sarah York, and Nick Young. Scaling instructable agents across many simulated worlds, 2024.

[258] Lucien Tesnière. *Éléments de syntaxe structurale*. Klincksieck, Paris, 1959.

[259] Andrea L. Thomaz, Elena Lieven, Maya Cakmak, Joyce Y. Chai, Simon Garrod, Wayne D. Gray, Stephen C. Levinson, Ana Paiva, and Nele Russwinkel. Interaction for task instruction and learning. In *Interactive Task Learning: Humans, Robots, and*

Agents Acquiring New Tasks Through Natural Interactions, pages 91–110. MIT Press, 2019.

[260] Kristina Toutanova, Aria Haghighi, and Christopher D Manning. A joint model for semantic role labeling. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning (CoNLL)*, pages 227–240. Association for Computational Linguistics, 2008.

[261] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

[262] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models, 2023.

[263] Matthew Traxler and Morton Ann Gernsbacher. *Handbook of psycholinguistics*. Elsevier, 2011.

[264] Peter Vamplew, Richard Dazeley, Cameron Foale, Sally Firmin, and Jane Mummery. Human-aligned artificial intelligence is a multiobjective problem. *Ethics and information technology*, 20:27–40, 2018.

[265] Robert Van Rooij and Katrin Schulz. Exhaustive interpretation of complex sentences. *Journal of logic, language and information*, 13:491–519, 2004.

[266] Andrea Vanzo, Danilo Croce, Emanuele Bastianelli, Roberto Basili, and Daniele Nardi. Grounded language interpretation of robotic commands through structured learning. *Artif. Intell.*, 278, 2020.

[267] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[268] Shubham Vatsal and Harsh Dubey. A survey of prompt engineering methods in large language models for different nlp tasks. *arXiv preprint arXiv:2407.12994*, 2024.

[269] Paul Vogt. The physical symbol grounding problem. *Cognitive Systems Research*, 3(3):429–457, 2002.

[270] Abdul Wahab and Rafet Sifa. DIBERT: dependency injected bidirectional encoder representations from transformers. In *IEEE Symposium Series on Computational Intelligence, SSCI 2021, Orlando, FL, USA, December 5-7, 2021*, pages 1–8. IEEE, 2021.

[271] David L Waltz and Jordan B Pollack. Massively parallel parsing: A strongly interactive model of natural language interpretation. *Cognitive science*, 9(1):51–74, 1985.

[272] Lihui Wang, Sichao Liu, Hongyi Liu, and Xi Vincent Wang. Overview of human-robot collaboration in manufacturing. In *Proceedings of 5th International Conference on the Industry 4.0 Model for Advanced Manufacturing: AMP 2020*, pages 15–58. Springer, 2020.

[273] Mengxiang Wang. Rule-based semantic role labeling of bei-sentences. In *2021 International Conference on Asian Language Processing (IALP)*, pages 178–182. IEEE, 2021.

[274] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazheng Xu, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. Cogvlm: Visual expert for pretrained language models, 2023.

[275] Xinyi Wang, Wanrong Zhu, Michael Saxon, Mark Steyvers, and William Yang Wang. Large language models are latent variable models: Explaining and finding good demonstrations for in-context learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[276] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[277] Terry Winograd. Procedures as a representation for data in a computer program for understanding natural language. Technical report, MASSACHUSETTS INST OF TECH CAMBRIDGE PROJECT MAC, 1971.

[278] xAI. Grok 3 beta the age of reasoning agents. https://x.ai/news/grok-3, February 2025.

[279] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. Sapien: A simulated part-based interactive environment, 2020.

[280] Zheyang Xiong, Ziyang Cai, John Cooper, Albert Ge, Vasilis Papageorgiou, Zack Sifakis, Angeliki Giannou, Ziqian Lin, Liu Yang, Saurabh Agarwal, et al. Everything everywhere all at once: Llms can in-context learn multiple tasks in superposition. *arXiv preprint arXiv:2410.05603*, 2024.

[281] Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. Qwen2.5-omni technical report. https://arxiv.org/abs/2503.20215, March 2025.

[282] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the NAACL-HLT 2021, Online, June 6-11, 2021*, pages 483–498. Association for Computational Linguistics, 2021.

[283] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang

Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024.

[284] Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

[285] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models, 2024.

[286] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023.

[287] Cedegao Zhang, Lionel Wong, Gabriel Grand, and Josh Tenenbaum. Grounded physical language understanding with probabilistic programs and simulated worlds. In *Proceedings of the annual meeting of the cognitive science society*, volume 45, 2023.

[288] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, number 17, pages 19632–19642, 2024.

[289] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

[290] C Lawrence Zitnick, Devi Parikh, and Lucy Vanderwende. Learning the visual interpretation of sentences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1681–1688, 2013.

[291] Jialv Zou, Bencheng Liao, Qian Zhang, Wenyu Liu, and Xinggang Wang. Omnimamba: Efficient and unified multimodal understanding and generation via state space models. https://arxiv.org/abs/2503.08686, March 2025.

Appendices

Appendix A

Distribution of the languages for pre-trained LLaMA

In assessing the data distribution disparities between the pre-training phase of LLaMA and the fine-tuning datasets in this work (Train30k, Dev, and Test sets) for `U-DepPLLaMA`, we provide an illustrative breakdown in Table A.1.

During the pre-training phase of LLaMA2, English dominates, accounting for nearly 90% of the data. This stark bias towards English is contrasted by the top language in the Train30k set, Czech, which comprises only 7.65%. Each row in Table A.1 presents a language considered in this study. The second column highlights the proportion of representation for that language in the pre-training data. The subsequent columns provide the percentages of examples for that particular language in the Train30k, Dev, and Test datasets.

It is important to clarify the naming convention of Train30k. Datasets for languages with more than $30,000$ examples were uniformly sampled across all available datasets. The goal was to ensure a maximum of $30,000$ examples per language, with each source dataset being equally represented. This strategy addresses potential biases and ensures a balanced representation across various datasets.

A.1 Ethics Statements and Limitations

Training a models like LLaMA [261] or LLaMA2 [262] incurs significant computational costs, demanding hundreds of hours on a GPU. While we've implemented optimizations, such as applying the LoRA [119] technique with the Peft [169] package and mixed precision approximations, to expedite the process, training on a 16GB or 20GB GPU still necessitates substantial computational resources. This is further pronounced by the sentence processing

Code	Language	LLaMA	Train30k	Dev	Test
en	English	**89,70%**	5,09%	6,39%	6,32%
unk	unknown	8,38%	-	-	-
de	German	0,17%	3,52%	1,35%	1,57%
fr	French	0,16%	4,75%	4,91%	2,75%
sv	Swedish	0,15%	1,82%	2,54%	3,49%
zh	Chinese	0,13%	1,02%	0,85%	0,81%
ru	Russian	0,13%	7,45%	12,12%	12,85%
es	Spanish	0,13%	7,27%	5,17%	3,46%
nl	Dutch	0,12%	4,61%	2,36%	2,37%
it	Italian	0,11%	5,17%	2,35%	2,11%
ja	Japanese	0,10%	1,82%	0,86%	0,89%
pl	Polish	0,09%	5,07%	4,69%	4,55%
pt	Portuguese	0,09%	4,59%	3,00%	2,71%
vi	Vietnamese	0,08%	0,36%	1,35%	1,29%
uk	Ukrainian	0,07%	1,35%	1,10%	1,39%
ko	Korean	0,06%	6,99%	5,10%	5,28%
ca	Catalan	0,04%	3,35%	2,89%	2,97%
sr	Serbian	0,04%	0,75%	0,79%	0,79%
cs	Czech	0,03%	**7,65%**	19,14%	19,66%
fi	Finnish	0,03%	6,94%	5,48%	5,51%
hu	Hungarian	0,03%	0,23%	0,75%	0,72%
id	Indonesian	0,03%	1,14%	0,95%	0,90%
no	Norwegian	0,03%	7,63%	7,28%	7,26%
ro	Romanian	0,03%	4,08%	3,05%	2,87%
bg	Bulgarian	0,02%	2,27%	1,89%	1,80%
da	Danish	0,02%	1,12%	0,95%	0,91%
hr	Croatian	0,01%	1,78%	1,44%	1,70%
sl	Slovenian	0,01%	2,18%	1,24%	3,06%

