ID N. AIDR01/16910

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

Enhancing Neoantigen Identification: The Role of Artificial Intelligence and Proteogenomics in Personalized Cancer Therapy

Supervisor

Prof. Fabio Massimo Zanzotto

Candidate

Michele Mastromattei

April, 2025

A:

Woody (Toy Story)

Angewomon (Digimon)

All'intera saga di Harry Potter

Lady Gaga

Piero e Alberto Angela

Che inconsapevolmente hanno plasmato la persona che sono ... e:

ai colori della Puglia

la sfacciataggine di Roma

il calore della mia famiglia

il suono delle risate dei miei amici

Che consapevolmente hanno definito il mio perimetro della parola "casa"

Ringraziamenti (*for italian readers*)

"Se tornassi indietro, non lo rifarei mai". Ricordo perfettamente la prima volta che dissi questa frase: 6 Luglio 2022, appena 8 mesi dopo l'inizio di questo percorso e che oggi, a quasi 3 anni da quel Luglio del '22, questa affermazione è più viva che mai, se non addirittura peggiorata. Il mio percorso da dottorando è stato, senza mezze parole, un inferno, che, a confronto, quello dantesco è paragonabile a Julie Andrews che canta *"The Sound of Music"* sulle montagne bavaresi. E come nei gironi danteschi, ho passato tre anni a osservare come l'eccesso di avarizia possa diventare anche buffo, l'accidia spietata, la superbia meschina e l'invidia ampiamente infettiva (seppur le ultime due siano in Purgatorio). E come il Sommo Poeta, alle pendici del Purgatorio, si lasciò orientare da quattro stelle, simbolo di virtù cardinali, anche nel mio di viaggio sono stato illuminato da quattro stelle che hanno reso questo percorso più soave: *famiglia*, *amici*, *Edimburgo* e *Heidelberg*.

Sono davvero fortunato a essere nato e cresciuto in una famiglia come la mia, che ha vissuto in prima persona il punto più basso in questi miei 29 anni e l'ha attraversato in punta di piedi, delicatamente, osservando un mondo per loro completamente nuovo e cercando di trovare, a modo loro, una soluzione a tutto. Devo dunque un enorme grazie ai miei genitori: Giuseppina e Matteo, a mia sorella Marianna, a Matteo, ai miei nonni Antonio e Anna e alla nuova piccola stella, la più brillante tra tutte, che ha impreziosito questa galassia, di nome Enrico, che illumina e colora il mio intero Universo.

Se *gli amici sono la famiglia che ti scegli*, allora sono fortunato due volte. Più che mai, in questo percorso sparso in tutta Europa, questa stella ha brillato di luce fortissima e si

è ingrandita di anno in anno. Un grazie speciale va dunque alle mie due amiche sempre presenti fin dai banchi di scuola: Flavia e Margherita, che, da più di 15 anni, sono una certezza costante in ogni passo della mia vita. La cosa più bella che Roma abbia potuto regalarmi è stata, senza ombra di dubbio, *la mia seconda famiglia*: Giulia, Caterina, Giusy, Matteo e Sofia sono, semplicemente, *casa*.

Se devo indicare il momento più bello in questo inferno, non posso che indicare la terza stella, *Edimburgo*, che è stata capace di regalarmi momenti unici, di farmi ritornare, dopo 10 anni, in una Nazione che ho sempre amato e che non mi ha mai deluso. Un grazie enorme va alla Prof. Ajitha Rajan, senza la quale questa esperienza non si sarebbe mai avverata e che mi ha fatto sentire suo studente anche quando, tecnicamente, non lo sono mai stato. Lo stesso vale per la Prof. Mirella Lapata che – senza volere nulla in cambio – mi ha mostrato (a mio avviso) il vero lato della ricerca accademica: più razionale, più vera, basata sul sacrificio, sugli errori (da cui c'è sempre e solo da imparare) e tanto, ma tanto studio. Più di chiunque altro, questa stella brilla grazie a tutte le persone che ho incontrato dentro e fuori dall'Informatics Forum e soprattutto ai miei amici che, quotidianamente, hanno reso questa esperienza immensamente bella: Kadj, Piyush e Liam.

La mia quarta e ultima stella è stata la più inaspettata, ma anche colei che mi ha fatto capire quanto io valga, dei traguardi che ho raggiunto e di quelli che posso ancora delineare. Un grandissimo grazie va a Catherine e Raman, che hanno reso possibile questo mio viaggio a NEC, a Sandra, che ha sempre creduto nelle mie capacità, a Clara e Flavio, per essere stati la spalla per ogni mio minuscolo problema e a tutti i ragazzi dell'ufficio 2.04, che hanno reso questa esperienza meravigliosamente leggera.

Infine, devo una scusa particolare a una sola persona: Dante Alighieri, dal quale ho preso in prestito la sua Opera più famosa per stilare queste pagine. Ma se lui iniziò questo suo viaggio nella Pasqua del 1300, anno di un Giubileo, anch'io, in un anno di Giubileo, posso dire, da oggi, di vivere la mia Pasqua.

Acknowledgements (*for non italian readers*)

"If I could go back, I would never do it again". I remember vividly the first time I said those words: 6 July 2022, just eight months into this journey. Now, almost three years have passed since that day, and that thought has not softened... on the contrary, it has only grown sharper, more bitter. My PhD journey has been, without exaggeration, a hellish experience — so much so that Dante's inferno looks like Julie Andrews singing *"The Sound of Music"* on the Bavarian Alps by comparison. Like the circles of Dante's hell, I've spent the past three years observing how avarice can verge on the ridiculous, sloth can be ruthless, pride can turn petty, and envy can spread like wildfire (even if those last two, technically, belong to Purgatory). And just as the Supreme Poet, at the foot of Purgatory, was guided by four stars, symbols of the cardinal virtues, so too have I been led by four stars that made this path a little more bearable: *family*, *friends*, *Edinburgh*, and *Heidelberg*.

I feel incredibly lucky to have been born and raised in a family like mine. They witnessed firsthand the lowest point of my 29 years and walked through it with grace and care, stepping lightly in a world that was entirely new to them, trying in their own way to make sense of it all. I owe an enormous thank you to my parents, Giuseppina and Matteo, to my sister Marianna, to Matteo, to my grandparents Antonio and Anna, and to the newest little star — the brightest of them all — who is brought light and color to this galaxy of mine: Enrico, who illuminates my whole Universe.

If *friends are the family you choose*, then I am blessed twice. More than ever, along this journey scattered across Europe, that star has shone with an intense, unwavering light,

growing brighter and larger with every passing year. A special thank you goes to my two friends who have been by my side since school days: Flavia and Margherita — constant companions for over 15 years, unwavering through every step of my life. Without a doubt, the greatest gift Rome gave me is my *second family*: Giulia, Caterina, Giusy, Matteo and Sofia who are, quite simply, *home*.

If I had to pick the brightest moment in this personal hell, it would be my third star: *Edinburgh*. That city gave me unforgettable experiences and brought me back, after ten years, to a nation I have always loved. I am deeply grateful to Professor Ajitha Rajan, without whom this adventure wouldn't have been possible. She made me feel like her student even though, technically, I never was. The same goes for Professor Mirella Lapata, who gave me one of the most important insights of this journey: that true research is not about prestige or outcomes, but about effort, honesty, learning from failure and an unshakable commitment to the process. More than anyone, this star shines thanks to the incredible people I met in and outside the Informatics Forum, especially my friends who made every day of that experience truly wonderful: Kadj, Piyush, and Liam.

My fourth and final star was the most unexpected, but also the one that reminded me of my worth, of the milestones I have achieved and of those still waiting to be reached. A heartfelt thank you to Catherine and Raman, who made my time at NEC possible; to Sandra, who always believed in my potential; to Clara and Flavio, for being there for even the smallest of my problems, and to everyone in office 2.04, who made the entire experience feel beautifully light.

Finally, I owe a special apology to just one person: Dante Alighieri, from whom I have borrowed his most famous work to write these pages. But as Dante began his journey during Easter of the Jubilee year 1300, I can say in this Jubilee year: today, I celebrate my own Easter.

Declaration

I hereby declare that this thesis is the product of my own original work and research, except where explicitly acknowledged. All sources used have been cited and referenced appropriately in accordance with the University template provided.

Furthermore, I confirm that no chapters, sections, or paragraphs within this thesis have been generated using artificial intelligence (AI) or any other automated writing tools.

The work contained herein has not been previously submitted for any degree or professional qualification. However, all published papers related to this thesis have been cited and appropriately declared and any papers still under review during the thesis submission, have been noted.

April 2025

Michele Mastromattei

Abstract

Cancer neoantigens, unique tumor-specific proteins, are emerging as promising targets for personalized cancer vaccines. However, accurately identifying these neoantigens remains a significant challenge, primarily due to the limitations of computational prediction tools. Proteogenomics, a field integrating genomics and proteomics, offers a more comprehensive approach to neoantigen discovery. Simultaneously, artificial intelligence (AI), particularly advanced transformer models, has revolutionized numerous fields, including healthcare.

This thesis undertakes a fully comprehensive research of the design, implementation, and improvement of AI models aimed at neoantigen classification.

We begin with a detailed review of cancer biology, advancing into a technical exploration of contemporary AI methodologies, emphasizing state-of-the-art models and the datasets essential for their training.

To boost the training and evaluation of our AI models, we conducted a comprehensive re-analysis of 75 publicly accessible mass spectrometry datasets, culminating in the creation of the CARMEN dataset. Utilizing this dataset, along with custom algorithms, we delineated and curated multiple corpora specifically for training our machine learning models in neoantigen classification. These models were rigorously evaluated against established benchmarks, demonstrating comparable or superior performance. Surprisingly, models trained and tested solely on peptide sequences achieved similar or better results compared to those incorporating HLA information.

In response to the escalating complexity and computational intensity of AI models, we introduced a novel pruning algorithm named KEN. This algorithm automates the reset of non-essential parameters within a trained model, yielding smaller, more efficient subnetworks with negligible performance loss. When applied to our neoantigen classification models, KEN produced models that were not only smaller but also more efficient. Notably, the integration of HLA sequences into these pruned models enhanced their performance relative to models relying solely on peptide sequences.

In summary, this thesis represents a substantial advancement in cancer immunotherapy by innovating in the development and optimization of AI models for neoantigen classification. By effectively leveraging proteogenomics datasets and pioneering AI methodologies, we have established robust, efficient models capable of accurately identifying neoantigens, thus laying the groundwork for personalized cancer vaccines that hold promise for improved patient outcomes.

Contents

1 Introduction **1**

1.1 Contributions 3

1.2 Structure 6

1.3 List of Published Papers 8

2 Background **11**

2.1 The cancer genome 12

2.1.1 The Genetic Landscape of Cancer 13

2.1.2 Cancer Treatment: Current and Future Approaches 14

2.2 Molecular Mechanisms of Cancer Immunogenicity 15

2.2.1 Proteins, Peptides, and Amino Acids 16

2.2.2 Neoantigens 17

2.2.3 Major Histocompatibility Complex I 20

2.3 Peptide-based vaccines 22

2.3.1 Peptide-based Vaccines for Renal Cell Carcinoma 23

2.4 Artificial Intelligence in Cancer Vaccines Development 24

2.4.1 MHC-based Machine Learning Approaches 25

2.5 Proteomics repositories 29

2.5.1 PRoteomics IDEntifications database (PRIDE) 30

2.5.2 MassIVE (MASS spectrometry Interactive Virtual Environment) . . . 31

2.6 Transformer models 32
2.6.1 Bidirectional Encoder Representations from Transformers (BERT) . . 33
2.7 Machine Learning Pruning algorithms 36
2.7.1 Pruning algorithm categorization 37
2.7.2 The lottery ticket hypothesis 38

3 Methods **40**
3.1 CARMEN: Cancer ImmunoPeptidogenomics Corpus 41
3.1.1 Corpus Development 41
3.1.2 Deep analysis and composition 44
3.2 Tasks Assessing Protein Embeddings (TAPE) 46
3.2.1 Other BERT-base Transformer Models for Neoantigens detection . . . 50
3.3 Kernel density Estimator for Neural Network compression (KEN) 52
3.3.1 KEN pruning algorithm 53
3.3.2 Experiments on sentiment analysis datasets 56
3.3.2.1 Experimental set-up 56
3.3.2.2 Model compression 58
3.3.3 Results on sentiment analysis datasets 59
3.3.3.1 Experiment results 59
3.3.3.2 Compression values 61
3.3.3.3 KEN$_{viz}$ 63

4 Experiments **64**
4.1 Corpora Generation for Neoantigen Classification 66
4.1.1 Data Pre-processing 67
4.1.1.1 Cleaning and Filtering Data 67
4.1.1.2 Merging Datasets and Creating Training and Test Sets . . . 68
4.1.1.3 Analysis of the Resulting Datasets 70

4.1.2 Neoantigen Classification Datasets 74
4.1.2.1 Balanced datasets 76
4.1.2.2 Analysis of the obtained datasets 78
4.2 TAPE Setup Experiments 79
4.3 KEN Setup Experiments 82

5 Results **84**
5.1 TAPE Results 86
5.1.1 Training results 86
5.1.2 TAPE test results 88
5.1.3 Additional Analysis 90
5.1.3.1 MHCflurry Evaluation 90
5.1.3.2 TransPHLA and the Pfam Dataset 92
5.2 Optimized models using KEN 94

6 Conclusions **99**

Appendices **125**

A CARMEN dataset composition **125**
A.0.1 Peptide lenght 126
A.0.2 Spectral counts 127
A.0.3 HLA distribution 128
A.0.3.1 HLA-A 128
A.0.3.2 HLA-B 129
A.0.3.3 HLA-C 130

B KEN pruning algorithm additional experiments **131**
B.1 How to prove the importance of selected parameters 132

B.2 Additional results 134
B.3 KEN_{viz} examples 135

C Linguistic Fingerprint in Transformer Models 138

C.1 Introduction 139
C.2 Background and related work 139
C.3 Methods 141
C.3.1 EPIC Corpus 141
C.3.2 KEN algorithm 141
C.4 Experiments 142
C.5 Results 144
C.6 Conclusion 146
C.7 Additional results 146
C.7.1 DeBerta model fingerprints comparison 147
C.7.2 Ernie model fingerprints comparison 147
C.7.3 BERT model fingerprints comparison 148
C.7.4 DistilBERT model fingerprints comparison 148
C.7.5 Electra model fingerprints comparison 149

D Other Works 150

D.1 Syntax and Prejudice: Unveiling the Ethically Charged Biases of a Syntax-Based Hate Speech Recognizer 152
D.2 Change My Mind: A Syntax-Based Approach to Understanding Perspective-Driven Hate Speech 154
D.3 Every Time I Fire a Conversational Designer, the Performance of the Dialog System Goes Down 156
D.4 Lacking the embedding of a word? look it up into a traditional dictionary . . 159

D.5 Exploring Linguistic Properties of Monolingual BERTs with Typological Classification among Languages 162

List of Figures

2.1 Cancer Progression and Metastasis 13
2.2 The relationship between amino acid side chains and protein conformation . 16
2.3 Neoantigens presentation and cell activation 18
2.4 MHC-I structure 20
2.5 MHC pathway 21
2.6 Machine translation task using an attention mechanism 33
2.7 BERT training tasks 34
2.8 The Evolution of Language Models from 2018 to 2024 36
2.9 Hierarchy of pruning algorithms. 38

3.1 CARMEN design steps 42
3.2 CARMEN `main` dataset peptides distribution 45
3.3 KEN workpath 52
3.4 Visual representation on KEN parameter selection 54
3.5 Comparison between KEN and LoRA 59
3.6 KEN_{viz} graphical examples 63

4.1 Training and test corpora generation 69
4.2 Peptide length distribution on Neoantigen classification corpus 70
4.3 HLA-I class distribution on Neoantigen classification corpus 71
4.4 HLA-A distribution on Neoantigen classification corpus 72

4.5 HLA-B distribution on Neoantigen classification corpus 72
4.6 HLA-C distribution on Neoantigen classification corpus 73
4.7 Comparison of HLA subclass distributions in training and test set 73
4.8 Antigens in `ENSP00000497242` protein 74
4.9 Antigens and decoys in `ENSP00000497242` protein 76
4.10 Input tokenziation 79
4.11 KEN *extract* and *inject* model 82

5.1 TAPE experiments provided based on the corpora and the input given . . . 86
5.2 Training results 87
5.3 ROC curves using 1:1 and 1:5 unbalanced datasets 89
5.4 ROC curves using 1:1 and 1:5 balanced datasets 89
5.5 [Pfam sub-datasets composition]HLA class composition in the training, independent, and external subsets of the Pfam sub-datasets 93
5.6 KEN pruning results on *unbalanced* datasets 96
5.7 KEN pruning results on *balanced* datasets 97

A.1 Peptide length within CARMEN dataset 126
A.2 Spectral counts distribution 127
A.3 HLA-A sub-classes distribution 128
A.4 HLA-B sub-classes distribution 129
A.5 HLA-C sub-classes distribution 130

B.1 KEN vs random parameter selection 132
B.2 KEN_{viz} key attention matrix visualization on BERT layer 0 136
B.3 KEN_{viz} key attention matrix visualization on BERT layer 12 137

C.1 Linguistic Fingerprint: Workflow overview 140
C.2 Optimal subnetworks comparison 144

C.3 DeBerta layer 12 fingerprints comparison 147
C.4 Ernie layer 11 fingerprints comparison 147
C.5 BERT layer 12 fingerprints comparison 148
C.6 DistilBERT layer 5 fingerprints comparison 148
C.7 Electra layer 12 fingerprints comparison 149

D.1 KERM-HATE architecture 153
D.2 Contrasting trees 155
D.3 CLINN overview 158
D.4 DefiNNet and DefBERT 161
D.5 Spearman's correlation between European language pairs, ranked by typological features and BERT similarity matrices 165

List of Tables

2.1 Neoantigens overview: classification and characteristics 19

2.2 MHC-based Machine Learning Approaches 25

3.1 CARMEN corpus composition 43

3.2 TAPE pre-training paper results 47

3.3 Model performances on the five proposed tasks 48

3.4 BERT-based model pre-trained for Neoantigens presentation task 51

3.5 Properties of the analyzed models 56

3.6 KEN results using different percentages of trainable parameters on sentiment analysis tasks 57

3.7 Pruning algorithm comparations on SST-2 datasets 58

3.8 Comparation between KEN and FLOP pruning algorithm 61

3.9 KEN .pt file compression comparison 62

4.1 Training dataset 69

4.2 Testing dataset 69

4.3 Fragment of the resulting datasets used to generate the classification ones . . 69

4.4 Neoantigens classification datasets composition 78

4.5 Hyperparameters used in the TAPE training experiments 81

5.1 Testing results on the eight cases provided 88

5.2 TAPE vs MHCFlurry 92

5.3 Pfam dataset statistics 94

5.4 KEN pruning results on TAPE models 95

B.1 Analyzed datasets 133

B.2 Additional results of KEN applied to sentiment analysis datasets 134

C.1 Parameter reset and F1-weighted performances 142

C.2 Similarity percentages between subnetworks specific to language variation . . 145

Chapter 1

Introduction

Cancer remains a significant global health challenge, claiming one in eight lives worldwide [65]. In 2022, the World Health Organization (WHO) and the International Agency for Research on Cancer (IARC) reported an alarming 20 million new cases and 9.7 million cancer-related deaths. While these figures highlight the disease's burden, there is a glimmer of hope: 53.5 million individuals are surviving beyond five years post-diagnosis, thanks to advancements in treatment and care.

Breast cancer remains the most common type, with 2.3 million new cases in 2022, followed closely by lung, colorectal (1.46 million), prostate (1.1 million), and stomach (around 970.000) cancers. Notably, certain cancers, like melanoma, endometrial, and pancreatic, are on the rise. The global distribution of cancer is uneven, with Asia bearing the heaviest burden (6.4 million new cases in 2022), followed by Europe (3.5 million), North America (2.2 million), and Latin America and the Caribbean (1 million).

Concerning demographic shifts, colorectal cancer is increasingly affecting younger individuals and liver cancer rates are rising among women. Furthermore, it is predicted that there will be 28 million new cases annually by around 2040 if the incidence remains stable, and population growth and aging continue according to recent trends [31]. This represents a 54.9 % increase from 2020, with the increase expected to be higher in men (60.6 %) than

in women (48.8 %).

This necessitates further research to understand these trends and develop targeted interventions.

While effective in some cases, traditional treatments like surgery, radiotherapy, and chemotherapy often have limitations and adverse side effects [116]. Cancer immunotherapy, which stimulates the patient's immune system, has emerged as a promising approach [19]. Neoantigen-based vaccines, particularly, have shown great potential in enhancing T-cell responses and are considered a leading candidate for successful immunotherapy [42].

Neoantigens, unique peptides found exclusively in cancer cells, are crucial for personalized vaccine development. By training the patient's T-cells to recognize these neoantigens, the immune system can be activated to target and destroy cancer cells. The process involves several stages [86]:

1. Obtaining tissue samples
2. Sequencing DNA and/or RNA
3. Identifying and prioritizing neoantigens
4. Inducing T-cell recognition in vitro
5. Clinical evaluation of the vaccine

Software tools and pipelines have been developed to streamline neoantigen detection. Advancements in artificial intelligence, particularly transformer models, have further enhanced the accuracy and efficiency of this process. However, challenges remain, including the low activation rate of detected neoantigens, the need for integrated data analysis, and the importance of accurate peptide-MHC and pMHC-TCR binding predictions. Addressing these challenges is crucial to unlocking the full potential of neoantigen-based cancer immunotherapy.

1.1 Contributions

This thesis addresses several critical limitations that affect current machine learning models for neoantigen classification. By focusing on challenges that reduce the effectiveness, robustness, and scalability of these models, we aim to improve their predictive performance and practical applicability. Our contributions target key bottlenecks that hinder progress in this domain, specifically regarding the training datasets, model scalability, and task-specific fine-tuning.

In particular, we seek to address the following challenges:

1. **Suboptimal performance of general-purpose pre-trained models:** Most existing models rely on pre-training datasets covering diverse genomic topics, resulting in suboptimal performance when applied to specific tasks like neoantigen classification. To overcome this limitation, we developed models fine-tuned on specialized datasets focused on peptide-protein interactions and their binding affinities with the HLA-I antigen, specifically targeting the HLA-I subtypes A, B, and C. This fine-tuning improves the models' capacity to detect relevant peptides with greater accuracy than general-purpose alternatives.

2. **Dataset overlap and inflated performance metrics:** A common issue with many existing datasets is the presence of overlapping data between the training and test sets, as observed in subsets like Pfam. This overlap can lead to inflated performance metrics, creating a misleading impression of a model's true predictive power. To address this, we curated datasets to ensure a clear separation between training and test data, allowing for a more robust and accurate evaluation of our models' generalization abilities.

3. **Scalability and efficiency of large-scale models:** Many transformer-based models are not scalable due to their computational complexity, limiting their usability in practical applications. In response, we developed compact versions of our models, reducing

parameter size by over 40% through a custom pruning algorithm. This algorithm effectively extracts efficient sub-networks from larger models, maintaining performance while significantly enhancing computational efficiency.

In addition to tackling these specific challenges, this thesis implements all key stages necessary to develop machine learning models tailored for neoantigen classification. This includes the creation of synthetic peptides, generated by mimicking the structure and composition of proteins from which real peptides are derived, generating corpora of approximately 205 million data records used to fine-tune our large language models (LLMs). Our approach encompasses:

1. **State of the Art and Future Challenges**: A thorough analysis of the scientific literature to identify the latest trends, existing gaps, and challenges in the field.

2. **Creation Neoantigens Classification Corpora**: The development of specific corpora for studying neoantigens, and their conversion into classification tasks, essential for training machine learning models. In line with the latest advancements in artificial intelligence, various versions of these corpora will be created to obtain different models trained on varying amounts of data, enabling the analysis of final results and how they change.

3. **Selection and Optimization of a Transformer Model**: A meticulous search to identify the transformer model best suited to our needs, followed by a fine-tuning phase to adapt it to the specific task.

4. **Results Analysis**: A detailed analysis of the results obtained from the models, highlighting the key differences between the results of the various proposed model versions.

5. **Pruning and Optimization**: Finally, given the increasing size of machine learning models and their training datasets in recent years, we propose a pruning algorithm that seeks to find an optimal trade-off between performance and the number of parameters.

The goal here is to generate efficient models that utilize the minimum necessary parameters while achieving the same results as the original model, leveraging the model intrinsic structure. For each proposed model, we will seek its optimal sub-network and analyze its results.

The innovative component of this thesis lies in its comprehensive vision of an artificial intelligence project, developing and analyzing each step from the creation of datasets used during the training phase to the analysis of results obtained after training the selected transformer models. Moreover, through pruning, we add a new step to the classic development of machine learning projects, promising to obtain efficient models using the minimum necessary parameters.

Ultimately, the contributions of this thesis directly support the development of personalized cancer vaccines by enhancing the precision and reliability of neoantigen identification, a critical step in creating effective and individualized immunotherapies. By improving the fine-tuning of machine learning models on neoantigen-relevant datasets, curating non-overlapping data to prevent biased performance metrics, and optimizing model scalability, this work lays the groundwork for more accurate detection of patient-specific neoantigens. These advancements facilitate the selection of immunogenic peptides tailored to a patient's unique tumor profile, thereby increasing the likelihood of eliciting a robust and targeted immune response – a core objective in the design of personalized cancer vaccines. This contribution aligns closely with the broader goal of precision oncology, providing AI-driven insights that can improve clinical outcomes and enhance therapeutic efficacy.

This thesis contributes to the European Horizon 2020 project KATY: *"Knowledge at the Tips of Your Fingers"*. KATY aims to empower medical professionals and the biomedical community by providing easy access to AI-enhanced knowledge. By integrating various omics data and employing advanced artificial intelligence algorithms, KATY seeks to uncover groundbreaking insights into the underlying biology of clear cell renal cell carcinoma (ccRCC). The project goal is to develop a reliable and effective AI-supported clinical decision

support system. This system will assist healthcare providers in selecting the most suitable therapy for individual ccRCC patients. KATY researchers believe that their patient-centered approach has the potential to revolutionize oncology by enabling precise clinical and molecular characterization of patients and providing tailored treatment recommendations. This advancement will ultimately drive the adoption of personalized medicine guidelines.

1.2 Structure

This thesis is organized into 6 chapters and 4 appendices, presenting a clear and coherent research pathway.

- **Chapter 1 - Introduction**: This chapter introduces the research problem, highlighting the thesis's original contributions to the scientific community. It also provides an overview of published or ongoing research related to this study.

- **Chapter 2 - Background**: This chapter provides a solid theoretical foundation, divided into three main areas:

  (A) *Cancer Biology* (Sec. 2.1-2.3): A comprehensive exploration of the biological concepts central to this thesis, from the fundamentals of cancer to treatment options, with a particular focus on vaccines and renal cell carcinoma, the ultimate goal of the KATY project.

  (B) *Artificial Intelligence*: This section describes the primary AI tools and techniques employed in the research, such as the current application of Artifical intelligence for cancer vaccines development (Sec. 2.4), online repositories of proteomic data (Sec. 2.5), and provides a general overview of transformer models (Sec. 2.6), with a specific focus on BERT, the core architecture of the machine learning models used throughout this work (Sec. 2.6.1).

(C) *Pruning Algorithms*: This section introduces the fundamental concepts of pruning algorithms, which underpin the development of the proposed algorithm (Sec. 3.3.1).

- **Chapter 3 - Methodology**: This chapter delves into the theoretical aspects of the research conducted. It is divided into three sections, each focusing on a different aspect of the thesis:

  (A) *Corpus*: Sec. 3.1 analyzes the corpus used to create the corpora for the neoantigen prediction task, examining its composition and key features.

  (B) *Machine Learning Model*: Sec. 3.2 presents the architecture of the machine learning model used for all experiments, also briefly discussing other models created based on the same architecture.

  (C) *KEN*: Sec. 3.3 theoretically introduces our developed pruning algorithm and its application in a sentiment analysis case study, presenting the results.

- **Chapter 4 - Experiments**: The main of the thesis. This chapter provides a detailed account of each phase of the experiments that were conducted. The structure is divided into three main sections:

  (A) *Corpora Generation* (Sec. 4.1): This section describes the entire process of creating the corpora used for neoantigen classification. Starting from the corpus introduced in Chapter 3, the steps of pre-training, analysis, and generation of the various output corpus versions are illustrated.

  (B) *Machine Learning Model Configuration* (Sec. 4.2): This section is dedicated to describing the hyperparameters, tokenization processes, and input transformations used in the experiments.

  (C) *Pruning Algorithm Configuration* (Sec. 4.3): Similarly to the previous section, the parameters and experimental methodologies adopted are detailed.

- **Chapter 5 - Results**: This is the key chapter where the obtained results are presented and discussed. The results of the training and testing phases of the developed transformer models are analyzed in detail (Sec. 5.1). To demonstrate the effectiveness of the results, the developed models are compared with other reference models, trained on the same or different SOTA datasets (Sec. 5.1.3). Finally, the results obtained with the pruning algorithm are presented, identifying the optimal threshold for obtaining high-quality sub-networks (Sec. 4.2).

- **Chapter 6 - Conclusions**: This chapter summarizes the main results obtained, highlighting the original contributions of the thesis and opening up future perspectives.

Finally, the four appendices provide in-depth analyses or supplementary results that complement the main body of the thesis. In particular, Apx. D - Other Works - presents a summary of projects completed and published during the doctoral program that are not directly related to the central themes of this thesis. While not integral to the main topic, these works remain firmly rooted in the field of artificial intelligence, with a specific focus on Natural Language Processing, a research area of fundamental importance both for this thesis and for my research conducted over these years.

1.3 List of Published Papers

During my PhD program, which spanned from November 2021 to October 2024, I was the author of nine scientific articles. Of these, seven have been published, and two are currently undergoing review. Four of them represent the main part of this thesis, while the others are included in Apx D – Other Works.

Published papers

7 published articles: 6 in conference proceedings (3 in ACL venues and 3 in others) and 1 in a peer-reviewed journal.

- **ACL (Association for Computational Linguistics) venues**

    1. *Findings of ACL 2024*: Less is KEN: a universal and simple non-parametric pruning algorithm for large language models [94]
    2. *Findings of EMNLP 2023*: Exploring Linguistic Properties of Monolingual BERTs with Typological Classification among Languages [127]
    3. *Findings of ACL 2022*: Lacking the embedding of a word? look it up into a traditional dictionary [128]

- **Not ACL (Association for Computational Linguistics) venues**

    1. *NLPerspectives 2022 (as part of LREC/COLING 2024)*: Linguistic Fingerprint in Transformer Models: How Language Variation Influences Parameter Selection in Irony Detection [95]
    2. *NLPerspectives 2022 (as part of LREC 2022)*: Change my mind: How syntax-based hate speech recognizer can uncover hidden motivations based on different viewpoints [98]
    3. *Proceedings of the Thirteenth Language Resources and Evaluation Conference (LREC 2022)* Every time I fire a conversational designer, the performance of the dialogue system goes down [166]

- **Journal**

    1. *PeerJ Computer Science (vol. 8 / 2022)*: Syntax and prejudice: ethically-charged biases of a syntax-based hate speech recognizer unveiled [93]

Paper references within the thesis

The four articles incorporated into the thesis reflect the methodologies and experiments presented in this work. They can be categorized as follows:

1. **CARMEN dataset**: 1 article under review. Described in Sec 3.1.
2. **Machine learning for neoantigen presentation**: 1 article under review. Described in Sec 4.1
3. **Pruning algorithm**: 2 published articles described in Sec. 3.4 [94] and Apx C [95]

Conclusion

The papers presented in this thesis offer a significant contribution to the fields of artificial intelligence and natural language processing. The obtained results demonstrate the effectiveness of the proposed methodologies for improving natural language understanding and generation. In particular, the development of the CARMEN dataset and the analysis of transformer models have opened new avenues for creating more accurate and efficient language models [128, 127]. Additionally, the proposed pruning algorithm represents a substantial advancement in the state-of-the-art, allowing for reduced model complexity without compromising performance [94, 95]. Furthermore, the published articles not directly related to the thesis core focus add valuable milestones in the field of artificial intelligence beyond healthcare, such as hate speech recognition [93, 92], improvements of Transformer models [128, 127] and chatbots [166].