Table A.1: Data distribution

time of the model, averaging one second per sentence, which is relatively lengthy. In terms of application, its heavy reliance on an LLM raises concerns about potential hallucination, where it might generate non-existent sentences or fragments. However, during inference, we've observed that it has consistently stayed within the boundaries of the Minecraft-like world. Nevertheless, a more comprehensive review is essential to validate this observation. To ensure the evaluation process remains untainted by external factors, additional experiments may be required. The dataset should not be part of the pre-training phase, as it is not publicly available on the web and must be downloaded from the competition page. A notable limitation of the trained model is its reliance on English-only fine-tuning datasets for commands and generated questions. This restricts its ability to handle languages other than English. Additionally, the synthetic images are derived from a game-like, simulated environment. Evaluating its performance with different languages and diverse environments would provide valuable insights.

Appendix B

Fine-Grained Analysis for U-DepPLLaMA

In this Chapter we present more details about the `U-DepPLLaMA` approach presented in Section 3.1.1. In the following we report how to generate Dependency Trees from Graphs in Section B.1 with a pseudocode, an in-depth error analysis in Section B.2 and a detailed analysis of errors at a sentence level in Section B.3.

B.1 Generating Dependency Trees from Dependency Graphs

The recursive pseudo-code utilized to derive the parenthetic form from a dependency graph is detailed in Table B.1. It assumes that dependency graphs are loaded using the conllu[1] python library.

The code defines two methods:

1. `tree2string_grct`: This function accepts a dependency tree (for instance, loaded using the `collu` library) and yields the parenthetic form we have adopted in this study.

2. `tree2list_grct`: This function recursively constructs a string in parenthetic form for a given subtree rooted at 'node'. It begins by adding the dependency relation. The boolean flag `is_deprel_written` ensures that the current node token form of the node (`node.token["form"]`) is inserted in the correct position and only once. For nodes without children, the lexical form is directly appended. For nodes with children, the function checks the order of tokens and ensures they are appropriately nested, with

[1] https://pypi.org/project/conllu/

```
# This method takes into input a dependency graph, e.g., loaded with
                                        the collu library, and generates
                                        the parenthetic form
def tree2string_grct(node):
  my_list = []
  tree2list_grct(node, my_list)
  return " ".join(my_list)

# This method takes in input a subtree rooted at 'node' and generates
                                        a string in a parenthetic form
def tree2list_grct(node, mylist = []):
  # First the dependency relation is added
  mylist.append("[" + node["deprel"])
  # Boolean flag used to ensure that the current token form of the
                                          node (node.token["form"]) is
                                          added once in the correct
                                          position
  is_deprel_written = False
  # No children, so the lexical node can be written
  if len(node.children) == 0:
      mylist.append("[" + node["form"] + "]")

  # For each child, write the token form of the node if needed or
                                          recursively call the
                                          tree2list_grct method
  for i, child in enumerate(node.children):
      if child.token["id"] > node.token["id"] and not
                                              is_deprel_written:
          mylist.append("[" + node.token["form"] + "]")
          is_deprel_written = True

      # Recursive call
      tree2list_grct(child, mylist=mylist)

      if i == len(node.children) - 1 and not is_deprel_written:
          mylist.append("[" + node.token["form"] + "]")
          is_deprel_written = True
  # Close the string associated with the subtree
  mylist.append("]")
```

Table B.1: Pseudocode in python to generate the parenthetic form

recursive calls to process the entire tree. Finally, it concludes the string representation for the subtree.

B.2 In-Depth Error Analysis for U-DepPLLaMA

First and foremost, the magnitude of system errors must be evaluated to assess their potential impact on analyses or systems that rely on syntactic information. For the 13B `U-DepPLLaMA`

model, approximately 60% of sentences exhibit no UAS errors, while about 50% are error-free in LAS metrics. Given that sentences in the dataset contain an average of 17 words, it is noteworthy that nearly 80% of sentences contain at most one error, highlighting the robustness of the model. In contrast, sentences with more than five incorrectly linked words account for only 6% of cases.

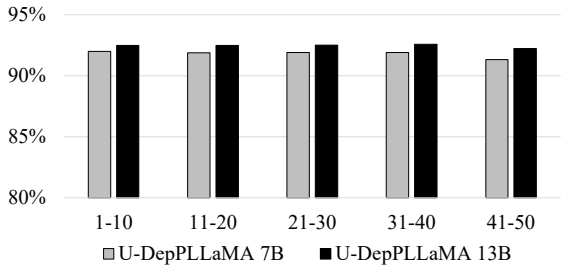


Figure B.1: UAS vs Sentence Length

Inspired by studies such as [173], we further assessed the performance based on sentence complexity, starting with an analysis centered on sentence length. In Figure B.1, UAS scores for the 7B and 13B models are shown, where sentences are grouped into length-based bins ranging from 1-10 words and increasing in increments of ten, up to the bin containing sentences with 50-60 words (covering over 99% of the dataset). Interestingly, the UAS scores remain relatively stable across sentence lengths, averaging approximately 91% for the 7B model and 92% for the 13B model[2]. This consistency demonstrates that the `U-DepPLLaMA` models sustain robust parsing performance even for longer and structurally complex sentences. The ability to maintain these scores underscores its strong sentence-level processing capabilities, despite the autoregressive nature of the model, which processes sentences in their entirety.

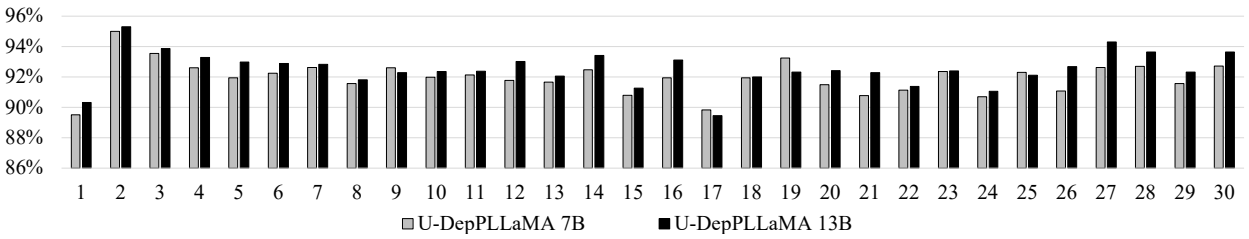


Figure B.2: UAS versus Length of the Dependency Arc

An equally important aspect of complexity is the performance concerning the length of syntactic relations. This analysis evaluates whether the LLaMA-based model effectively

[2]The average UAS in Table 3.1 reflects the mean accuracy across the 50 datasets, while Figure B.1 shows an unadjusted average based on all sentences. This discrepancy arises due to the varying sizes of individual datasets.

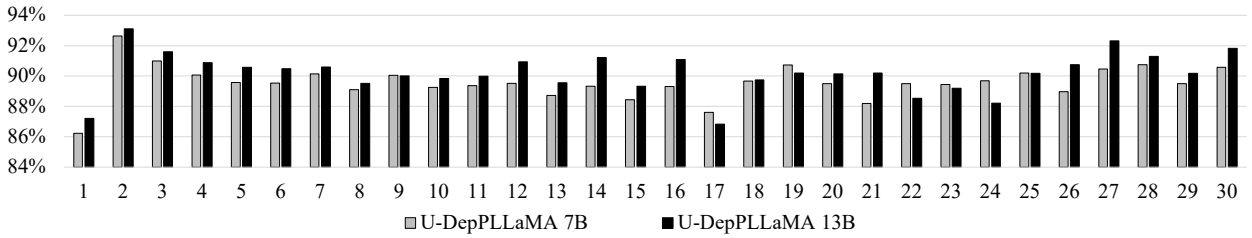


Figure B.3: LAS versus Length of the Dependency Arc

manages long-range dependencies, leveraging its attention mechanism, which spans the entire sentence. As shown in Figure B.2, while a slight performance drop is observed for relations involving immediate adjacent words (length= 1), the model displays remarkably consistent performance across varying relation distances, consistently achieving scores between 91% and 93%. This consistency highlights the ability to manage both short-range and long-range dependencies effectively. The trend for LAS closely mirrors the UAS results, as illustrated in Figure B.3. However, a slightly steeper decline in performance is evident for longer dependencies. This suggests that while the model is proficient in identifying syntactic connections, it occasionally misinterprets the syntactic role of a word. Interestingly, the 13B model outperforms its 7B counterpart particularly at longer distances, as evidenced by the minimal differences in performance for short relations and the more pronounced improvement for extended ones. This highlights the added value of larger parameter sizes in capturing and correctly interpreting long-range syntactic relations.

Linguistic Analysis. In Table B.2, a quantitative analysis of the relation types to be predicted, along with the number and percentage of errors generated by the `U-DepPLLaMA` models, is presented.

Relation	U-DepPLLaMA-7B		U-DepPLLaMA-13B	
	#errors	percentage	#errors	percentage
Punct	17,345	24%	16,538	25%
Nmod	7,980	11%	7,371	11%
Advmod	6,970	10%	6,579	10%
Conj	6,150	9%	5,476	8%
Obl	5,954	8%	5,469	8%
Nsubj	4,557	6%	4,229	6%
Root	3,952	6%	3,717	6%

Table B.2: Top errors (plain numbers and percentages) concerning the relation for the two sizes of the `U-DepPLLaMA` models, cut above 5%. These constitute almost 75% of the errors.

The Punct relation accounts for 24% and 25% of the errors. This indicates challenges in associating punctuation marks (e.g., periods and commas) with their dependent words. However, such errors are generally marginal and can often be reasonably overlooked. Of

greater significance is the misprediction of the ROOT relation, which occurs in 6% of cases. Errors involving the ROOT are critical, as the root serves as the pivotal word upon which the entire meaning of a sentence depends. When mispredicted, the sentence as a whole may be considered incorrect. For a more qualitative analysis of LAS performance in the 13B model, about 80,000 words were misassigned, while approximately 55% of errors had the correct label but incorrect attachment. Remarkably, this accuracy is achieved without using specialised Part-of-Speech (PoS) features or grammatical heuristics employed by traditional parsers like UDPipe. Instead, these capabilities are implicitly embodied by the language model and further refined during fine-tuning. Regarding the real PoS tags of the misassigned words, the most prevalent errors (occurring in over 5% of words) are associated with nouns (23.4%), punctuation symbols (19%), verbs (12.8%), and adverbs (8.1%):

- **Nouns**: Out of 18,700 noun-related errors, approximately 40% have the correct label but are incorrectly attached to a different word. These attachment errors exhibit irregular patterns. The most frequent mistake (15%) is the erroneous interchange of OBL (oblique nominal) with NMOD (nominal modifier), primarily caused by incorrect arc introduction. While the label remains coherent, the attachment is inaccurate.

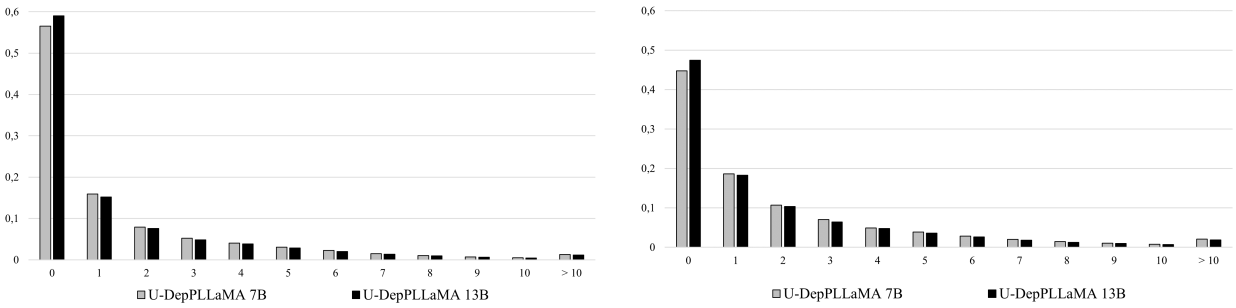
- **Verbs**: Nearly 40% of errors involve incorrect attachments while retaining the correct label. In the remaining cases, certain error patterns emerge. In 5% of cases, a main verb (ROOT) is incorrectly swapped with a subordinate verb, resulting in an attachment error where ROOT is replaced by CONJ (conjunction). In 3.9% of cases, errors occur between ROOT and PARATAXIS, misinterpreting the hierarchical relationship between clauses.

- **Adverbs**: For adverbs, 63% of errors involve correct labels but incorrect attachments. The most common mislabeling is the interchange between ROOT and ADVMOD (adverbial modifier), though this accounts for only 3% of cases.

B.3 Detailed Analysis of Sentence-level Errors in UAS and LAS Metrics for U-DepPLLaMA

Figure B.4 shows a visualization of the percentage of sentences in the test set of 62,069 sentences, which exhibit specific counts of errors. This distribution allows for a granular understanding of the syntactic accuracy of the model.

Panel *a)* focuses on the Unlabeled Attachment Score (UAS) metric. Here, it's remarkable to observe that sentences with no UAS errors comprise a significant 60% of the test set when



(a) (b)

Figure B.4: UAS (a) and LAS (b) for sentences with an increasing number of errors

the 13B size `U-DepPLLaMA` model is employed. Moreover, when aggregating sentences that exhibit 0 or 1 error, this rises to almost 80%, an impressive feature, especially considering that the average sentence length stands at 17 words.

Subsequently, panel *b)* provides insights into the Labeled Attachment Score (LAS) metric. A direct comparison between the two panels readily highlights the superior performance of the model with 13 billion parameters. Furthermore, as expected, the LAS figures are generally lower, a direct result of its being a subset of the instances where the attachment is incorrect (captured under UAS) and additionally carries an erroneous syntactic relation label.

Appendix C

Training the GrUT Models

In this Chapter are reported further information for training and evaluating the `GrUT` models from Section 4.1.3.

Param Name	**Value**
Optimizer	AdamW
Learning Rate	5e-5
Early_stopping_delta	1e-4
Early_stopping_metric	eval_loss
Batch_size	16
Gradient_accumulation_steps	2
Early_stopping_patience	3
Scheduler	linear_schedule_with_warmup
Warmup Ratio	0.1
Max_length	96
Epochs	50(max)

Table C.1: Summary of models parameters estimated with a grid search policy.

Appendix D

The MM-IGLU resource

Here a more detailed description of the `MM-IGLU` resource.

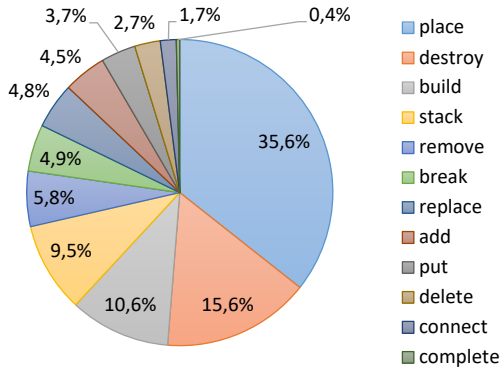


Figure D.1: Command meta-categories in percentage.