Chapter 2

Background

This section provides a comprehensive overview of the key concepts necessary for understanding the research presented in this thesis. It covers the biological basis of cancer, advances in cancer treatment, and the role of artificial intelligence in vaccine development.

The chapter begins with a thorough analysis of the main subject of the thesis: *the cancer*. This includes an exploration of the biological mechanisms underlying a cancer genome and its progression (Sec. 2.1.1), as well as a review of the latest treatments and future approaches (Sec. 2.1.2).

Subsequently, two crucial biological concepts central to the thesis are presented: Neoantigens (Sec. 2.2.2) and MHC-I (Sec. 2.2.3). The sections dedicated to these topics delve into their nature and significance within the context of our study.

Sec. 2.3 shifts attention to peptide-based vaccine development, providing a general overview and specifically addressing its application to clear cell renal cell carcinoma (ccRCC) (Sec. 2.3.1). To achieve this goal, Sec. 2.4 focuses to the development of cancer vaccines using artificial intelligence, showcasing how these approaches have gained momentum in recent years, especially in studies centered on MHC. Finally, Sec. 2.5 discusses about the repositories used to train AI models for biomedical tasks: PRIDE (Sec. 2.5.1) and MassIVE (Sec. 2.5.2).

The rest of this chapter focuses on machine learning and the methods used to optimize these models, which are growing in terms of memory and parameters. It begins with a brief introduction of transformer models (Sec. 2.6) and BERT-based models (Sec. 2.6.1). The chapter then delves into pruning algorithms, discussed in Section 2.7, covering their classification and methodologies (Sec. 2.7.1), with particular attention given to the latest approach in this field known as *The Winner Ticket lottery Hypothesis* (Sec. 2.7.2).

2.1 The cancer genome

Cancer is a complex disease characterized by uncontrolled cell growth, invasion of surrounding tissues, and the potential spread to distant organs — a process known as *metastasis*. It is a significant global health burden, affecting approximately one in eight people worldwide [146]. In 2020, there were about 19.3 million new cancer cases (excluding non-melanoma skin cancer) and roughly 10 million cancer-related deaths globally. There are over 100 distinct cancer types, each with unique characteristics, risk factors, and treatment challenges.

According to the World Health Organization, the most common cancers in 2020 based on new cases were: breast (2.26 million cases), lung (2.21 million), colon and rectal (1.93 million), prostate (1.41 million), skin (non-melanoma, 1.20 million), and stomach (1.09 million). Lung cancer was the primary cause of cancer-related deaths (1.80 million deaths), followed by colon and rectal (916,000), liver (830,000), stomach (769,000), and breast cancers (685,000).

Insights into the genetic basis of cancer began to emerge in the late 19th and early 20th centuries. Microscopic examination of cancer cells revealed abnormal chromosomal aberrations. These observations led to the hypothesis that cancers are - essentially - cell clones driven by and characterized by abnormalities in hereditary material [157, 20]. Subsequent discoveries of DNA as the genetic material [97] and its double-helical structure [162] solidified this concept. Studies also showed that DNA-damaging agents can promote cancer development [83].

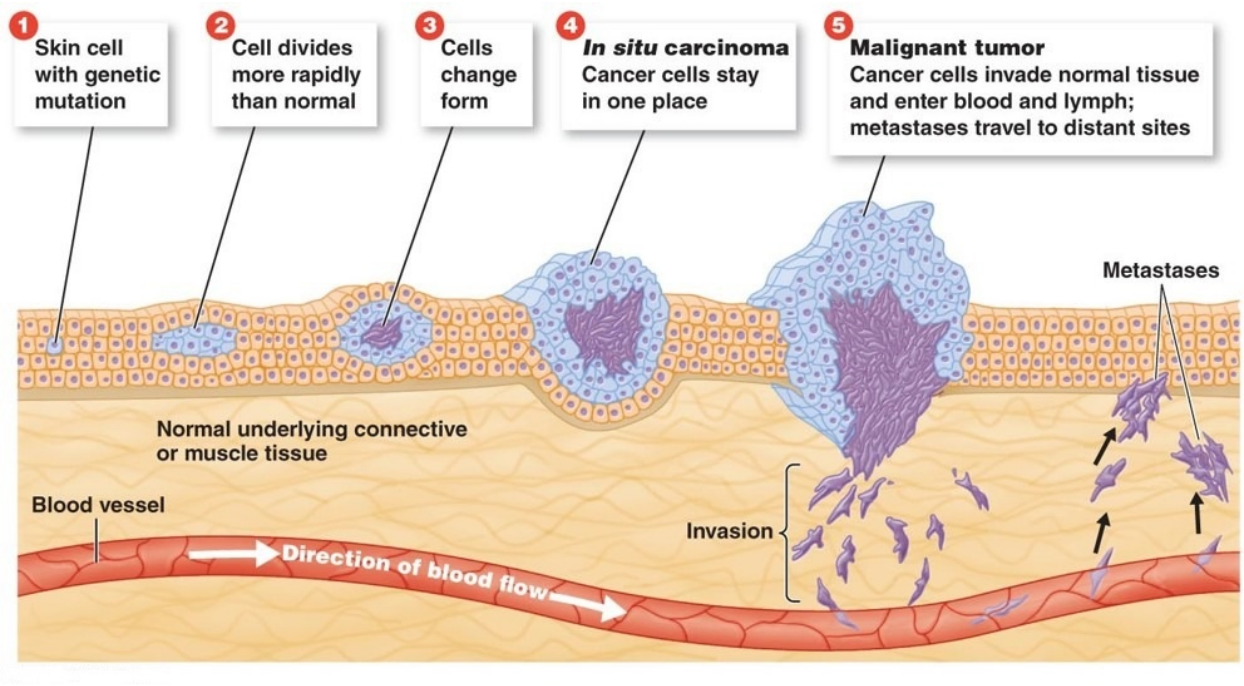


Figure 2.1: Cancer Progression and Metastasis. This diagram illustrates the complex process of cancer development, beginning with initial genetic alterations in cells (1), through stages of *hyperplasia* (2) and *dysplasia* (3), leading to in *situ cancer* (4), and culminating in the formation of *malignant tumors* (5) with its spread to distant sites (*metastases*).

2.1.1 The Genetic Landscape of Cancer

Cancer arises from the accumulation of genetic alterations, known as mutations, within normal cells. Various factors can trigger these mutations, including random errors during cell division, inherited genetic predispositions, and environmental exposures. Over time, mutated cells begin to proliferate, dividing at a faster rate than normal and resulting in the growth of extra cells (*hyperplasia*). However, at this stage, the cells and tissue structure appear normal under microscopic examination. After this stage, cancer evolves into *dysplasia*, which represents a more severe condition characterized by abnormal cell growth and disrupted tissue organization. Unlike hyperplasia, at the dysplasia stage, cells appear abnormal and there are alterations in the tissue organization. Subsequently, cancer progresses to the in situ stage, where it becomes fully formed and detectable under a microscope. From here, the cancer can invade surrounding tissues and spread to distant organs, a process known as

metastasis.

A visual representation of these stages is provided in Fig.2.1.

Cancer cells display specific biological characteristics, known as *hallmarks* of cancer [54]. These hallmarks include continuous signals for cell proliferation, resistance to growth suppressors, evasion of cell death, sustained replicative potential, promotion of blood vessel growth (angiogenesis), and activation of invasion and metastasis. The tumor microenvironment, comprising cancer cells, immune cells, blood vessels, and extracellular matrix, plays a crucial role in cancer advancement [47]. It influences tumor growth, spread to other parts of the body, and response to therapy. Understanding the complex interactions within the tumor microenvironment is essential for developing effective treatments.

Carcinogens, which promote cancer development, exert their influence through various mechanisms. For example, tobacco smoke reduces DNA repair in the lungs [150] and creates an environment favorable for tumor growth, while *Helicobacter pylori* (causative of gastric ulcers) induces inflammation that contributes to stomach cancer risk [117, 22]. This resulting inflammation may increase the risk of stomach cancer by both elevating DNA-damaging agents in the tissue and creating an abnormal tissue environment favoring cancer development. Advanced cancers are complex ecosystems consisting of diverse cancer cell populations, each with its own unique genetic makeup. This diversity poses a significant challenge for effective treatment.

2.1.2 Cancer Treatment: Current and Future Approaches

Despite significant progress in cancer research, finding a cure remains a major challenge. The current focus of treatment is primarily on managing tumor growth and enhancing patients' quality of life.

For decades, cytotoxic chemotherapy has been a cornerstone of cancer treatment [104]. However, its effectiveness is limited by side effects and the development of drug resistance. Advances in cancer biology understanding, have led to the creation of targeted therapies that

specifically inhibit key cancer-driving proteins. While these therapies have shown promise for certain types of cancer, such as *chronic myeloid leukemia*, resistance to treatment remains a significant obstacle.

Immunotherapy is another approach that harnesses the body's immune system to combat cancer, holding great potential. By triggering the immune response against tumor cells, immunotherapy can lead to long-lasting improvements in some patients. Nevertheless, not all patients benefit from this treatment, and further research is required to maximize its effectiveness.

The future of cancer treatment lies in personalized medicine, which aims to customize therapies based on individual patient's genetic makeup and tumor characteristics. Identifying specific genetic alterations driving tumor growth enables researchers to create targeted treatments with fewer side effects and greater efficacy. Additionally, comprehending the intricate interaction between cancer cells and the tumor microenvironment will be pivotal for devising innovative therapeutic strategies.

2.2 Molecular Mechanisms of Cancer Immunogenicity

The immune system is a complex network of cells and molecules that defend the body against pathogens and aberrant cells, including cancer. A critical component of this defense mechanism involves the recognition and elimination of cancer-specific antigens, known as *neoantigens*. These unique molecular markers, generated by somatic mutations in tumor cells, are presented on the cell surface by major histocompatibility complex (MHC) class I molecules.

The upcoming sections will provide detailed explanations of all the listed molecular mechanisms starting with a fundamental description of amino acids, peptides, and proteins (Sec. 2.2.1). Then it examines neoantigens (Sec. 2.2.2) and MHC class I molecules (Sec. 2.2.3), offering a foundational understanding of their roles in the immune response to cancer.

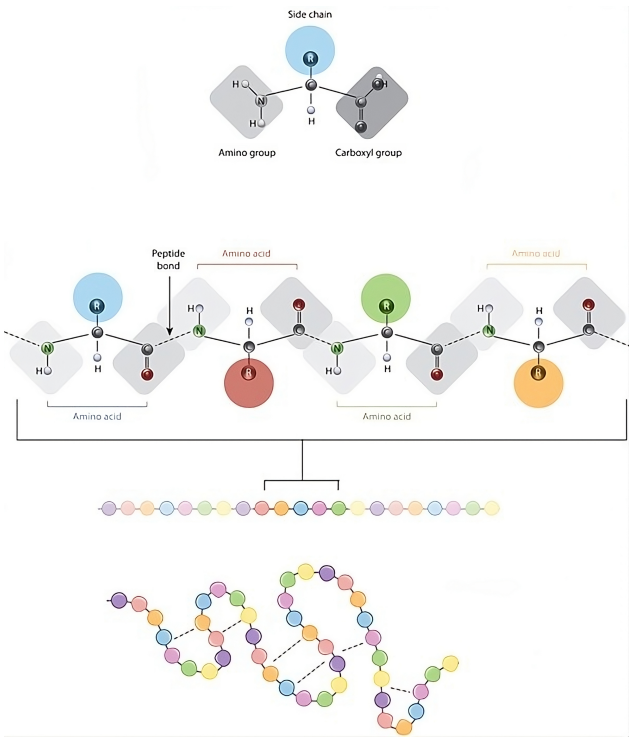


Figure 2.2: Relationship between amino acid side chains and protein conformation. Amino acids are defined by their side chains, represented by colored circles. When linked by peptide bonds, they form a polypeptide, also known as a protein, which then folds into a specific conformation based on interactions (dashed lines) between the amino acid side chains. Source [111]

2.2.1 Proteins, Peptides, and Amino Acids

Proteins are remarkable molecules essential for life, performing diverse functions from tissue construction to catalyzing biochemical reactions. These complex structures are assembled from smaller units called amino acids and peptides.

Amino acids are the foundational components of proteins. Each contains a central α-carbon atom bonded to an amino group, a carboxylic acid group, and a unique side chain (R group). The R group imparts distinct properties to each amino acid. While most proteins utilize L-amino acids, D-amino acids play roles in bacterial cell walls and certain antibiotics.

Peptides form when amino acids connect through peptide bonds, linking the carboxyl group of one amino acid to the amino group of another. Dipeptides consist of two amino acids joined by a single peptide bond. As more amino acids join, longer chains called polypeptides

emerge. Proteins are essentially complex polypeptides composed of typically over 50 amino acids.

The structure of a protein is intimately linked to its function, determined largely by the R groups of its constituent amino acids. Hydrophobic R groups tend to cluster within the protein's interior, while hydrophilic R groups often reside on the surface and participate in active sites. For instance, histidine's imidazole ring acts as both an acid donor and acceptor, crucial for many enzyme active sites. Cysteine, with its thiol group, functions as a nucleophile in enzymatic reactions and can form disulfide bonds (cystine bridges) that contribute to protein structure and stability.

A protein's unique amino acid sequence dictates its shape, which is acquired through a folding process involving the twisting and bending of the polypeptide chain. Protein structure is categorized into four levels: primary (amino acid sequence), secondary (local folding patterns like alpha helices and beta sheets), tertiary (overall 3D structure), and quaternary (assembly of multiple polypeptide chains). Disrupting this intricate folding process can render a protein inactive.

With an estimated 100,000 different proteins in the human body alone, their diversity underscores their vital role in life's processes. Understanding how amino acids combine to create functional proteins is essential for comprehending the intricacies of life itself.

2.2.2 Neoantigens

Antigens are substances that stimulate an immune response, prompting the production of antibodies to neutralize foreign substances. *Neoantigens* are specific antigens referring to cancer cells, resulting through various mechanisms, such as genomic mutation, aberrant transcriptomic variants, viral open reading frames (ORFs) [40, 159] and post-translational modifications (PTMs) [143, 160, 165].

These mutations modify the cell's DNA, causing the production of abnormal RNA and, subsequently, proteins with altered amino acid sequences. These modified proteins, known

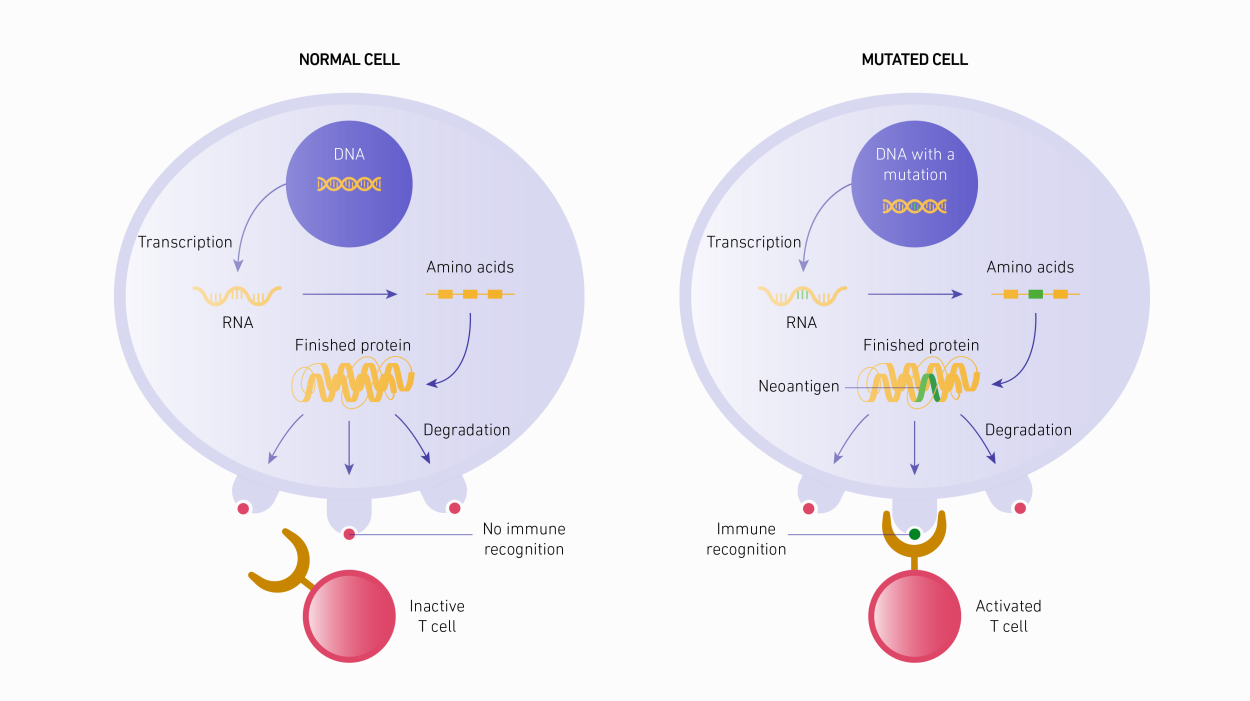


Figure 2.3: Neoantigens presentation and cell activation. Differences between antigen presentation in a normal cell (left) and neoantigen production and presentation in a mutated cell (right). Source: [134]

as neoantigens, are presented on the cell surface by MHC molecules. The immune system's T cells identify these neoantigens as foreign, initiating an immune response against them (Fig. 2.3).

Neoantigens can be classified into two main categories: *shared* and *personalized* [138, 129]. Shared neoantigens occur in multiple patients with the same cancer type, often due to common genetic alterations, while personalized neoantigens are specific to individual patients, reflecting their tumor's unique genetic makeup. Furthermore, antigens can be categorized based on their pattern of expression: *Tumor-Specific Antigens* (TSAs), which are present only on tumor cells and not on any other cell and *Tumor-Associated Antigens* (TAAs), which are present on some tumor cells and also some normal cells. In addition to this classification, neoantigens are ranked based on their variations, source, and the relevant types of cancer as described in Table 2.1 [165].

Neoantigens offer a distinct advantage over other tumor antigens like cancer-testis anti-

Neoantigens calssification	Sources	Advantages	Disadvantages	Relevant cancer
Genomic variants	Single-nucleotide variants (SNVs)	Simple prediction	- Similar to self-antigen - Rarely shared between patients	- Melanoma - Glioblastoma - Lung cancer - Bladder cancer
	Insertions and deletions (INDEL)	- More potential targets per mutation - Dissimilar from self-antigen	Relatively low burden	- MSI-H tumors - Renal cell carcinoma
	Fusion genes	- Dissimilar from self-antigen - Shared targets between tumors - More potential targets per mutation - More immunogenic	Relatively low burden	- Acute myelocytic leukemia - Acute lymphocytic leukemia - Chronic myeloid leukemia - Sarcomas
	Chromosomal rearragements	High immunogenicity	Less well studied	Malignant pleural mesothelioma
Transcriptomic variants	RNA splicing	- A large number of predicted targets - More dissimilar from self-antigen	- Fewer tools available - Not well validated in pre-clinical models - Current tools do not account for Nonsense Mediated Decay (NMD)	- Acute myelocytic leukemia - Chronic myelomonocytic leukemia - Chronic lymphocytic leukemia - Myelodysplastic syndrome
	Polyadenylation (pA) and RNA editing	Easy prediction	Less well studied	- Chronic lymphocytic leukemia
	Allegedly non-coding regions	- Relatively high burden - More potential targets	- Less well studied - Fewer tools avaiable	- Acute lymphoblastic leukemias - Lung cancer
Proteomic variants	Post-translational modifications (PTMs)	Shared between patients	Less well studied	- Leukemia - Renal cancer - Non-small cell lung cancer
	Proteasome processing	High specificity	- Less well studied - Fewer tools avaiable	Acute myeloid leukemia
	T-cell epitopes associated with impaired peptide processing (TEIPP)	TEIPP-specific T-cell can escape thymic selection	- Less well studied - Limited in HLA-I low or TAP-deficient tumors - limited in specific tumors	Lung cancer
Viral-derived neoantigens	Viral open reading frames	- High immunogenicity - More dissimilar from self-antigen - Shared between patients - Without apparent toxicity to normal tissues	Limited specific tumors	- Hepatocellular canciroma - Merkel cell canciroma - Nasopharygeal canciroma - Head cancer - Neck cancer - Cervical cancer - Anal cancers

Table 2.1: Neoantigens overview: classification and characteristics

gens (CTAs) and TAAs due to their exclusive presence on cancer cells and absence in healthy tissues. This makes them prime targets for personalized cancer treatments [63, 21, 165]. Their unique nature allows T cells to evade the body's natural tolerance mechanisms, resulting in a larger pool of neoantigen-specific T cells and a stronger anti-tumor immune response. Moreover, the potential for long-term immunological memory following immunotherapy-induced neoantigen-specific T-cell responses offers hope for durable cancer protection.

Advancements in tumor gene sequencing, neoantigen identification, and neoantigen-based immune therapies are driving the development of personalized cancer vaccines (PCVs) and adoptive cell therapy (ACT).

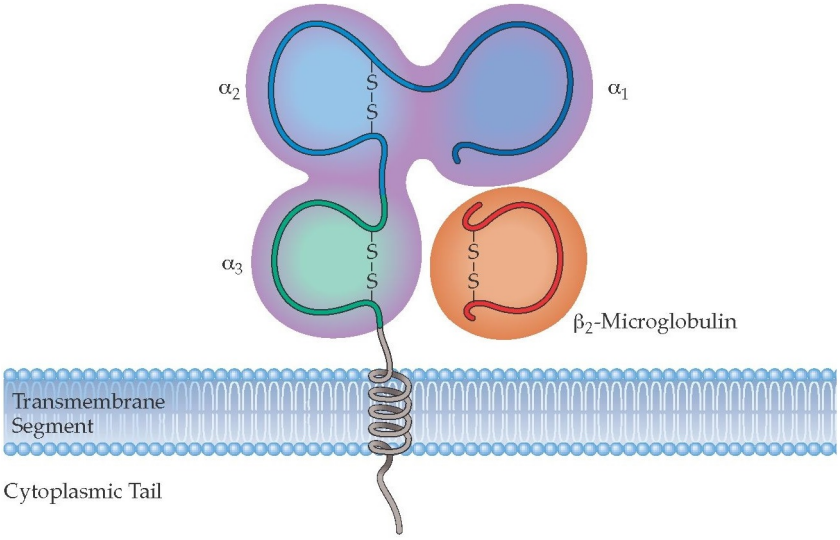


Figure 2.4: MHC-I structure

2.2.3 Major Histocompatibility Complex I

The major histocompatibility complex (MHC) class I gene family plays a pivotal role in the immune system by encoding proteins found on most nucleated cells. These proteins, known as MHC class I molecules, act as cellular flags, presenting protein fragments to the immune system, which helps distinguish between healthy "self" and abnormal proteins from pathogens, tumors, or dying cells. Moreover, MHC class I molecules play a role in educating cytotoxic T lymphocytes (CTLs), a crucial part of the adaptive immune system.

Interestingly, most of the peptides presented by MHC class I molecules come from incompletely synthesized or degraded proteins rather than normal cellular proteins [137, 121]. This mechanism allows for rapid cellular surveillance, enabling CTLs to quickly identify and eliminate infected or abnormal cells.

MHC class I molecules are complex structures consisting of two polypeptide chains and a bound peptide. The alpha chain (α), which is encoded by MHC genes, and beta-2-microglobulin (β_2M) - a non-MHC-linked protein, come together to form a *heterotrimer*. The stability and presentation of the molecule on the cell surface heavily rely on a peptide bound within it. As shown in Fig. 2.4, the alpha chain contributes three domains (α_1, α_2,

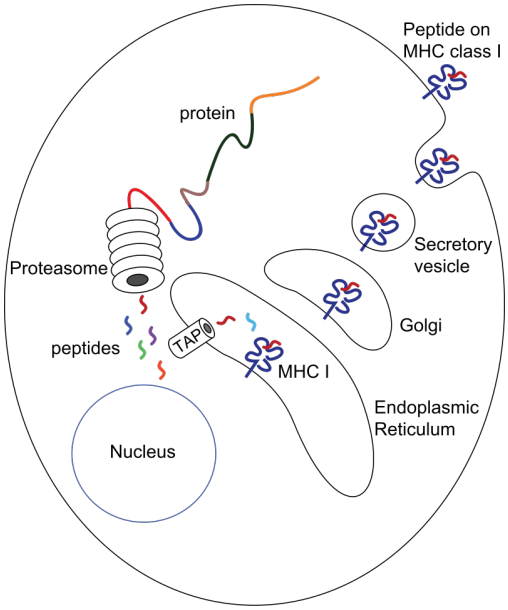


Figure 2.5: MHC pathway

and α_3), while β_2M forms a single domain. Together, these domains create a peptide-binding groove.

The assembly of MHC class I takes place in the endoplasmic reticulum (ER), where the molecule undergoes folding and acquires a peptide. Specialized transporters associated with antigen processing (TAP) [1, 154] are crucial in transporting peptides from the cytosol to the ER, and they also play a role in the final assembly of MHC class I. Chaperone molecules assist in folding and, together with the MHC class I molecule, β_2M, and TAP, form a complex that is stabilized by *tapasin* [163]. Tapasin selects high-affinity peptides for binding. Once a suitable peptide is bound, the MHC class I-peptide complex is transported through the Golgi apparatus [145] to the cell surface for immune surveillance (Fig. 2.5).

The HLA-I region is composed of three distinct groups: HLA-A, HLA-B, and HLA-C. These groups are distinguished by their high allelic variability, with currently identified alleles of 108 HLA-A, 223 HLA-B, and 67 HLA-C [17, 91, 67]. This variability suggests a functional specialization among the loci. The HLA nomenclature reflects this complexity; while serology uses the simple designations HLA-A, HLA-B, and HLA-C, molecular biology employs a more detailed system that includes an asterisk (e.g., HLA-A*) followed by a

series of digits to specify the exact allele. This precision is essential in contexts like organ transplantation and the study of disease associations.

Functionally, HLA-A, HLA-B, and HLA-C molecules play a critical role in presenting antigenic peptides to CD8+ cytotoxic T cells, which is a key process in cellular immunity. Nevertheless, there are specific differences among these molecules:

- Genetic Variability: Polymorphism is highest in HLA-B, followed by HLA-A, and then HLA-C [124]. This contributes to the diversity of the immune system but complicates the transplantation process.
- Expression Levels: HLA-A is generally more highly expressed on the cell surface compared to HLA-B and HLA-C [6].
- Disease Associations: Certain alleles are linked to autoimmune diseases, such as HLA-B27 and ankylosing spondylitis, or HLA-C and psoriasis [48].
- Transplantation Role: HLA compatibility is crucial, and mismatches in HLA-B can significantly affect transplant outcomes [44].
- Immune Cell Interactions: While all three interact with CD8+ T cells, there may be subtle differences in how they interact with specific receptors [114].

2.3 Peptide-based vaccines

Peptide-based vaccines offer significant potential for cancer immunotherapy, yet their clinical success remains limited. Current research focuses on optimizing vaccine design to elicit robust anti-tumor immune responses. Traditional vaccines are designed to activate a specific part of the immune system, such as *B lymphocytes* (B-cells), which produce antibodies, or *T lymphocytes* (T-cells), which directly eliminate tumor cells.

One of the main advantages of peptide vaccines is their ability to trigger powerful T-cell responses, leading to clinical benefits for certain patients [30]. Their adaptable design allows

for the incorporation of multiple MHC-I epitopes in a single vaccine, making them suitable for diverse patient populations [30]. This wide-ranging approach enhances the potential for effective immune responses across different individuals. Additionally, multi-epitope vaccines can help guard against tumor evasion by stimulating a varied T-cell repertoire [108, 72, 173, 140]. However, for optimal effectiveness, these epitopes must come from different proteins. Peptide vaccines can also include MHC class II and B-cell epitopes to activate helper T cells and stimulate antibody responses.

Despite these advantages, challenges persist. Vaccine efficacy relies heavily on the tumor expressing targeted antigens, necessitating immunoproteomic approaches for antigen discovery and selection [30]. Additionally, peptide-based vaccination can induce immunosuppressive *regulatory T cells* (Tregs), hindering vaccine effectiveness [80, 14, 16]. Complete tumor eradication, especially in advanced disease, may require more than peptide vaccines alone.

Fortunately, strategies to address these limitations exist. Identifying more immunogenic peptides through immunoproteomics and advanced adjuvant plus cytokine therapies holds promise for developing more effective vaccine regimens.

2.3.1 Peptide-based Vaccines for Renal Cell Carcinoma

Renal cell carcinoma (RCC) is a common and aggressive adult cancer with a poor prognosis for advanced cases (less than 10% five-year survival rate) [136]. Vascular endothelial growth factor receptor 1 (VEGFR1) plays a critical role in RCC progression, making it a promising target for T cell-based immunotherapies using peptide vaccines.

Early clinical trials showed promise with a two-peptide VEGFR1 vaccine targeting HLA-A2 and HLA-A24 epitopes. This vaccine, used with Montanide ISA-51 adjuvant, elicited specific CD8+ T cell responses with $IFN - \gamma$ secretion in 15 out of 18 patients [169]. Encouragingly, two patients achieved partial responses, and nine demonstrated stable disease for at least five months [169].

Another approach utilized immunoproteomics to identify nine HLA-A2-restricted epitopes from RCC patient samples [158]. These epitopes were incorporated into the IMA901 vaccine, which was injected intradermally with GM-CSF in RCC patients. This strategy resulted in CD8+ T cell responses against multiple antigens, correlating with disease control [158]. Importantly, the inclusion of cyclophosphamide three days before IMA901 administration further improved outcomes, highlighting the crucial role of Tregs within the tumor microenvironment. Peptide-based vaccines may benefit from Treg depletion before injection, or additional strategies to modulate the immunosuppressive environment, potentially through cytokine treatment. However, caution is needed in selecting cytokines, as evidenced by trials combining DC vaccines with interleukin (IL)-2, which transiently induced significantly higher Treg levels [80, 14].

2.4 Artificial Intelligence in Cancer Vaccines Development

The use of machine learning models in understanding biological systems has become crucial. In the pharmaceutical industry, there is a growing reliance on computational approaches to optimize and predict therapeutic outcomes at the organism level, highlighting the vital role of computational biology in this field [77].

Understanding the behavior of cancer vaccines requires a deep comprehension of the immune system, one of the most complex biological systems. Both basic and applied immunological research are faced with this intricate challenge. As a response, computational immunologists are increasingly turning to mathematical modeling and computer simulation to study the immune system and its response to pathogens [59, 115].

In vaccination systems, machine learning techniques are primarily used to train on a set of known data and identify the best combination of genes and parameters to activate the immune system and predict vaccination outcomes.