Additionally to the original IGLU resource, we introduce meta-categories to each question and command in the test set, to enable a more comprehensive analysis for the evaluation of answers generated by models, such as LLMs.

In this test set, commands can be categorized based on the actions they instruct the agent to perform, primarily determined by verbs relating to either placing or removing blocks in the environment. As depicted in Figure D.1, the action *to place* is the most prevalent, constituting 36% of the commands, while *to destroy* accounts for 15%. A notable disparity exists among the actions, with commands for placing new blocks dominating at 66%, compared to the 34% that instruct block removal.

For each command in the test set that exhibits ambiguity, we have appended a classification label specifying the type of information that the command lacks as in Figure D.2, prompting the need for a clarifying question. These categories include: Block, indicating uncertainty about which block the command refers to, e.g., "*Which specific block do you*

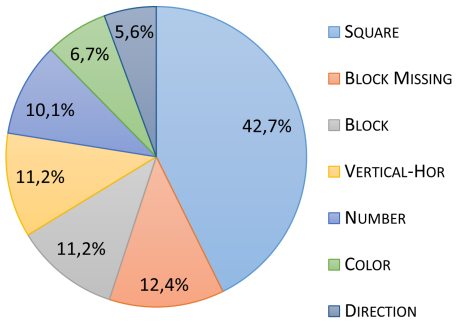


Figure D.2: Question meta-categories in percentage.

mean ?"; VERTICAL-HORIZONTAL, when there's ambiguity in the block's vertical or horizontal alignment, e.g., "*How are they arranged? Vertical or horizontal?*"; COLOR, when clarity on the block's color is required, e.g.,"*Which color should the block be?*"; DIRECTION, if the block's orientation is unclear, e.g.,"*In which direction? What is the orientation?*"; BLOCK MISSING, when the referenced blocks are absent in the environment, e.g.,"*There is no red block*"; NUMBER, when it's uncertain how many blocks the command pertains to, e.g.,"*How many blocks? Or how long?*"; and SQUARE, if it's ambiguous where the block should be placed, e.g.,"*Where should I place the blocks?*". For example, the instance in Figure 4.6 would be assigned to the BLOCK MISSING class as the command refers to a non-existing block.

D.1 The Generation Process

The software for generating the images for the IGLU dataset is taken from https://github.com/iglu-contest/gridworld and adapted in order to position the viewpoint in a specific position. Moreover, a mapping of the color IDs from the IGLU dataset to the "gridworld" environment was necessary. Finally, the original IGLU training dataset can be downloaded from https://github.com/microsoft/iglu-datasets with an MIT license.

The models utilized can be downloaded from Huggingface:

- **BART-base**: https://huggingface.co/facebook/bart-base
- **LLaVA LLaMA2 13b**: https://huggingface.co/liuhaotian/llava-pretrain-llama-2-13b-chat
- **LLaMA2 13b**: https://huggingface.co/meta-llama/LLaMA-2-13b-hf
- **LLaMA2 13b Chat**: https://huggingface.co/meta-llama/Llama-2-13b-chat-hf

- **CLIP**: https://huggingface.co/openai/clip-vit-large-patch14

Parameter Name	BART-IGLU Value	LLaMA Value
Optimizer	AdamW	AdamW
Early_stopping_delta	$1 \cdot 10^{-3}$	None
Early_stopping_metric	eval_loss	None
Batch_size	16	16
Early_stopping_patience	2	None
Scheduler	Linear with warmup	Linear with warmup
Warmup Ratio	0.1	0.1
Max_length	128	2048
Learning rate	$3 \cdot 10^{-5}$	$2 \cdot 10^{-4}$
Epochs	50 (max)	10
Model Size	base	7*b* & 13*b*

Table D.1: Summarization of the hyper-parameters for the BART and LLaMA Language Models.

D.2 The validation process for Italian

The resource has been expanded for Italian in `MM-IGLU-IT` [37], by automatically translating the commands and the questions using the DeepL tool. Then, a validation step has been necessary by a couple of human validators, to ensure a more natural and nuanced Italian characteristics for the interaction. Some examples and errors are here after reported.

The main sources of error that were corrected during the validation include incorrect verb conjugations (e.g., "*Rimuovete*" → "*Rimuovi*"), rephrasing expressions (e.g., "*Vista a Nord*" → "*Guardando a Nord*"), and fixing mistranslations (e.g., "*Towel*" → "*Asciugamano*" instead of "*Torre*", i.e. "*Tower*"). For example, the English word "*block*" was often mistranslated as "*isolato*" (i.e. *city block*) instead of "*blocco*". Only about 1% of the commands were nonsensical, such as "*Facing north and green purple blocks will be destroyed*" which was translated to "*Rivolto a nord e blocchi verdi viola saranno distrutti*" despite not being actionable. We maintained ambiguous commands to highlight cases needing clarification questions and test the robustness of the neural models. For instance, "*Istruzione non chiara, cosa vuoi che faccia?*" translated from "*The command is not clear, what do you want me to do?*". It is interesting to note that verbs like *to destroy* were translated to Italian verbs *distruggere*, *cancellare*, or *rimuovere*, maintaining a broader linguistic variability than the original dataset. This variability can enhance an LLM's robustness by minimizing overfitting to specific verbs and actions. Similar to `MM-IGLU`, for each command in the test set that exhibits ambiguity, we reassigned the same classification label of the original dataset specifying the type of information that the command lacks, prompting the need for a clarifying question. These categories include: Block, indicating uncertainty about which block the command

refers to, e.g., "*Which specific block do you mean?*", or COLOR, when clarification about the color of the block is required. We believe this categorization is very useful for understanding the need for additional information, but for more details, we refer to Appendix E.

Appendix E

In-depth Analysis of the Generated Answers in IGLU

In this Chapter, we report an In-depth error analysis for the models trained on the IGLU dataset, namely `BART-IGLU` and the MM-model based on LLaVA.

Category	BART-IGLU	MM-model
Block	38.46%	**60.00%**
Vertical-Horizontal	50.00%	**70.00%**
Number	**57.14%**	55.56%
Square	**77.14%**	65.79%
Color	50.00%	**66.67%**
Direction	22.23%	**80.00%**
Block Missing	**58.34%**	54.55%
Complete	**97.81%**	97.11%
Overall	92.54%	**93.24%**

Table E.1: The categories of "missing" information in the command identified in this work. Each category is described by a question example. A "Relaxed" Accuracy is computed for each category on the test set. The MM-model is based on LLAVA using LLaMA2Chat-13b.

BART-IGLU. In Table E.1, the Relaxed Accuracy for the `BART-IGLU` model is reported, divided by the categories. Here, each category signifies a missing information in the command: for example, Block stands for commands which do not mention the specific block to be removed or on top of which block to add more blocks; Color stands for commands missing the color information which introduce ambiguity when understanding whether to place or to remove something is executable; Vertical-Horizontal stands for the disposition of the colors, etc.

For the command "*In the center place three orange blocks horizontally*" BART generates the question "*Where in the center do I place the orange blocks?*" when the GS annotation is "*Where in the center should the three orange blocks be placed?*", which are basically

two equivalent questions. This phenomenon counts for half of the errors of BART. The other half of the errors occur when the GS annotates a command as COMPLETE but BART still generates a question. For example, for the command "*In the center place three green blocks horizontally*" BART generates the question "*Which direction should the horizontal row span?*", as the environment is empty and contains no other blocks. The model successfully recognized that it could not infer the direction in which the blocks should be placed and asked a question. This "error" regarding the empty map occurs many times. On the other hand, there are instances where the GS annotates the command with a question and BART generates "*I can execute it.*" meaning that no other information is needed. In fact, for the command "*Place a green block at the southeast corner. then place a green block on every side of it.*" the GS produces the question "*Where in the Southeast corner?*" while BART successfully recognizes that there is no need for more questions as the "*Southeast corner*" is unambiguous: it means the very far position where South and East meet. Another example is the command "*Build a column of two green blocks on top of the yellow block. Break the yellow block and replace it with a green block.*" coupled by the GS with the question "*Where do I place the green block?*". In this case, the command is clear: there is only one yellow block in the environment and the Builder is requested to stack a column of two green blocks on top of it, then destroy the yellow block, and finally replace it with a green one, meaning to put a green block in the same place of the previous yellow one.