Paper title	Goal	Result
Predicting HLA class II antigen presentation through integrated deep learning [24]	Predicting the presence of HLA-II antigen through integrated deep learning	- MARIA: ability to identify immunological epitopes - Model specificity: 99.5% - Accuracy: 38.7%
Antitumor effects of an iPSC-based cancer vaccine in pancreatic cancer [110]	Checking anticancer vaccine effects based on iPCS in pancreas cancer using ML algorithms	- Increase vaccine immunization C+ I - Checking comparison cytotoxic T cell CD8+ and PBS control - Predicting T cell and anticancer CD4+ affects - Preventing tumor generation of pancreas cancer till 75%
Role of in silico structural modeling in predicting immunogenic neoepitopes for cancer vaccine development [171]	- Development of a peptide vaccine that targets the predicted neoantigens by the MHC binding algorithm pipeline called NetMHC - Investigation of structural changes in and around the MHC junction groove selected by the minimum neoepitopes and the effect of mutation as function of its location	- Improving the prognosis of neoepitopes for cancer vaccine based on structural parameters including SASA and location of mutation in a sequence
MHCSeqNet: a deep neural network model for universal MHC binding prediction [167]	MHC binding prediction and MHC ligand dataset	Probability of peptide binding and MHC based on input (MHC allele and peptides)
Predicting HLA-I peptide immunogenicity with deep learning and molecular dynamics [71]	- Determining the characteristics related to immunogenicity of peptides - Integrated molecular dynamics - Display human sequences - Pathogens and tumor peptides expressed by HLA-I	- Identification of structural and dynamic features by the model that is correlated with immunogenicity and prediction and classification of cancer neoantigens and immunogenic neoepitopes in melanoma patients - AUC: 0.85 - Accuracy: The model identified immunogenic peptides in the range between [0.6, 0.7]
Systematically benchmarking peptide-MHC binding predictors: From synthetic to naturally processed epitopes [175]	Identification of immunogenic T cell epitopes based on binding affinity to MHC-I and MHC-II	- Artificial intelligence approaches performance are better than regression-based ML - Among the benchmarked predictors, MHCFlurry and NN-align performance are the best for NHC call 19-MER and class 15-MER - NetMHCpan also had considerable predictive power - Custom MHCFlurry for pan-HLA predictors shows the same accuracy as NetMHCpan - The most accepted approach is to reply on ML classifiers based on complex peptide data or large-scale MHC training - Provide accuracy criteria, identify the best algorithms and their limitations - Investigate the T cell response to the predicted antigens and provide a comparative table of accuracy in the prediction tools, as well as, a table of the technologies used in each model and its details

Table 2.2: Latest ML approaches for MHC-based prediction and cancer vaccine development. Table development inspired by [61]

2.4.1 MHC-based Machine Learning Approaches

Recent progress in computational immunology has had a significant impact on cancer research, especially in the development of effective immunotherapies. One of the main focuses has been on understanding and predicting MHC-peptide interactions (Sec. 2.2.3), which play a crucial role in antigen presentation and activating the immune response.

Machine learning models have become powerful tools for addressing this challenge. By analyzing large datasets of MHC-peptide binding affinities, amino acid sequences, and clinical outcomes, these models aim to identify immunogenic peptides that can be utilized in

designing cancer vaccines. Additionally, they can aid in predicting patient responses to immunotherapy, enabling the development of personalized treatment strategies.

The following section offers a comprehensive comparison of state-of-the-art methods for predicting MHC-peptide interactions. Key features of these models are summarized in Tab. 2.2 for easy reference.

MARIA (Major Histocompatibility Complex Analysis with Recurrent Integrated Architecture) In the study conducted by Chen et al. [24], they introduced a novel deep learning framework named MARIA (Major Histocompatibility Complex Analysis with Recurrent Integrated Architecture). The primary function of MARIA is to predict how human leukocyte antigen (HLA) class II molecules present antigens, which is a vital process in understanding how the immune system recognizes and targets specific antigens.

MARIA stands out from existing methods by leveraging a variety of data sources, including in vitro binding measurements, mass spectrometry data, and gene expression levels. It utilizes a recurrent architecture, specifically LSTM (Long Short-Term Memory), to efficiently process peptide sequences of varying lengths.

One of the significant contributions of MARIA is its ability to predict the probability of antigen presentation by HLA class II molecules. This prediction is a critical step in the development of vaccines and cancer immunotherapies. With a model specificity of 99.5%, MARIA demonstrates its capability to identify immunological epitopes across various cancers and autoimmune diseases.

Checking anti-cancer vaccine effects based on iPCSs Ouyang et al. [110] explore the potential of induced pluripotent stem cells (iPSCs) as a vaccine platform specifically for pancreatic cancer. They demonstrate that a vaccine based on iPSCs, when paired with a particular immune stimulant, effectively initiates cytotoxic T cell and humoral immune responses against cancer in mice.

The researchers employ machine learning algorithms, namely flowsom, NetNHC, and

MARIA, to pinpoint potential vaccine targets within a group of 111 cancer-associated genes. The findings of the study suggest an enhancement in vaccine immunization and a reduction in pancreatic cancer tumor generation by up to 75%.

The Role of Structural Information in Peptide Vaccine Development In their research, Zaidi et al. [171] concentrate on the creation of peptide vaccines that target neoantigens. They highlight the shortcomings of conventional methods that depend solely on the predicted binding affinity between neoantigens and molecules of the immune system (MHC) through NetMHC.

The researchers propose the inclusion of structural information in neoantigen prediction, arguing that this strategy could improve the identification of immunogenic epitopes for vaccine design. The findings of the study emphasize the significance of considering structural parameters, such as the solvent-accessible surface area (SASA) and the location of mutations in sequence modeling.

MHCseqNet: A Deep Learning Approach to MHC Binding Prediction and Nanomedicine Design Yamanluirt et al. [167] present the MHCseqNet, a deep learning model designed for MHC binding prediction and MHC ligand dataset. The model employs a neural network and RNN-based Natural Language Processing (NLP) to represent amino acid sequences and MHC epitopes.

The unique aspect of the model is its input treatment, where peptides are considered as sentences and amino acids as individual words. This approach allows the model to accommodate MHC alleles and peptides of any length, enhancing its versatility.

The study reveals that only a minuscule fraction of the nanomedicine-design space has been explored. This limitation is attributed to the structural complexity of nanomedicines and the absence of relevant high-throughput synthesis and analysis methods.

ACPS: A Novel Tool for Precision-Based Anti-Cancer Peptide Identification In their study, Kaushik et al. [71] strive to identify the characteristics associated with the immunogenicity of peptides. They integrate molecular dynamics and display human sequences, as well as pathogen and tumor peptides expressed by HLA-I. To predict immunogenicity, they utilize a convolutional neural network. This model identifies structural and dynamic features correlated with immunogenicity and predicts and classifies cancer neoantigens and immunogenic neoepitopes in melanoma patients with an AUC of 0.85. The model identified immunogenic peptides in the range between 0.6 and 0.7.

The study emphasizes the critical role of anti-cancer targets in cellular signaling processes. Engaging these targets without affecting the native cellular function is nearly impossible. Therefore, an approach has been developed to create an Anti-Cancer Scanner (ACPS) tool aimed at recognizing anti-cancer marks in the form of peptides.

The proposed ACPS tool allows for rapid fingerprinting of anti-cancer targets, which is of extreme significance in current bioinformatics research. While there are existing tools that offer these features on a single platform, the performance of ACPS was compared with these preexisting online tools and was observed to offer greater than 95% accuracy, which is comparatively much higher.

The ACPS scans the anti-cancer marked sequences of proteins supplied by the operators against the anti-cancer target datasets and provides precision-based anti-cancer peptides. The proposed tool, contrived in the PERL programming language, is an extended version of A-CaMP codes. These codes are highly scalable with an extensible application in cancer biology and robust coding architecture. The availability of tools like ACPS will greatly benefit researchers in the field of oncology and structure-based drug design.

Machine Learning benchmarking peptide-MHC binding predictors Zaidi et al. [175] embarked on a journey to pinpoint immunogenic T cell epitopes by evaluating their binding affinity to MHC class I and II. They leveraged a database of MHC class I and II,

and employed various machine-based predictors.

The findings of the study revealed that artificial neural network approaches outperformed regression-based machine learning methods. Among the predictors, the ML model-based MHCFlurry and neural networks emerged as the top performers for MHC class I and class II, respectively. NetNHcpan also demonstrated significant predictive power, and custom MHCFlurry for pan-HLA predictors exhibited the same accuracy as NetMHCpan.

The study established accuracy criteria and identified the best algorithms along with their limitations. It is widely acknowledged that the most preferred approach is to depend on machine learning classifiers based on complex peptide data or large-scale MHC training.

2.5 Proteomics repositories

Proteomics repositories play a crucial role in advancing proteomics research by providing a centralized platform for the storage, sharing, and analysis of mass spectrometry data. These repositories offer researchers access to a vast collection of protein and peptide identifications, enabling them to compare and contrast their findings with existing knowledge.

Two prominent proteomics repositories are PRIDE and MassIVE. Sec. 2.5.1 delves deeper into PRIDE, exploring its comprehensive database of proteomics identifications, including experiment details, literature references, protein and peptide sequences, and sample descriptions. Sec. 2.5.2, instead, focuses on MassIVE, discussing its specialized role in reproducible quantitative mass spectrometry-based proteomics, including its hierarchical structure for data storage, comprehensive analysis tools, and commitment to data reproducibility.

By utilizing these repositories, researchers can benefit from a wealth of publicly accessible data, facilitating the discovery of new biological insights and accelerating the pace of proteomics research.

2.5.1 PRoteomics IDEntifications database (PRIDE)

PRIDE [89, 68] is a public repository for proteomics identifications, offering a comprehensive database of protein and peptide identifications. These identifications are typically associated with specific species, tissues, subcellular locations, and may be related to particular disease conditions.

PRIDE includes a significant dataset from the HUPO Plasma Proteome Project [109] and a profile of the human platelet proteome [90]. The database is accessible through a web application, allowing for easy submission, searching, and data retrieval. Users can search by experiment accession number, protein accession number, literature reference, and sample parameters. Data can be retrieved in various formats, including machine-readable PRIDE or mzData XML and human-readable HTML.

PRIDE aims to be a central resource for proteomics data, similar to ArrayExpress for microarray data. Other publicly available proteomics databases include Global Proteome Machine Database (gpmDB) [32], PeptideAtlas [37], and Open Proteomics Database [118].

PRIDE stores the following information:

1. Experiment details, including title, description, and contact information
2. Literature references
3. Protein and peptide identifications with corresponding sequences and mass spectra
4. Post-translational modifications
5. Sample description, including species, tissue, subcellular location, disease state, and other relevant annotations
6. Instrumentation details and data processing settings
7. Processed peak lists in mzData format

PRIDE data can be public or private. Private data can be shared through a collaborative mechanism, allowing for controlled access. This mechanism enables manuscript reviewers to access corresponding PRIDE entries confidentially.

2.5.2 MassIVE (MASS spectrometry Interactive Virtual Environment)

MassIVE is a repository and data resource dedicated to reproducible quantitative mass spectrometry-based proteomics. Its extension, MassIVE.quant [25], accommodates all mass spectrometry data acquisition types and offers comprehensive computational analysis tools. A hierarchical structure enables systematic storage of raw data, experimental design metadata, analysis scripts, intermediate files, and alternative reanalyses.

To ensure reproducibility, MassIVE.quant stores intermediate output files at each step of the workflow, allowing users to inspect, reproduce, or modify any component. The workflow includes:

1. Experimental design annotations
2. Sample preparation and data acquisition strategies
3. Peptide ion identification
4. Quantification
5. Statistical analysis.

While MassIVE.quant supports diverse workflows, it employs standard formats like mzTab for analyte identification. Quantitative and statistical analysis results are presented in a tabular (.csv) format to highlight biologically relevant information.

The branching structure allows users to view and compare reanalyses performed with different tools and settings. MassIVE.quant also maintains datasets with four curation levels

(bronze, silver, gold, platinum) to ensure reproducibility and facilitate submission. By providing a platform for large-scale data deposition and comprehensive analysis, MassIVE.quant contributes significantly to advancing quantitative proteomics research.

2.6 Transformer models

The introduction of transformer models has revolutionized natural language processing (NLP) [155]. Unlike traditional recurrent neural networks (RNNs) and convolutional neural networks (CNNs), transformers can process input sequences in parallel, which significantly speeds up training and inference. The transformer architecture relies on *positional encoding* and *self-attention*. Positional encoding assigns unique numerical values to each word, preserving sequential information. Self-attention allows the model to assess the importance of every word in relation to others within a sentence. This mechanism enables the model to "learn" grammatical rules based on statistical patterns of language usage.

Transformer models operate by passing input data through multiple layers of self-attention and feedforward neural networks. This process involves:

1. **Input embedding**: Converting words into numerical representations. Positional encoding: Adding positional information to word embeddings.

2. **Multi-head attention**: Calculating attention weights across multiple attention heads to capture diverse relationships between words (Fig.2.6).

3. **Layer normalization and residual connections**: Stabilizing and accelerating training.

4. **Feedforward neural networks**: Applying non-linear transformations to capture complex patterns.

5. **Stacked layers**: Building hierarchical representations through multiple layers.

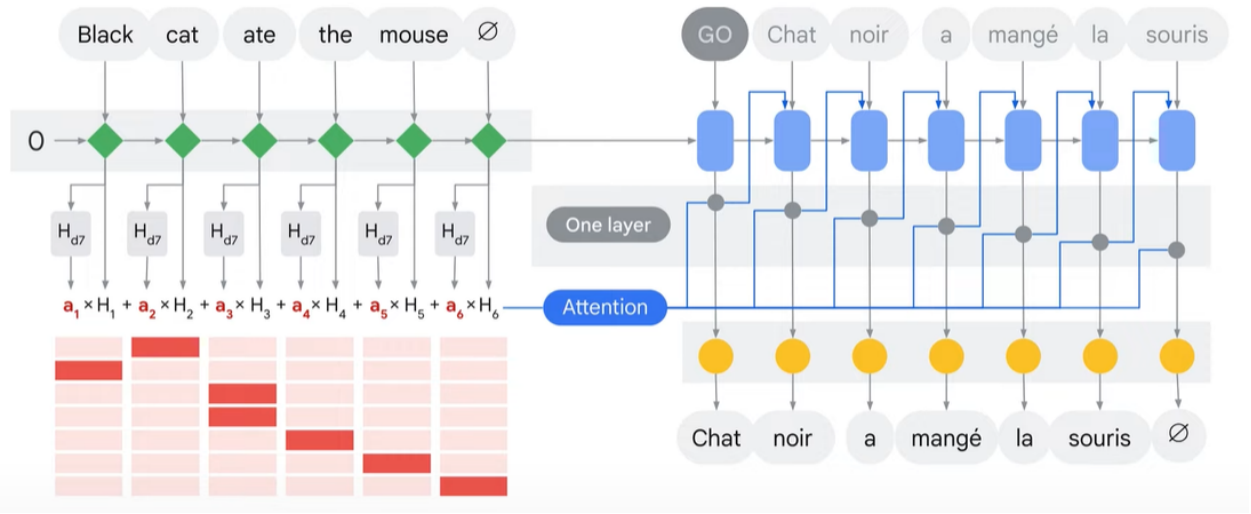


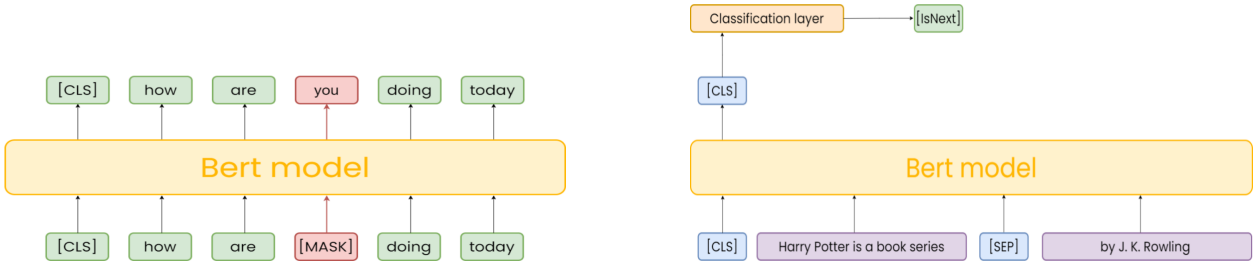
Figure 2.6: Machine translation task using an attention mechanism. Source: [Technology]

6. **Output layer**: Generating output sequences for tasks like machine translation.

Transformers are trained using supervised learning to minimize the difference between predicted and actual outputs. Once trained, they can be applied to various NLP tasks, including text generation, summarization, and question-answering. One of the most influential transformer-based models in NLP is BERT (Bidirectional Encoder Representations from Transformers), which was introduced in 2018. Pre-trained on extensive text data, BERT comprehends word context within a sentence. Unlike previous models, BERT processes text bi-directionally, capturing word meaning based on both preceding and succeeding words. This bidirectional approach significantly enhances the model's ability to grasp language nuances and achieve state-of-the-art accuracy in a wide range of NLP tasks.

2.6.1 Bidirectional Encoder Representations from Transformers (BERT)

BERT (Bidirectional Encoder Representations from Transformers) [38], was a revolutionary language model built on the transformer architecture. It has significantly outperformed previous state-of-the-art models and remains one of the most valuable transformer architectures [126], especially after the emergence of the *"very large language models"* like GPT [119] or Llama [153].



(a) Masked Language Modeling (b) Next sequence prediction

Figure 2.7: BERT training tasks

Initially designed for the English language, BERT came in two variations:

- $BERT_{BASE}$: 12 encoder layers, 12 attention heads, and 109 million parameters
- $BERT_{LARGE}$: 24 encoder layers, 16 attention heads, and 340 million parameters

Both models were pre-trained using the extensive *Toronto Book Corpus* [178] (800M words)) and English Wikipedia datasets (2,500M words).

In contrast to traditional encoder-decoder architectures or to the latest architectures [119], BERT is an *encoder-only* transformer. This means that its fundamental components include:

1. **Embedding**: This part converts text into numerical representations by assigning words or subwords to dense vectors. Before the embedding, BERT tokenizes the input using WordPiece tokenization, a subword-level technique that effectively handles out-of-vocabulary words.
2. **Encoder Stack**: This involves a series of transformer encoder blocks that process the input embeddings, capturing intricate word relationships through *bidirectional self-attention*
3. **Decoder**: This module converts the final representation vectors into one-hot encoded tokens by producing a predicted probability distribution over the token types

The model was pre-trained for two specific tasks:

- **Masked Language Modeling (MLM)**: (Fig. 2.7a) enforces bidirectional learning from text by masking (hiding) a word in a sentence and forcing BERT to bidirectionally use the words on either side of the covered word to predict the masked word. Through MLM, BERT can use context clues from the surrounding words, in addition to existing knowledge, to predict the hidden word. During training, around 15% of tokenized words were hidden, and BERT's task was to accurately predict these hidden words. The selected token was replaced with a `[MASK]` token with an 80% probability, replaced with a random word token with a 10% probability, and not replaced with a 10% probability. Following the processing of the input text, the model's fourth output vector was provided to its decoder layer, which in turn produced a probability distribution across its 30,000-dimensional vocabulary space.

- **Next Sentence Prediction (NSP)**: (Fig. 2.7b) involves predicting if two spans of text appeared sequentially in the training corpus. The model outputs either `[IsNext]` or `[NotNext]`. The first span begins with a special token `[CLS]` (*classify*), and the two spans are separated by a special token `[SEP]` (*separate*). After processing the two spans, the output vector (the vector coding for `[CLS]`) is passed to a separate neural network for binary classification into `[IsNext]` and `[NotNext]`. As a result of this training process, BERT learns latent representations of tokens and text in context.

Through these pre-training processes, BERT acquired comprehensive contextual representations of words and their relationships within sentences. These representations can be fine-tuned for specific NLP tasks with minimal data, leading to state-of-the-art performance in various applications such as question answering, text classification, and natural language inference.

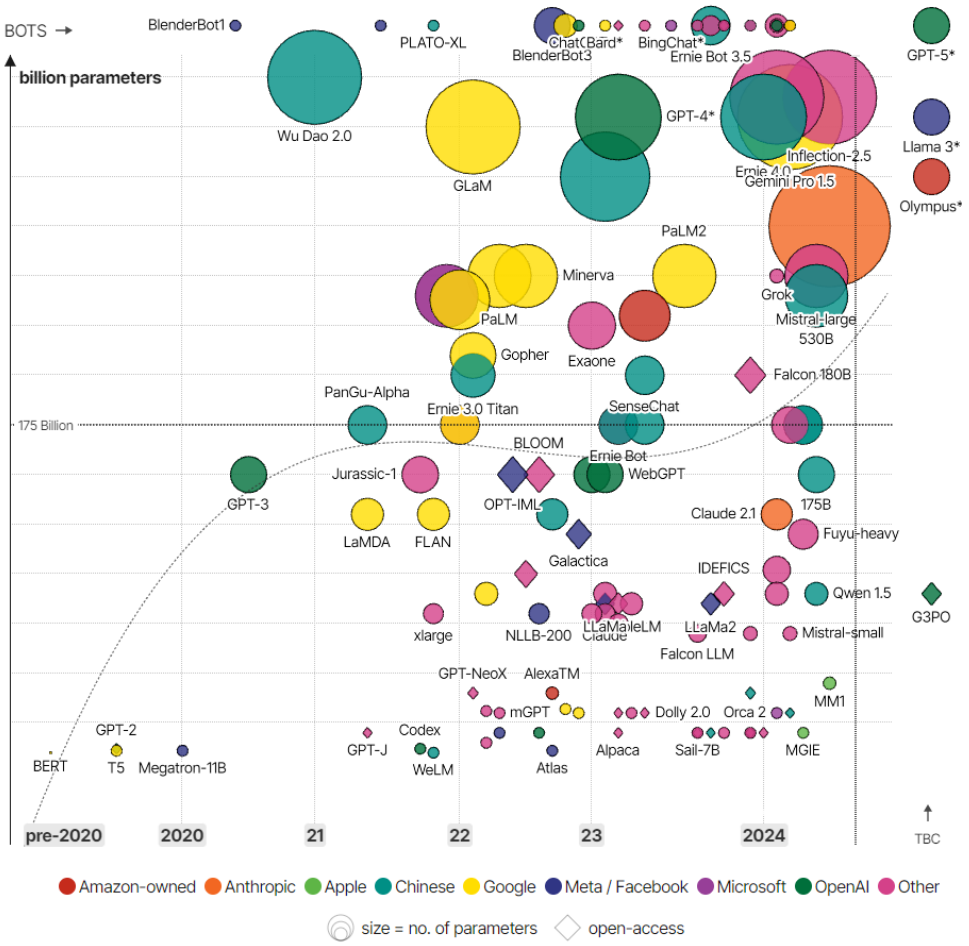


Figure 2.8: The Evolution of Language Models from 2018 to 2024. Source: [96]

2.7 Machine Learning Pruning algorithms

Large Language Models (LLMs) have emerged as the most effective and straightforward solution for achieving cutting-edge results in numerous natural language processing (NLP) applications. However, the growing utilization of neural networks (NNs) and transformer models [155] has increased computational expenses due to the intricate arithmetic computations, larger matrices, and the addition of more layers. Consequently, these models have become more elaborate in terms of weight and structure, demanding high levels of computation and memory resources.

The performance of LLM models is significantly influenced by both the volume of training

data and the number of parameters. In the NLP domain, a hypothesis akin to Moore's Law posits that model capabilities are growing exponentially [12] as shown in Fig. 2.8.

One of the best approaches to address the overwhelming size of LLMs is to reduce their resources through *pruning algorithms*. These algorithms can eliminate parameters or entire components in a NN, making it lighter without compromising its original performance. Pruning algorithms emerged in parallel with the earliest use of NNs [103, 66, 79], but they have gained significant importance in the last decade due to the widespread use of these networks in various fields. There are many pruning algorithms in literature [15], each with a unique approach or adapted old algorithms for these new architectures [11]. However, the complexity of neural networks can pose a challenge when creating pruning algorithms, as these may require new complex theorems to make the models lightweight [39, 87]. Additionally, existing pruning algorithms often exhibit shortcomings in their completeness [15] and fail to consider a critical aspect: the efficient storage of the pruned result. Some algorithms compress models at runtime but lack mechanisms to preserve the reduced NN for future use. Consequently, most algorithms prioritize the speed of reduction and execution, neglecting this critical final stage essential in resource-limited environments [168, 149].

2.7.1 Pruning algorithm categorization

Compression algorithms can be summarized in three areas of research: *weight pruning* [53, 177], *quantization* [49, 176] and *knowledge distillation* [7, 74]. These techniques aim to make models lighter, but each of them takes a different approach. Weight pruning removes model parameters according to the chosen algorithm and strategy, while quantization reduces the number of bits necessary to represent each parameter. Knowledge distillation, instead, tries to minimize the learned large knowledge of a model into a smaller one without affecting its *validation*.

Focusing on weight algorithms, there are different approaches depending on the strategy and algorithm adopted. Pruning algorithms can be classified as either *structured* or *un-*

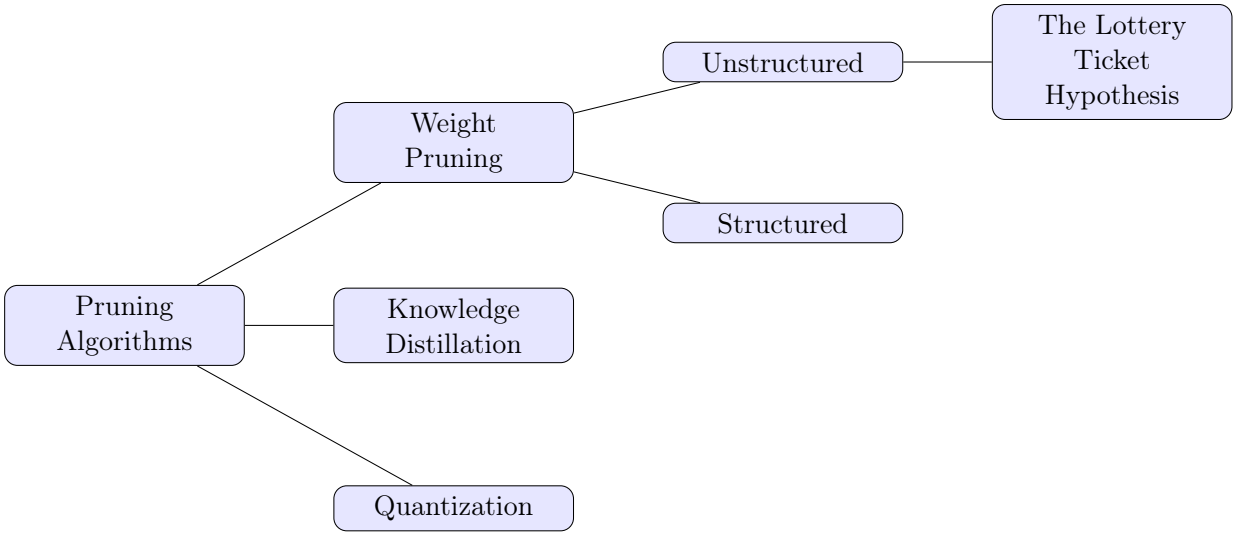


Figure 2.9: Hierarchy of pruning algorithms.

structured, based on the approach applied and *magnitude-based* or *impact-based*, according to the algorithm used. Structured pruning [64, 161, 50] removes weights in groups, such as entire neurons, filters or layers, while unstructured pruning [53, 45, 78, 11] does not consider any relationship between parameters and selects weights to prune based on their impact or magnitude. Magnitude-based algorithms [55, 103, 50] analyze the absolute value of each parameter to determine its importance. In contrast, impact-based algorithms [79, 56, 142] work on the loss function and its variation caused by removing a parameter.

2.7.2 The lottery ticket hypothesis

The *lottery ticket ticket* hypothesis [45], is a recent advancement in pruning techniques. A *winning ticket* is a subnetwork within a trained model that - when trained in isolation - can achieve performance comparable to the original model even after significant pruning.

The identification of a winning ticket involves the following steps:

Step 1: Initialization and Training of the Neural Network A pre-trained neural network $f(x;\theta_0)$ is given, where θ_0 are the initial model parameters. This model is trained for j iterations, yielding the parameter set θ_j.

Step 2: Generation of a Pruning Mask A pruning percentage $p\%$ is determined from the fine-tuned model $f(x;\theta_j)$. This percentage represents the proportion of parameters to be removed. A binary mask $m \in \{0,1\}^{|\theta|}$ is then created, which decides for each parameter whether it will remain in its fine-tuned state or reset to its pre-trained value.

Step 3: Creation of the Winning Ticket Using the mask m, the resulting $f(x; m \odot \theta_0)$ model is the *winning ticket*.

In essence, the winning ticket is identified from a fine-tuned model by applying a pruning criterion to generate a binary mask. This mask is used to select which parameters will stay in their fine-tuned state and which will be reset to their pre-training state. The model that results, with $p\%$ of its parameters reset, is the *winning ticket model*. This process can be repeated multiple times or in a one-shot manner.

Chapter 3

Methods

To explore neoantigen presentation through large language models (LLMs), we need two key components: a robust dataset and a suitable model.

This chapter begins with an in-depth look at the CARMEN dataset (Sec. 3.1), which is crucial for training and evaluating our neoantigen presentation prediction models. In Sec. 3.1.1, we closely examine the data collection process, detailing each stage leading up to the peptide mapping. We wrap up this section with a summary of all datasets within CARMEN, presented in Tab. 3.1. Next, Sec. 3.1.2 provides a thorough overview of the CARMEN dataset, highlighting the number of unique peptides, the distribution of HLA alleles, and the variety of cancer types included. For deeper insights into specific elements, Apx. A offers further analysis.

Shifting our attention to model architecture, Sec. 3.2 introduces TAPE (Task Assessing Protein Embedding), a transformer model crafted for protein evaluation. This section delves into the model composition, parameter count, pre-training steps, and performance benchmarks.

In recent years, LLMs have grown both in size and complexity (Sec. 2.7). To tackle this, we have designed a novel pruning algorithm named KEN [94]. KEN is efficient and universal, capable of compressing LLMs by leveraging their parameter distribution through

Kernel Density Estimators (KDEs). In Sec. 3.3, we discuss how KEN is implemented to optimize LLMs. A detailed breakdown of the KEN algorithm is provided in Sec. 3.3.1, and in Sec. 3.3.2, we evaluate its effectiveness on sentiment analysis datasets, with results showcased in Sec. 3.3.3.

By integrating the strengths of CARMEN with advances in Transformer models and the efficiency of KEN, we aim to improve the development of more effective cancer immunotherapies, tools for protein sequence analysis, and simplified neural network models.

3.1 CARMEN: Cancer ImmunoPeptidogenomics Corpus

The CARMEN (Cancer ImmunoPeptidogenomics) database [70] is a collection of data obtained from a thorough re-analysis of 75 publicly available mass spectrometry datasets spanning from 2015 to 2023. It focuses on identifying mutations in immune-visible regions based on presentation frequencies, population demographics, and mutational loads.

CARMEN introduces the concept of *"epitope contigs"* and *"scaffolds"* to describe the genomic and proteomic context of peptide presentation. Through this re-analysis, researchers have identified over 810,000 unique canonical and non-canonical peptides, expanding our understanding of cryptic tumor-associated antigens. By linking peptides to their genomic origins, CARMEN facilitates the prioritization of mutations in immune-visible regions.

Unlike existing databases like caAtlas, SysteMHC, SYFPEITHI, Tantigen, IEDB, and CEDAR, CARMEN prioritizes non-canonical immunopeptidome derived from non-standard open-reading frames. This focus, along with the inclusion of 8,500 non-canonical peptides, underscores CARMEN's unique contribution to the field of cancer immunotherapy research.