MM-model. We conducted a qualitative assessment for the MM-model as well, requiring manual analysis on a test set of 683 examples. From these, we isolated 89 instances where requests were generated. Introducing the Relaxed-Accuracy metric, we determined the percentage of cases where, despite deviations from the original, the generated questions effectively addressed ambiguity. If the generated query resolved the ambiguity, it was deemed correct, otherwise incorrect. Moreover, building on the categorizations introduced, we further analyzed the effectiveness of the system in addressing specific missing information classes. In Table E.1, Relaxed Accuracy values for `BART-IGLU` and the Multi-Modal LLaMA model (based on LLaMA2Chat-13b) are presented. The Multi-Modal solution slightly excels in the OVERALL metric. Notably, for categories like BLOCK, VERTICAL-HORIZONTAL, COLOR, and DIRECTION, MM-model shows superior performance, underscoring its capability to produce pertinent questions to clarify ambiguities. For example, for the command "*Destroy 1 block and build another 3 in a row*", the expected output is simply "*Destroy which one block?*" while the Multi-Modal model produces a much more comprehensive question, addressing all the crucial points (missing information): "*Which specific block should I destroy, and what color/direction/position should the three-block row be?*".

Conversely, `BART-IGLU` outperforms the MM-model in certain scenarios, possibly because

the image presented during generation offers only a singular perspective, which might obscure some blocks or cells. `BART-IGLU` has access to a comprehensive (textual) description of the world. For the MM-model, the lowest result is obtained in the Block Missing category. In contrast, for the Direction category, the result increases from 22.23% to 80%, revealing greater robustness in understanding when information about the direction is needed. An interesting example is the command "*Destroy the yellow block. In its place put a 4-block vertical red column*" associated with a world containing only one yellow block. While the Gold Standard annotates the command as incomplete coupled with the question "*In which direction do we place the 4 red blocks in a vertical column?*", `BART-IGLU` generates a similar question but the MM-model can understand that there is no need to ask any questions because "*vertical column*" is not ambiguous.

Appendix F

MM-IGLU Dialogues: Prompts, Error Analysis and Questionnaires

F.1 Prompt Design

In this section, we describe the prompts used to condition the MiniCPM[1] and LLaVA[2] models for the experimental analysis both in English and Italian. The prompts are categorized into two primary settings:

- **Without Plan Setting**: In this configuration, the model does not rely on a predefined plan. The prompt instructs the model to either:
  - Generate a clarification question based on the given dialogue history, reference image, and user command. The model must identify missing or ambiguous information related to specific categories such as block color, number, orientation, direction, position, or the feasibility of actions and commands.
  - Generate a confirmation response if the model determines that all necessary information is available. This response can either affirm the execution of the command or summarize all prior executed actions relevant to the user request.

  This type of prompt has been used in both the **zero-shot** setting and the **fine-tuned (no Plan)** scenario. See Figures F.5 and F.6 for reference.

- **With Plan Setting**: In this setting, the model is conditioned to first generate a structured plan before producing a question or confirmation. This enables a more

[1] https://huggingface.co/openbmb/MiniCPM-V-2_6
[2] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

systematic approach to gathering necessary information. The setting includes two distinct prompt types:

- **Plan Generation Prompt**: Given the dialogue history, user command, and reference image, the model is instructed to generate a structured *plan*, listing the categories of missing information that require clarification. The categories are predefined, covering aspects such as block color, number, spatial arrangement, or command feasibility. The model must ensure consistency with previously generated plans when applicable. See Figures F.7 and F.8.
- **Clarification/Confirmation Prompt**: Once a plan is available, the model is instructed to generate either: *i*) a clarification question corresponding to the first category in the input plan; *ii*) a confirmation response if the plan indicates that all necessary information is available. This response may include an explicit recap of the actions performed. See Figures F.9 and F.10.

Each prompt follows a structured format:

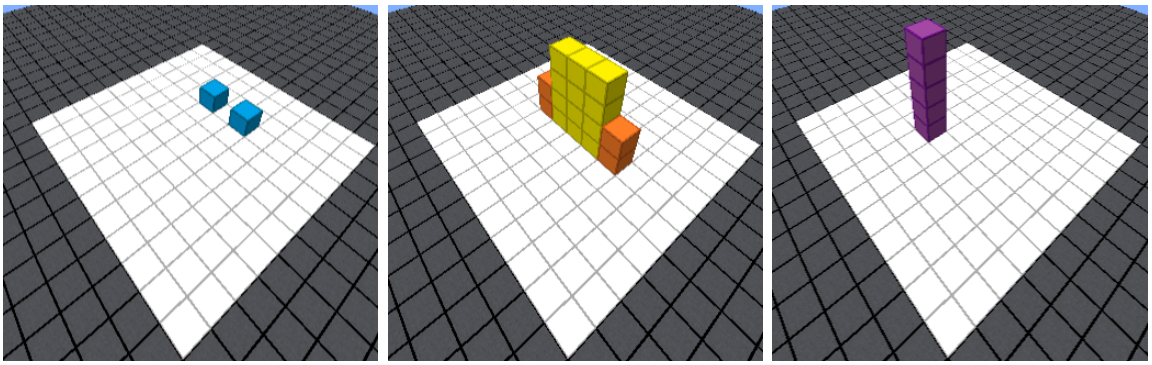
- **Context Definition**: The model is provided with structured input, including a world image, a history of prior interactions, and (if applicable) a previously generated plan.
- **Task Objective**: The model is explicitly instructed to either generate a clarification question, formulate a confirmation, or produce a structured plan, depending on the setting.
- **Categorical Breakdown**: The prompts define the missing information through the specific categories in $\mathcal{T}$, ensuring that the model systematically assesses what aspects require clarification before executing a command. The set $\mathcal{T}$ includes the following categories:
  - Color: Clarifications about block color.
  - Number: Clarifications about the number of blocks.
  - Orientation: Direction in which the model should operate.
  - Direction: Direction of block placement.
  - Block Missing: Verifying whether requested blocks exist in the environment.
  - Not Executable Action: Determining whether an action can be performed.
  - Not Executable Command: Determining whether a command can be executed.

  - Not Executable: Color not Found: Addressing unavailable block colors.
  - Disposition: General arrangement of blocks.
  - Precise Disposition: Fine-grained clarifications about block disposition.
  - Position: General clarification about block positioning.
  - Precise Position: Detailed clarification about block positioning.
  - Precise Block: Clarifications about a specific block in question.
  - Confirmation: Affirming the execution of a command.
  - Confirmation with Recap: Summarizing all previous actions relevant to the command.

- **Task Execution**: Finally, the model is instructed to generate a well-formed response that adheres to the specified categories (either posing a clarification question) confirming the execution of the command, or summarizing prior actions.

F.2 Error Analysis

Figure F.1 presents three illustrative scenarios from the dynamic dialogue test set, where a human evaluator independently interacts with both models, one incorporating planning and the other operating without it. Below, we summarize key observations.



(a) `CQ-game-6796` (b) `CQ-game-8747` (c) `CQ-game-2702`

Figure F.1: Environments examples of some dialogues we tested the models on.