3.1.1 Corpus Development

For the corpus composition, CARMEN was defined using the four steps described below:

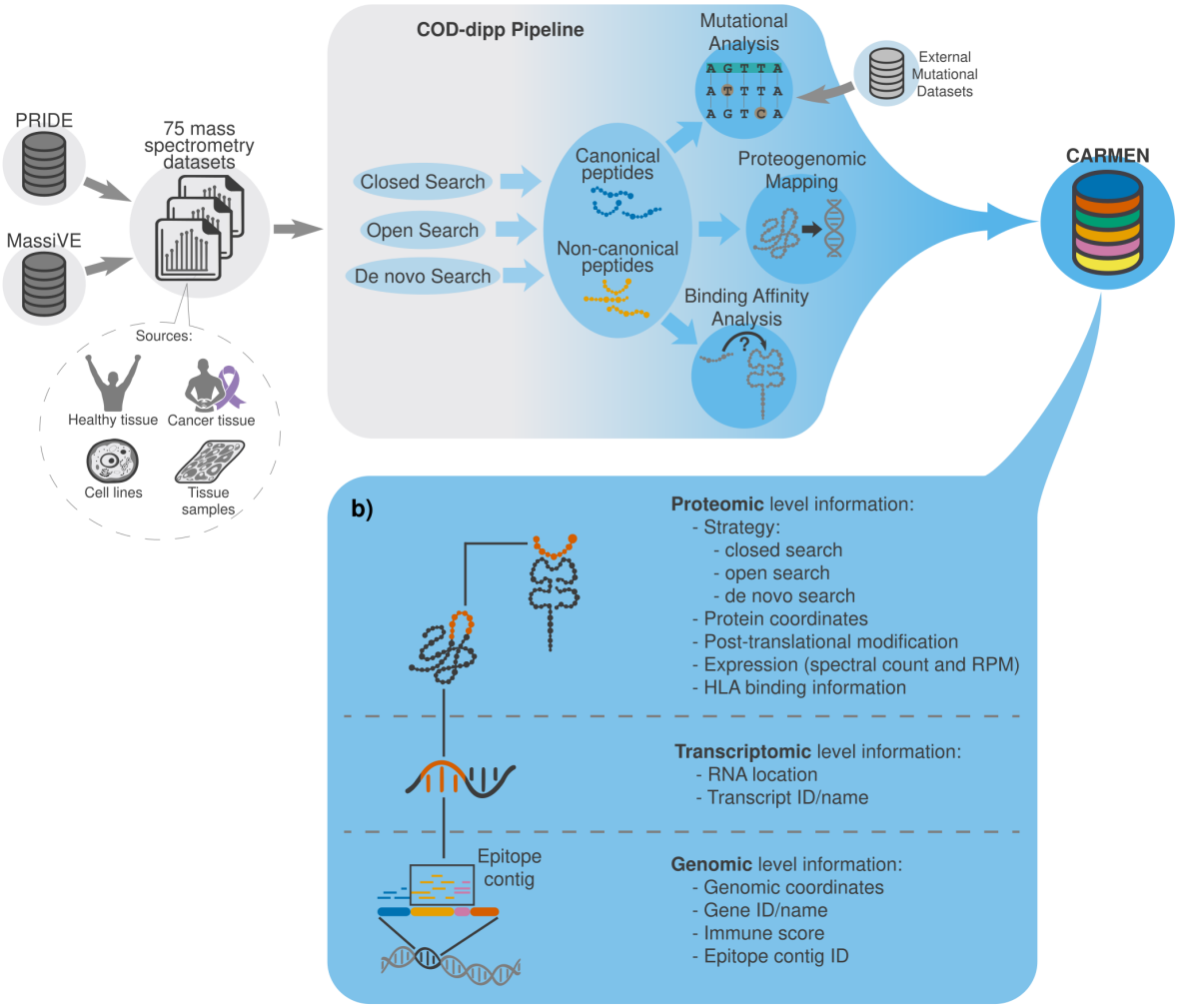


Figure 3.1: CARMEN design steps

Step 1: Data collection and processing Seventy-five immunopeptidomic mass spectrometry datasets were collected from PRIDE (Sec. 2.5.1) and MASSIVE (Sec. 2.5.2), representing 2,300 samples and 34 different cancers. The datasets were selected based on specific keywords related to sample types (*cancer* and *normal*), mass spectrometry technologies (*LTQ Orbitrap*), experiment types (*immunopeptidomics*), and data acquisition methods (*Data-dependent acquisition*). Raw data files were converted to the mzML format using the MSConvert tool from the ProteoWizard suite.

Dataset name	# records	Description
`main`	7,896,364	The main database table includes primary data for each peptide identification, such as sample and experiment details, sequence, type, and binding HLA allele.
`annotation-msfragger`	3,750,647	The proteomic data includes peptide sequence, gene/transcript/protein of origin, and protein start and endpoints.
`annotation-pogo`	3,129,375	The genomic data associated with each peptide, including the peptide sequence and genomic coordinates obtained from the PoGo algorithm.
`hla-sequences`	10,614	A table with all the allele HLA sequences and pseudo-sequences database

Table 3.1: CARMEN corpus composition

Step 2: Mass spectrometry searching: first and second-round searches Authors employed the *"COD-Dipp"* pipeline without modification for MS mass spectrometry searches [10]. This pipeline integrates MSGF+ for *"Closed"* searches, MS-Fragger for *"Open"* searches, and DeepNovo V2 for *"de novo"* searches. mzML files underwent two rounds of false-discovery rate-controlled proteomic database searching against the `ENSEMBL Human Proteome hg38.p13`. The first round identified *modified*, *unmodified* and *de novo* immune peptides, while the second round validated their presence.

Step 3: Binding affinity analysis of peptides The binding affinity of the peptides obtained from the previous steps was annotated using NetMHCpan-4.1 [122] against the haplotypes for each sample. Binders were annotated as strong, weak, and non-binders. (*strong binders* ≤ 0.2 and *weak binders* ≥ 1.2).

Step 4: Proteogenomic mapping of peptides Pogo was used to map identified peptides onto the genome, and genomic coordinates were annotated for each peptide. The mapped peptides and their chromosomal coordinates were output into a comprehensive tab-separated BED file for interoperability to genomic pipelines.

All of these steps are shown graphically in Fig.3.1, composing the final corpus of 4 different databases, described in Tab. 3.1.

3.1.2 Deep analysis and composition

The CARMEN database offers a comprehensive overview of antigen presentation across various cancers, aiding in the identification of highly visible regions in the human genome for immunotherapy. It covers 31 cancer types and 30 tissue types, with a total of 2,323 samples, including 1,390 (60%) cancerous, 849 (37%) healthy, 51 (2%) benign, and 33 (1%) unclassified samples.

The majority of the samples consist of patient-derived tissues (1,169, 50%), followed by cell lines (910, 39%), and primary cells (244, 11%). Tissue samples are preferred for studying the immunopeptidome due to their cellular heterogeneity and preservation of the tumor microenvironment. Cell lines such as A375, Jurkat, and OVCAR are included in the database.

CARMEN `main` dataset contains 816,222 unique peptides, including 797,561 canonical, 8,743 non-canonical, and 9,918 unclassified peptides. These peptides cover approximately 3% of the known human proteome and 97% of all possible 9-mers from the human proteome. The length distribution of peptides is skewed towards 8-10 amino acids (569,928 or 69.82%), with 9 amino acids being the most prevalent (338,658 or 41.4%). Only a small fraction (154 or 0.01%) of peptides are 20 amino acids or longer.

Cancer samples show a high level of uniqueness, with 332,316 (40.71%) distinct peptides that could potentially be used as specific tumor antigens. Additionally, there are at least 134,564 shared peptides that could be considered as potential tumor-associated antigens. The corpus analysis also assessed the universality of peptides for developing *"off-the-shelf"* vaccines and found that 64% of the peptides were shared across samples, indicating a high degree of shared immunopeptide motifs. This suggests that certain genomic regions, known as *"hotspot"* proteomic regions, are more immunologically visible than others. These regions, well-characterized at the protein level [105, 107, 69], frequently present peptides due to their common recognition by HLA molecules. It is important to note that proteomic *"hotspots"* dif-

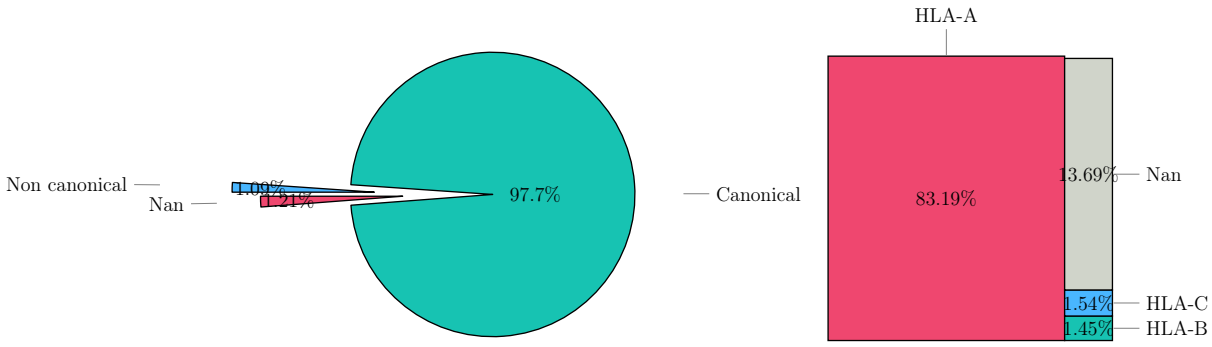


Figure 3.2: Canonical peptides distribution (left) and HLA-I distribution (right)

fer from *"mutation hotspot"*, which are genomic regions with high tumor mutational burdens [106].

Regarding HLAs, the database covers a total of 202 HLA-I alleles. The authors focused on HLA-I peptides due to their interest in CD8+ T-cell responses, which are associated with the shorter length of HLA-I peptides. Peptides longer than 15 amino acids may indicate additional HLA-II binders. From the 816,222 unique peptides, 14,190 (1.7%) were common to healthy, cancerous, and benign tissues, while 264,919 (32.45%) were shared between healthy and cancer tissues.

HLA binding affinity was assessed using NetMHCpan4.1, and specific HLA alleles binding to each peptide were identified. The primary types of alleles identified were HLA-A (56 unique alleles), HLA-B (90 unique alleles), and HLA-C (56 unique alleles). Figure 3.2 shows the canonical peptide distribution and the HLA-I distribution of the CARMEN `main` dataset. Additional results are presented in Apx. A, highlighting the core dataset composition from different perspectives, including peptide length (Apx. A.0.1), spectral counts distribution (Apx. A.0.2), and a detailed analysis of the HLA-A, HLA-B, and HLA-C subclasses (Apx. A.0.3).

3.2 Tasks Assessing Protein Embeddings (TAPE)

Tasks Assessing Protein Embeddings (TAPE) [120] is a benchmark comprising five biologically relevant semi-supervised learning tasks spanning diverse domains of protein biology. The authors meticulously curated training, validation, and test splits for each task to assess biologically meaningful generalization applicable to real-world scenarios.

Pfam [101], a comprehensive database of over thirty-one million protein domains widely used in bioinformatics, served as the corpus for TAPE. Pfam sequences are organized into evolutionarily related clusters known as *families* (random or heldout). Leveraging this structure, the authors constructed a test set consisting entirely of unseen families (approximately 1% of the data). The remaining data was randomly split into 95% for training and 5% for validation. Perplexity on the uniformly random split test set evaluates in-distribution generalization to proteins with lower evolutionary relatedness to the training set.

To assess the performance of different model architectures on TAPE, the authors trained five neural networks: an LSTM [60], a Transformer [155] (Sec. 2.6), a dilated residual networks (ResNet) [170], a bidirectional LSTM (B-LSTM) [13], and a unidirectional mLSTM [4]. The details of each architecture are described below:

- **LSTM**: 1024 hidden units corresponding to the forward and backward language models, whose outputs were concatenated in the final layer.
- **Transformer**: 12 layers, a hidden size of 512 units, and 8 attention heads
- **ResNet**: 35 residual blocks, each containing two convolutional layers with 256 filters, kernel size 9, and dilation rate 2
- **Bi-LSTM**: composed of three 512 hidden unit bidirectional LSTMs
- **mLSTM**: with 1900 hidden units

Model	Random Families			Heldout Families		
	Accuracy	Perplexity	ECE	Accuracy	Perplexity	ECE
Transformer	**0.45**	**8.89**	**6.01**	**0.30**	**13.04**	**10.04**
LSTM	0.40	8.89	6.94	0.16	14.72	15.21
ResNet	0.41	10.16	6.86	0.29	13.55	10.32
Bi-LSTM	0.28	11.62	10.17	0.14	15.28	16.02
mLSTM	0.32	11.29	9.08	0.12	16.36	16.92

Table 3.2: TAPE pre-training paper results

Hyperparameters for the LSTM, ResNet, B-LSTM, and mLSTM were chosen to match the Transformer's parameter count. The Transformer and ResNet were trained using masked-token prediction, while the LSTM, B-LSTM, and mLSTM were trained with next-token prediction.

Table 3.2 presents a summary of the training performance of the five architectures on random and held-out families. The metrics used for evaluation are accuracy, perplexity, and exponentiated cross entropy (ECE). As expected, the Transformer consistently outperforms the other models on all tasks, demonstrating its effectiveness in learning meaningful protein representations. This superior performance emphasizes the advantages of the Transformer architecture, including its ability to capture long-range dependencies and utilize self-attention mechanisms.

After pre-training the five models, the authors developed the 5 tasks as follows:

1. **Task 1: Secondary Structure Prediction**

   Predicts the secondary structure (helix, strand, or other) of each amino acid in a protein sequence.

   - Dataset: Klausen et al. [76]
   - Metric: Per-amino acid accuracy [33]

2. **Task 2: Contact Prediction**

   Determines whether two amino acids within a protein sequence are in spatial proximity.

   - Dataset: ProteinNet [5]

Model	Tasks				
	#1	#2	#3	#4	#5
Transformer	0.73	0.36	0.21	**0.68**	**0.73**
LSTM	**0.75**	0.39	**0.26**	0.67	0.69
ResNet	**0.75**	0.29	0.17	0.21	**0.73**
Bi-LSTM	0.73	**0.40**	0.17	0.33	0.64
mLSTM	0.73	0.34	0.23	0.67	**0.73**

Table 3.3: Model performances on the five proposed tasks

- Metric: Precision of the L/5 most likely contacts for medium- and long-range interactions on the ProteinNet CASP12 test set [102]

3. **Task 3: Remote Homology Detection**

   Assigns each protein to one of 1195 potential protein folds.

   - Dataset: Hou et al. [62]
   - Metric: Overall classification accuracy on the fold-level heldout set from Hou et al. [62]

4. **Task 4: Fluorescence Landscape Prediction**

   Maps each protein sequence to its corresponding log-fluorescence intensity.

   - Dataset: Sarkisyan et al. [135]
   - Metric: Spearman's ρ (rank correlation coefficient)

5. **Task 5: Stability Landscape Prediction**

   Maps each protein sequence to a value representing its stability under extreme conditions.

   - Dataset: Rocklin et al. [125]
   - Metric: Spearman's ρ (rank correlation coefficient)

As shown in Table 3.3, the models exhibit significant performance variations across the five tasks. Notably, the Transformer model demonstrates a substantial performance ad-

vantage in tasks 4, "*Fluorescence Landscape Prediction*", and 5, "*Stability Landscape Prediction*", both of which involve mapping protein landscapes. This superior performance likely stems from the Transformer's ability to leverage a high-dimensional embedding space to effectively capture complex data representations. Conversely, the LSTM model achieves the highest performance in task 1, "*Secondary Structure Prediction*", and task 3, "*Remote Homology Detection*". While its performance in task 3 marginally surpasses that of the Transformer and mLSTM, the Bi-LSTM variant surprisingly shows a considerable performance drop compared to the standard LSTM (0.17 vs. 0.26). This discrepancy is not observed in task 1, where all models achieve relatively similar performance, ranging from 0.73 to 0.75. This consistency suggests that the inherent architectural strengths of all models enable accurate prediction of amino acid structures in this particular task. The ResNet model, despite its architecture featuring extensive convolutional filters (256), does not consistently outperform the other models. Although it achieves competitive results in tasks 1 and 5, sharing leadership with LSTM in task 1 and Transformer/mLSTM in task 5, its performance is generally comparable. This could be attributed to its distinct convolutional-based architecture, differing significantly from the LSTM and attention-based (Transformer) models. While the mLSTM maintains performance parity with the standard LSTM, the Bi-LSTM exhibits a divergent pattern. It excels in task 2, "*Contact Prediction*", but otherwise performs comparably to the other models, with performance scores differing by less than 0.01.

The results presented in Table 3.3 collectively demonstrate that the Transformer and LSTM models consistently achieve superior or equivalent performance compared to ResNet, Bi-LSTM, and mLSTM across the five tasks. This advantage is attributable to their architectures, which are specifically optimized for sequential data processing. Although ResNet, Bi-LSTM, and mLSTM show promise in certain contexts, their limitations in handling sequential information become apparent in others.