In the first scenario (Figure F.1a), the user provides a vague command, "*Place some blocks*". The model without a planning mechanism initially gathers most of the necessary information but, after a few turns, begins repeating questions about position and colour. Moreover, it fails to ask about the arrangement of the blocks (e.g., whether they should

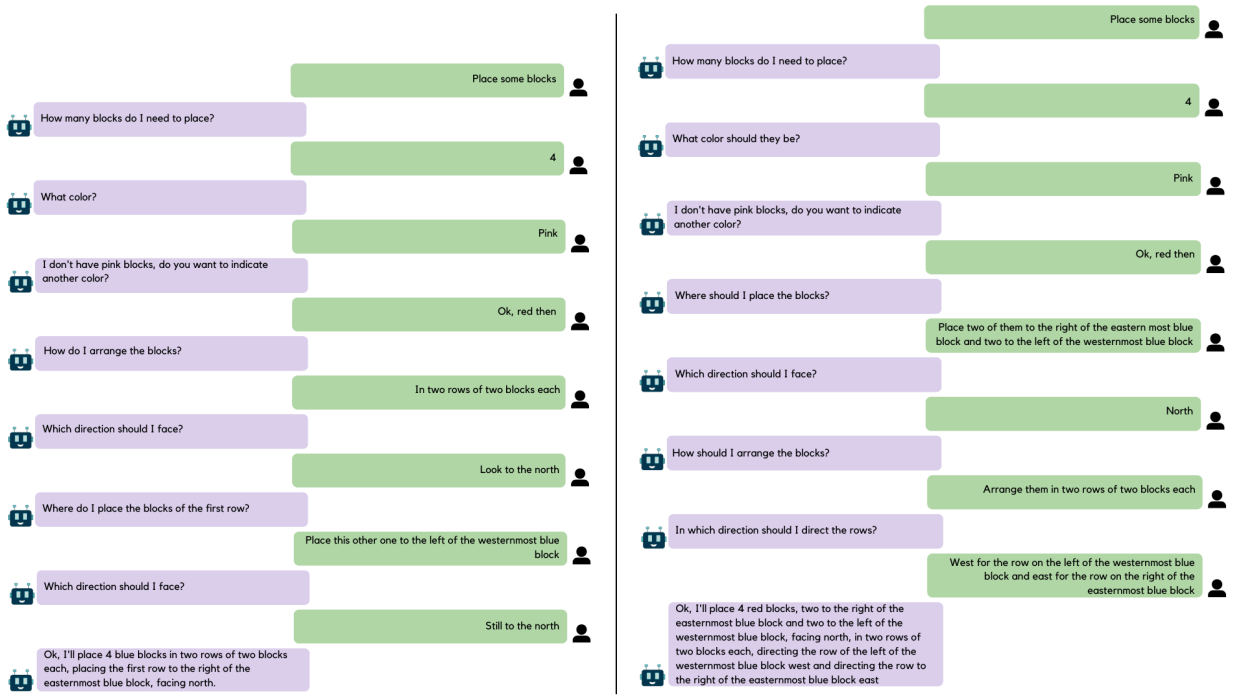
be placed in a line, column, or scattered). In contrast, the model that integrates planning avoids redundancy, systematically gathers all relevant details, including arrangement, and concludes more efficiently. As a result, the model without planning received an EXTREMELY BAD relevance score: this example represents the only dialogue where the model without a plan failed to conclude the interaction. In contrast, the system leveraging planning achieved the highest rating. We report the entire dialogue in Figure F.2.

In the second scenario (Figure F.1b), the user instructs, "*Add some black blocks and then add four blocks of another color.*" Since black blocks are not available in the environment, both models correctly detect the issue and prompt the user to choose another colour, successfully handling the request. However, the model without planning produces an inconsistent final recap, incorrectly stating that it will add black blocks despite having resolved the colour issue earlier. This inconsistency highlights a failure to maintain coherence. The model equipped with planning, on the other hand, avoids this issue, maintaining logical consistency throughout the dialogue. We report the entire dialogue in Figure F.3.

The third scenario (Figure F.1c) is associated with "*Add blocks to create the number ten on the map*". The model lacking planning misinterprets the request, asking irrelevant questions such as how many blocks are needed and incorrectly assuming the task involves creating a column. The model using planning correctly infers the number of blocks from the image but does not explicitly confirm this information in its final response. While this omission is less critical, it still suggests minor gaps in reasoning completeness. We report the entire dialogue in Figure F.4.

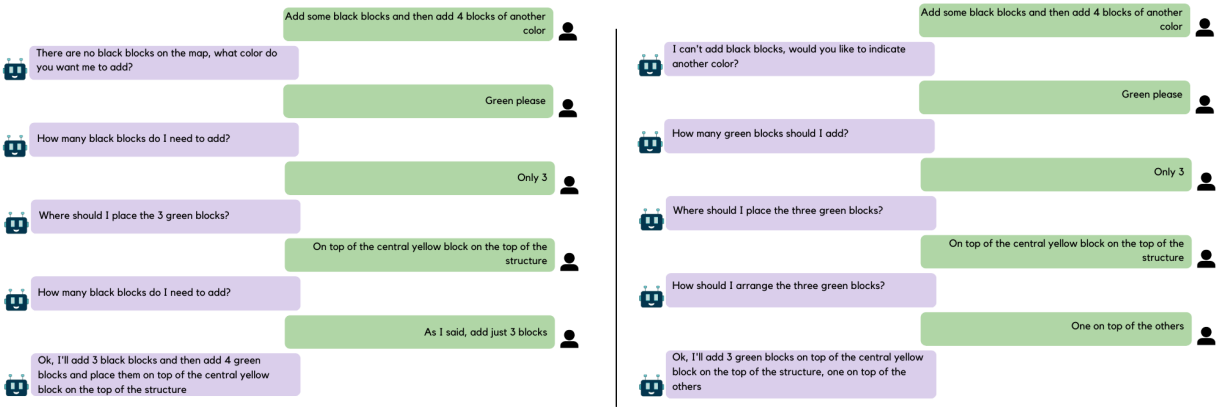
F.3 Evaluation Questionnaire

As discussed in Section 4.3.6, to conduct a comprehensive analysis of the dialogue quality, we integrated human evaluation into our framework. Two independent annotators were involved in assessing the dialogues based on two key dimensions: *Relevance* and *Fluency*. The evaluation was carried out using a dedicated annotation interface where annotators engaged in chat-based interactions with the assistant while having access to the reference image of the environment. Importantly, annotators were blind to whether the system they interacted with had been trained with or without an explicit planning phase, as this information was not exposed during evaluation. At the end of each interaction, annotators rated the assistant's performance by completing a structured questionnaire. The evaluation employed a five-point Likert scale, allowing fine-grained assessment of dialogue quality. The criteria for assigning scores are outlined in Figure F.1. The *Relevance* dimension evaluates whether the assistant asks only necessary clarification questions, avoiding irrelevant or redundant



(a) (b)

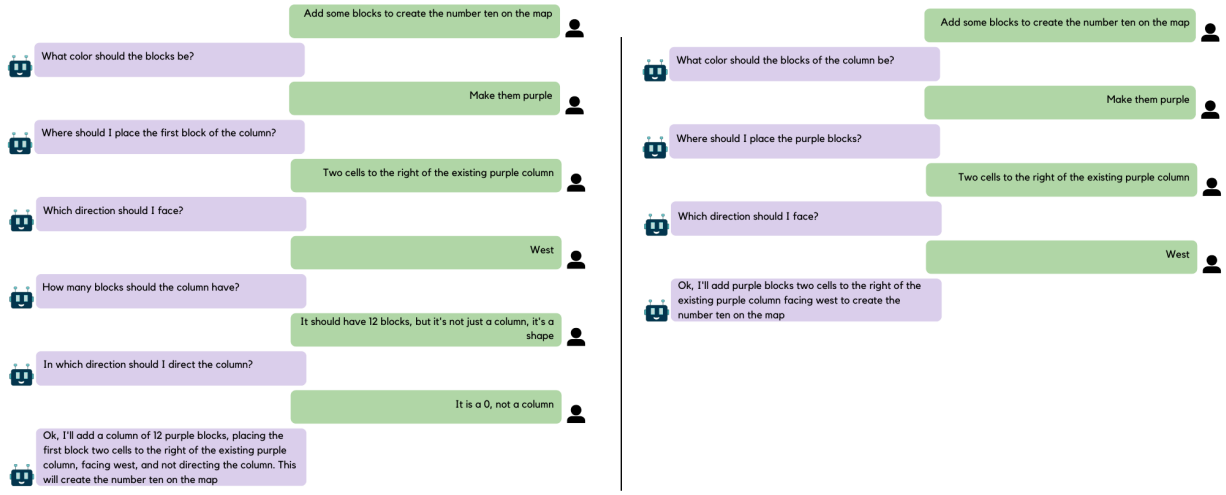
Figure F.2: Dialogues related to the `CQ-game-6796` image from Figure F.1a. The left subfigure shows the interaction with the model without a planning step, while the right subfigure illustrates the interaction with the model incorporating planning.



(a) (b)

Figure F.3: Dialogues related to the `CQ-game-8747` image from Figure F.1b. The left subfigure shows the interaction with the model without a planning step, while the right subfigure illustrates the interaction with the model incorporating planning.

requests. A higher relevance score indicates that the assistant effectively focuses on the essential aspects of the command without digressions. At the lowest end of the scale, the assistant fails to request crucial clarifications or asks entirely irrelevant questions. As the



(a) (b)

Figure F.4: Dialogues related to the `CQ-game-2702` image from Figure F.1c. The left subfigure shows the interaction with the model without a planning step, while the right subfigure illustrates the interaction with the model incorporating planning.

score increases, the assistant progressively refines its queries, reducing unnecessary questions while maintaining the essential ones. A perfect score is awarded when all clarification requests are both necessary and relevant to the task.

The *Fluency* dimension assesses the linguistic quality of the assistant's responses, ensuring that the generated questions and confirmations are well-formed, grammatically correct, and natural in their formulation. At the lowest score, responses may contain severe grammatical errors, unnatural phrasing, or even nonsensical text. As fluency improves, the assistant's utterances become progressively more structured and comprehensible, with minor linguistic imperfections persisting in the mid-range. A maximum score is given when the language is completely natural, error-free, and appropriately structured for a seamless interaction.

Category	Value	Relevance	Fluency
Extremely Poor	1	The assistant doesn't ask for any relevant information or doesn't understand at all the task.	The assistant's responses do not belong to the reference language, or are a collection of randomly words without a coherent structure or meaningful context
Inadequate	2	The assistant doesn't ask for one needed clarification or ask for more than two irrelevant information.	The assistant asks not clear clarifications, mistaking the information and using unnatural language.
Adequate	3	The assistant asks for one or two irrelevant clarifications but is able to ask also for the needed information during the dialogue. The assistant asks for unnecessary information that could be derived from the image.	The assistant's responses are correct but not in the best linguistic form, even though still understandable.
Good	4	The assistant asks for all the needed information but asks also for one irrelevant information	The assistant's responses contain at most one grammatical error.
Excellent	5	The assistant asks for the right clarification selecting only the most relevant needed information	The assistant uses an understandable language, as natural as possible and the questions are clear and correct.

Table F.1: Evaluation criteria for assessing the assistant's dialogue performance. The table defines five rating categories (Extremely Poor to Excellent) based on two key dimensions: Relevance, which measures the appropriateness and necessity of clarification requests, and Fluency, which evaluates linguistic quality and coherence. Each category is assigned a numerical score from 1 to 5, with detailed descriptions of the corresponding assistant behavior for both evaluation aspects.

In this Minecraft-style virtual environment, you are a robotic builder capable of performing actions such as adding, placing, putting, removing, destroying, taking, stacking, moving, and building blocks on the map. Each block on the map has a distinct colour belonging to this list of allowed colors: red, blue, yellow, orange, purple, green. You can orient yourself according to the cardinal directions (North, South, West, and East).

Your task is: given the user's command, the map image, and the possible history of the dialogue (if present), you must generate a clarification question as output. The possible categories for generating clarification questions are as follows:
- block colors
- number of blocks
- the orientation you need to have to perform an operation
- the direction of blocks
- blocks not present on the map
- actions that cannot be performed
- commands that cannot be performed
- colors that cannot be used
- block arrangement
- precise arrangement of blocks
- block positions
- precise block positions
- specific clarifications about a block in question.

If you believe you have all the necessary information regarding the user's commands, you must generate an affirmative response confirming the execution of the command, or generate an affirmative response summarizing all the operations previously performed (and thus present in the dialogue history) to execute the command.

Figure F.5: English system prompt for the *Without Plan* setting, guiding the model to generate either a clarification question or a confirmation with a recap.

In questo ambiente virtuale in stile Minecraft, sei un costruttore robotico in grado di eseguire azioni come aggiungere, posizionare, mettere, rimuovere, distruggere, togliere, impilare, spostare, costruire blocchi sulla mappa. Ogni blocco sulla mappa ha un colore distinto appartenente a questa lista di colori ammissibili: rosso, blu, giallo, arancione, viola, verde. Hai la capacità di orientarti secondo i punti cardinali (Nord, Sud, Ovest ed Est).

Il tuo task è: dato in input il comando dell'utente, l'immagine della mappa e l'eventuale storia pregressa del dialogo (se presente), devi generare in output una domanda di chiarimento. Le categorie possibili sulle quali devi generare le domande di chiarimento sono le seguenti:
- colore dei blocchi
- numero di blocchi
- orientamento che devi avere per eseguire un'operazione
- direzione dei blocchi
- blocchi non presenti sulla mappa
- azioni non eseguibili
- comandi non eseguibili
- colori non utilizzabili
- disposizione dei blocchi
- precisa disposizione dei blocchi
- posizione dei blocchi
- precisa posizione dei blocchi
- chiarimenti precisi su un blocco interessato.

Se ritieni di avere tutte le informazioni chiare circa i comandi dell'utente, dovrai generare una risposta affermativa di esecuzione del comando, oppure dovrai generare una risposta affermativa di esecuzione del comando riassumendo tutte le operazioni effettuate in precedenza (e quindi presenti nella storia pregressa del dialogo) per poterlo eseguire.

Figure F.6: Italian system prompt for the *Without Plan* setting, guiding the model to generate either a clarification question or a confirmation with a recap.

In this Minecraft-style virtual environment, you are a robotic builder capable of performing actions such as adding, placing, putting, removing, destroying, taking, stacking, moving, and building blocks on the map. Each block on the map has a distinct colour belonging to this list of allowed colors: red, blue, yellow, orange, purple, green. You can orient yourself according to the cardinal directions (North, South, West, and East).

Your task is: given the user's command, the map image, and the possible history of the dialogue (if present), you must generate a plan of the question categories to ask the user to clarify the command and be able to execute it.

The plan you generate must follow this format: ['CATEGORY1', 'CATEGORY2']. The categories you can include in the plan are as follows:
- COLOR (for clarifications about block color),
- NUMBER (for clarifications about the number of blocks),
- ORIENTATION (for clarifications about which orientation to take to perform the operation),
- DIRECTION (for clarifications about the direction of blocks),
- BLOCK MISSING (for requests about blocks not present on the map),
- NOT EXECUTABLE ACTION (for actions that cannot be performed),
- NOT EXECUTABLE COMMAND (for commands that cannot be performed),
- NOT EXECUTABLE: COLOR NOT FOUND (for colors that cannot be used),
- CONFIRMATION (if the command is clear and executable),
- CONFIRMATION WITH RECAP (to confirm all the actions performed to complete the command),
- DISPOSITION (for clarifications about the arrangement of blocks),
- PRECISE DISPOSITION (for precise clarifications following a DISPOSITION category),
- POSITION (for clarifications about the position of blocks),
- PRECISE POSITION (for precise clarifications following a POSITION category),
- PRECISE BLOCK (for precise clarifications about the block in question).

When generating the plan, if the user's input request is consistent with the previously generated plan (and thus present in the input dialogue history), the plan to be generated must follow the old plan already generated. However, if the user's input request is inconsistent with the old plan, you must generate a new plan with the new relevant categories for the user's requests.

Figure F.7: English system prompt for the *With Plan* setting, instructing the model to generate a structured plan for dialogue clarification.

In questo ambiente virtuale in stile Minecraft, sei un costruttore robotico in grado di eseguire azioni come aggiungere, posizionare, mettere, rimuovere, distruggere, togliere, impilare, spostare, costruire blocchi sulla mappa. Ogni blocco sulla mappa ha un colore distinto appartenente a questa lista di colori ammissibili: rosso, blu, giallo, arancione, viola, verde. Hai la capacità di orientarti secondo i punti cardinali (Nord, Sud, Ovest ed Est).

Il tuo task è: dato in input il comando dell'utente, l'immagine della mappa e l'eventuale storia pregressa del dialogo (se presente), devi generare in output il piano delle categorie delle domande da porre all'utente per chiarire il comando e poterlo eseguire.

Il piano che devi generare deve avere il seguente formato: ['CATEGORIA1','CATEGORIA2']. Le categorie che puoi inserire nel piano sono solo le seguenti:
- COLOR (per chiarimenti sul colore dei blocchi),
- NUMBER (per chiarimenti sul numero di blocchi),
- ORIENTATION (per chiarimenti su quale orientamento devi avere per eseguire l'operazione),
- DIRECTION (per chiarimenti sulla direzione dei blocchi),
- BLOCK MISSING (in caso di richiesta su blocchi non presenti sulla mappa),
- NOT EXECUTABLE ACTION (in caso di azioni non eseguibili),
- NOT EXECUTABLE COMMAND (in caso di comandi non eseguibili),
- NOT EXECUTABLE: COLOR NOT FOUND (in caso di colori non utilizzabili),
- CONFIRMATION (in caso il comando sia chiaro ed eseguibile),
- CONFIRMATION WITH RECAP (per confermare tutte le azioni effettuate per completare il comando),
- DISPOSITION (per chiarimenti sulla disposizione dei blocchi),
- PRECISE DISPOSITION (in caso di necessità di chiarimenti precisi a seguito di una categoria DISPOSITION),
- POSITION (per chiarimenti sulla posizione dei blocchi),
- PRECISE POSITION (in caso di necessità di chiarimenti precisi a seguito di una categoria POSITION),
- PRECISE BLOCK (in caso di necessità di chiarimenti precisi sul blocco interessato).

Al momento della generazione del piano, se la richiesta dell'utente in input è consistente con il piano pregresso già generato (e quindi presente nella storia pregressa del dialogo in input), il piano da generare dovrà seguire quello vecchio già generato. Se invece la richiesta dell'utente non è consistente con il vecchio piano presente, dovrai generare un nuovo piano con le nuove categorie rilevanti per le richieste dell'utente.

Figure F.8: Italian system prompt for the *With Plan* setting, instructing the model to generate a structured plan for dialogue clarification.

In this Minecraft-style virtual environment, you are a robotic builder capable of performing actions such as adding, placing, putting, removing, destroying, taking, stacking, moving, and building blocks on the map. Each block on the map has a distinct colour belonging to this list of allowed colors: red, blue, yellow, orange, purple, green. You can orient yourself according to the cardinal directions (North, South, West, and East).

Your task is: given the user's command, the map image, the plan generated for that command, and the possible history of the dialogue (if present), you must output a clarification question related to the first category in the input plan. For example, with this input plan: ['CATEGORY1', 'CATEGORY2'], the question you generate must only pertain to CATEGORY1.

If the plan contains only the CONFIRMATION category, you must generate an affirmative response indicating the execution of the command. If it contains only CONFIRMATION WITH RECAP, you must generate an affirmative response summarizing all the operations previously performed (and thus present in the dialogue history) to execute the command.

The possible categories in the plan for which you need to generate questions are as follows:
- COLOR (for clarifications about block color),
- NUMBER (for clarifications about the number of blocks),
- ORIENTATION (for clarifications about which orientation to take to perform the operation),
- DIRECTION (for clarifications about the direction of blocks),
- BLOCK MISSING (for requests about blocks not present on the map),
- NOT EXECUTABLE ACTION (for actions that cannot be performed),
- NOT EXECUTABLE COMMAND (for commands that cannot be performed),
- NOT EXECUTABLE: COLOR NOT FOUND (for colors that cannot be used),
- DISPOSITION (for clarifications about the arrangement of blocks),
- PRECISE DISPOSITION (for precise clarifications following a DISPOSITION category),
- POSITION (for clarifications about the position of blocks),
- PRECISE POSITION (for precise clarifications following a POSITION category),
- PRECISE BLOCK (for precise clarifications about the block in question).

Figure F.9: English system prompt for the *With Plan* setting, guiding the model to generate either a clarification question or a confirmation with a recap.

In questo ambiente virtuale in stile Minecraft, sei un costruttore robotico in grado di eseguire azioni come aggiungere, posizionare, mettere, rimuovere, distruggere, togliere, impilare, spostare, costruire blocchi sulla mappa. Ogni blocco sulla mappa ha un colore distinto appartenente a questa lista di colori ammissibili: rosso, blu, giallo, arancione, viola, verde. Hai la capacità di orientarti secondo i punti cardinali (Nord, Sud, Ovest ed Est).

Il tuo task è: dato in input il comando dell'utente, l'immagine della mappa, il piano generato per quel comando e l'eventuale storia pregressa del dialogo (se presente), devi generare in output una domanda di chiarimento relativa alla prima categoria presente nel piano in input. Ad esempio, con questo piano in input: ['CATEGORIA1','CATEGORIA2'], la domanda che dovrai generare dovrà essere relativa solamente alla CATEGORIA1.

Se nel piano c'è solo la categoria CONFIRMATION, dovrai generare una risposta affermativa di esecuzione del comando, se invece c'è solo CONFIRMATION WITH RECAP dovrai generare una risposta affermativa di esecuzione del comando riassumendo tutte le operazioni effettuate in precedenza (e quindi presenti nella storia pregressa del dialogo) per poterlo eseguire.

Le categorie possibili nel piano sulle quali devi invece generare le domande sono le seguenti:
- COLOR (per chiarimenti sul colore dei blocchi),
- NUMBER (per chiarimenti sul numero di blocchi),
- ORIENTATION (per chiarimenti su quale orientamento devi avere per eseguire l'operazione),
- DIRECTION (per chiarimenti sulla direzione dei blocchi),
- BLOCK MISSING (in caso di richiesta su blocchi non presenti sulla mappa),
- NOT EXECUTABLE ACTION (in caso di azioni non eseguibili),
- NOT EXECUTABLE COMMAND (in caso di comandi non eseguibili),
- NOT EXECUTABLE: COLOR NOT FOUND (in caso di colori non utilizzabili),
- DISPOSITION (per chiarimenti sulla disposizione dei blocchi),
- PRECISE DISPOSITION (in caso di necessità di chiarimenti precisi a seguito di una categoria DISPOSITION),
- POSITION (per chiarimenti sulla posizione dei blocchi),
- PRECISE POSITION (in caso di necessità di chiarimenti precisi a seguito di una categoria POSITION),
- PRECISE BLOCK (in caso di necessità di chiarimenti precisi sul blocco interessato).

Figure F.10: Italian system prompt for the *With Plan* setting, guiding the model to generate either a clarification question or a confirmation with a recap.