Despite not achieving the absolute best performance in every task, the Transformer's over-

all superiority and adaptability make it the optimal choice for our research. Its pre-trained scores, as detailed in Table 3.2, highlight its ability to capture long-range dependencies and utilize self-attention mechanisms, both of which are crucial for protein sequence analysis and structure prediction. Given the Transformer's strong foundation and its potential for further enhancement through our proposed pruning algorithm (KEN), we have selected it as the primary model for the remainder of this manuscript. By fine-tuning the pre-trained Transformer model, we aim to achieve even more remarkable results and contribute significantly to the field of protein sequence analysis.

Consequently, throughout the rest of this manuscript, we refer to the selected Transformer model as *"TAPE"* in accordance with the reasons described above and for its reference paper.

3.2.1 Other BERT-base Transformer Models for Neoantigens detection

Here, we provide an overview of various pre-trained transformer models on biological datasets, offering a comprehensive view of the current state-of-the-art (SOTA) models in this field.

TAPE is a well-known transformer model pre-trained for biological tasks, including neoantigen presentation. However, several other transformer architectures based on BERT have been developed and trained specifically for this purpose. Notable examples include ProtBert-BFD, ProtT5 [41], ESM-1 [123], and ESM-2 [82], each with distinct strengths and benchmark evaluations.

ProtBert-BFD, part of the ProtTrans family, is trained exclusively on the extensive BFD dataset. This family includes larger variants such as ProtT5-XL and ProtT5-XXL, which range from 420 million to 11 billion parameters. Initially trained on BFD, these models were later fine-tuned on UniRef50 for enhanced performance. Interestingly, despite its smaller size, ProtT5-XL outperformed ProtT5-XXL, likely due to the necessity of even larger datasets to fully leverage the capabilities of massive models.

Model	Pre-trained Dataset	Dataset Sample	Layers	Hidden size	Attention head	Total Parameters
TAPE	Pfam	30M	12	768	12	92M
ProtBert-BFD	BFD	2122M	30	1,024	16	420M
ProtT5-XL	Uniref50/BFD	2122M	24	1,024	32	3B
ProtT5-XXL	Uniref50/BFD	2122M	24	1,024	128	11B
ESM-1 (6 layers)	Uniref50	60M	6	768	12	43M
ESM-1 (12 layers)	Uniref50	60M	12	768	12	85M
ESM-1 (34 layers)	Uniref50	60M	34	1,280	20	670M
ESM-1b	Uniref50	60M	34	1,280	20	650M
ESM-2 (6 layers)	Uniref50	60M	6	320	20	8M
ESM-2 (12 layers)	Uniref50	60M	12	480	20	35M
ESM-2 (30 layers)	Uniref50	60M	30	640	20	150M
ESM-2 (33 layers)	Uniref50	60M	33	1,280	20	650M
ESM-2 (36 layers)	Uniref50	60M	36	2,560	20	3B
ESM-2 (48 layers)	Uniref50	60M	48	5,120	20	15B

Table 3.4: BERT-based model pre-trained for Neoantigens presentation task: Credit [86]

ESM-2, a member of the Evolutionary Scale Modeling (ESM) family, improves upon its predecessor, ESM-1b, by eliminating dropout and incorporating Rotary Position Embedding (RoPE). While RoPE slightly increases training costs, it significantly enhances model quality, particularly for smaller architectures. The authors trained ESM-2 on the non-redundant UniRef50 dataset from UniProt, a curated collection of protein sequences.

Tab. 3.4 provides a summary of the BERT-based models utilized for neoantigen detection, highlighting their structure and the datasets used for pre-training.

From the information in Tab. 3.4, it is clear that TAPE stands out as one of the lighter models, having been trained on a notably smaller dataset in comparison to its counterparts. Its architecture, which includes the number of layers, attention heads, and hidden size, closely resembles that of the original BERT model. While other models like ESM are also popular in this field and deliver excellent performance, we chose to go with TAPE due to its BERT-like design.

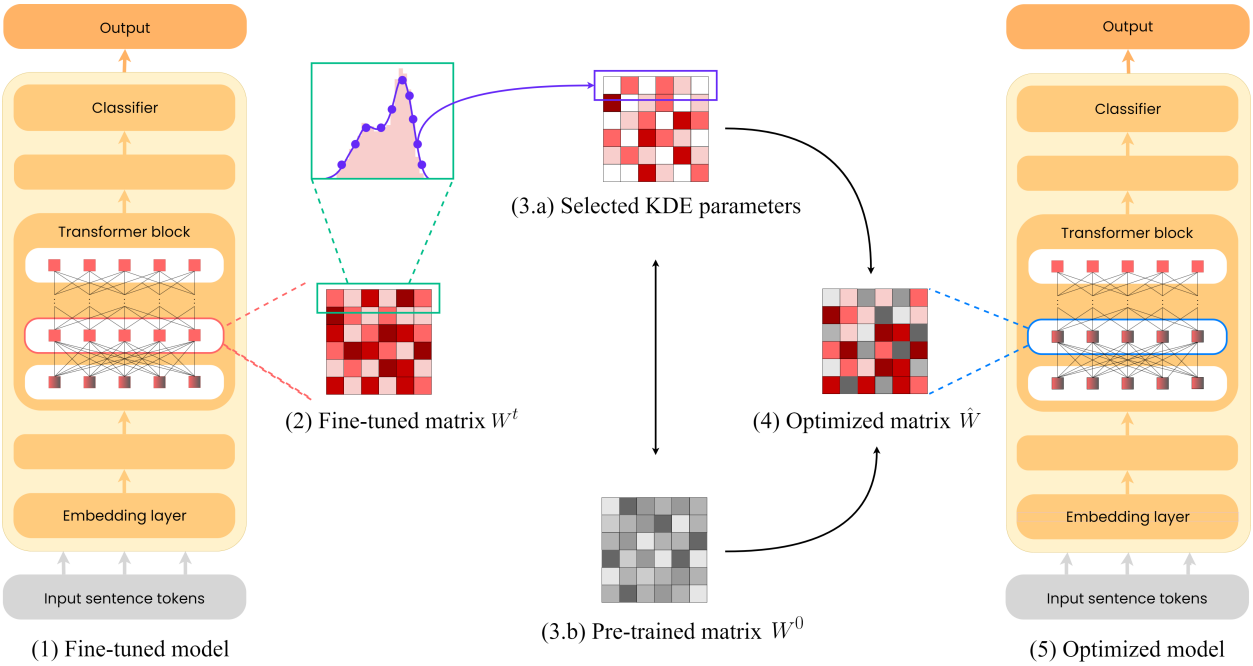


Figure 3.3: KEN workpath: From a fine-tuned model (1), for each of its fine-tuned matrices (2), the row distribution and the respective KDE (Kernel Density Estimator) are calculated. All values within the KDE are selected (3.a), while the remainder are restored to their pre-tuned value (3.b). The resulting optimized matrix (4) is then fed back into the model (5)

3.3 Kernel density Estimator for Neural Network compression (KEN)

This section introduces KEN (Kernel Density Estimator for Neural Network Compression), a novel pruning algorithm developed by Mastromattei [94]. To demonstrate its effectiveness, KEN was initially evaluated on sentiment analysis datasets and seven different Large Language Models, extending beyond the primary focus of this research. This approach was motivated by the widespread use of LLMs and sentiment analysis datasets in Natural Language Processing applications, providing a comprehensive assessment of KEN's capabilities.

The following sections delve into the details of our experiments. Sec. 3.3.1 formally presents the KEN algorithm, including its parameter selection process and matrix replacement strategy. Sec. 3.3.2 outlines the experimental setup (Sec. 3.3.2.1) and model compression techniques employed (Sec. 3.3.2.2). Finally, Sec. 3.3.3 presents the experimental

Algorithm 1 Generate the optimized $\hat{W}$ matrix using KEN

Data: $W^0 = \{w^0_{1,1}, ..., w^0_{n,m}\}, \quad W^t = \{w^t_{1,1}, ..., w^t_{n,m}\}, \quad k$
Result: $\hat{W}$

$\hat{W}[n, m] \leftarrow 0$
for i = 1 to n **do**
    `best_points` $\leftarrow KDE(r^t_i, k)$
    for j = 1 to m **do**
        $\hat{r}^t_i \leftarrow []$
        if $r^t_i[j]$ in `best_points` **then**
            $\hat{r}^t_i[j] \leftarrow r^t_i[j]$
        else
            $\hat{r}^t_i[j] \leftarrow r^0_i[j]$
        end if
    end for
    $\hat{W}[i] \leftarrow \hat{r}^t_i$
end for
return $\hat{W}$

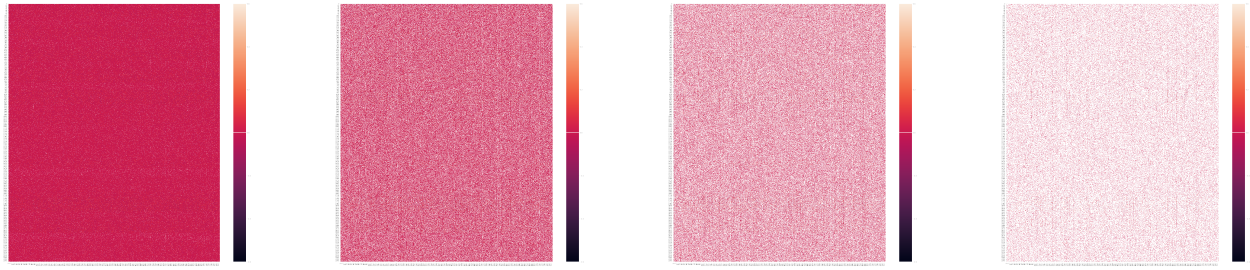
results, with a particular emphasis on KEN_{viz} (Sec. 3.3.3.3): its visualization tool.

Furthermore, Apx. C, show another application of the KEN algorithm in language variant overlap on irony detection.

3.3.1 KEN pruning algorithm

KEN (**K**ernel density **E**stimator for **N**eural network compression) pruning algorithm is designed to identify and extract the most essential subnetwork from transformer models following the main idea of the *winning ticket* hypothesis [45]. Our algorithm effectively prunes the network by employing Kernel Density Estimators (KDEs), retaining only the essential parameters and resetting the rest to their pre-trained values. The optimized subnetwork can be stored independently and seamlessly integrated into its pre-trained configuration for downstream applications.

KEN through KDEs, generalizes the point distribution of each transformer matrix, capturing the *smoothed* version of the original fine-tuned model. To prevent the complete deconstruction of the initial matrix composition, KEN applies KDEs to individual rows.



(a) Fine-tuned matrix

(b) Reset params 34.90%

(c) Reset params: 60.94%

(d) Reset params: 86.98%

Figure 3.4: Comparing the impact of KEN parameter selection on the same fine-tuned matrix (a). Matrix (a) represents the `in_proj` matrix at layer 0 of a DeBERTa model trained on the `AG_NEWS` dataset. No selected parameters are blank

The KDE calculation requires a k value, which defines the number of points employed in the distribution calculation which directly influences the number of retained fine-tuned parameters. Thus, a lower k value indicates a closer resemblance to the pre-trained model while a higher k value reflects a closer alignment with its fine-tuned version.

KEN algorithm operates in three primary steps:

Step 1: Parameter Extraction and KDE Calculation Given a pre-trained matrix W^0 of a layer l:

$$W^0 = \{w^0_{1,1}, ..., w^0_{n,m}\} \quad | \quad W^0 \in \mathbb{R}^{n \times m}$$

and its corresponding fine-tuned counterpart W^t:

$$W^t = \{w^t_{1,1}, ..., w^t_{n,m}\} \quad | \quad W^t \in \mathbb{R}^{n \times m}$$

for each row r^t_i of the fine-tuned matrix W^t:

$$r^t_i = \{w^t_{i,1}, ..., w^t_{i,m}\} \quad \forall i \in [1, n]$$

KEN calculates the KDE distribution of the row r_i^t using a bandwidth parameter h determined following Scott's rule of thumb [139].

$$h = 1.06 \cdot \hat{\sigma} \cdot n^{-\frac{1}{5}}$$

where $\hat{\sigma}$ is the standard deviation of r_i^t.

Step 2: Parameter Retention and Pre-trained Value Reset The k points that best fit the r_i^t row distribution are identified using the KDE likelihood, while the others are reset to their pre-trained values. This process results in an optimized row $\hat{r}_i$:

$$\hat{r}_i = \{\hat{w}_{i,1}, ..., \hat{w}_{i,m}\} \quad \forall i \in [1, n]$$

computed using the following binary function:

$$f(\hat{w}_{i,j}) = \begin{cases} w_{i,j}^t & \text{if} \quad w_{i,j}^t \quad \in \quad \text{KDE} \quad \text{likehood} \\ w_{i,j}^0 & \text{otherwise} \end{cases}$$

Step 3: Matrix Replacement and Optimized Fine-tuned Model After applying the previous step on each row, the optimized matrix $\hat{W}$:

$$\hat{W} = \{\hat{w}_{1,1}, ..., \hat{w}_{n,m}\} \quad | \quad \hat{W} \in \mathbb{R}^{n \times m}$$

will replace the original fine-tuned matrix W^t within the model.

KEN operates iteratively, replacing the W^t matrix with $\hat{W}$ during each iteration. Therefore, after the $t-th$ iteration, the model will have $t-optimized$ matrices, effectively replacing the fine-tuned matrices without creating any additional model versions. This versatility allows KEN to prune the entire model or specific layer ranges.

Algorithm 1 provides a more formal explanation of the three steps described for generating

Model		# Layers	# params
BLOOM_{1B7}	Workshop et al. [164]	24	1.72 B
BLOOM_{560k}	Workshop et al. [164]	24	560 M
DeBERTa	He et al. [58]	12	138 M
Bert	Devlin et al. [38]	12	109 M
Ernie	Sun et al. [148]	12	109 M
DistilBERT	Sanh et al. [133]	6	66 M
Electra	Clark et al. [28]	12	33 M

Table 3.5: Properties of the analyzed models

the optimized matrix $\hat{W}$. Additionally, the graphical representation in Fig. 3.3 offers a clear and comprehensive visualization of all KEN steps, while Fig. 3.4 displays different $\hat{W}$ matrices obtained using various k values.

3.3.2 Experiments on sentiment analysis datasets

To validate our algorithm, we conducted a series of extensive case studies. Sec. 3.3.2.1 describes the experimental setup, including the models employed and the k values tested. Additionally, Sec. 3.3.2.2 focused on investigating the feasibility of saving and loading compressed data.

3.3.2.1 Experimental set-up

We conducted a thorough series of experiments using seven distinct transformer models to evaluate KEN pruning algorithm performance across different architectures and datasets. We uniformly divided each dataset into training, validation, and test sets to maintain consistent evaluation conditions. These divisions remained consistent throughout our experiments and across models. All datasets were imported from Huggingface[1]. To achieve optimal performance, we fine-tuned each model before applying KEN algorithm per each dataset, adjusting the number of epochs until the fine-tuned model achieved the best F1-weighted score. Despite what the literature suggests, we used the F1 measure instead of classical accuracy as a

[1] https://huggingface.co/datasets

Model	Trainable params	Reset params (%)	AG-NEWS	EMO	IMDB	YELP_POLARITY	glue-sst2
$BLOOM_{1B7}$	442M	74.31	87.5 (±0.1)	**88.0** (±0.1)	76.6 (±0.1)	**96.1** (±0.1)	**80.4** (±0.1)
	531M	69.17	92.2 (±0.1)	**90.6** (±0.1)	84.2 (±0.1)	**96.3** (±0.1)	**90.9** (±0.1)
	664M	61.46	**93.1** (±0.1)	**90.1** (±0.1)	**87.6** (±0.1)	**96.5** (±0.1)	**92.9** (±0.1)
$BLOOM_{560k}$	411M	26.34	91.3 (±0.1)	**81.4** (±0.1)	**82.7** (±0.1)	**95.2** (±0.1)	**92.4** (±0.1)
	420M	24.80	91.8 (±0.1)	**83.0** (±0.1)	**84.3** (±0.1)	**95.3** (±0.1)	**92.1** (±0.1)
	429M	23.26	**92.1** (±0.1)	**84.0** (±0.1)	**85.8** (±0.1)	**95.3** (±0.1)	**92.3** (±0.1)
DeBERTa	92M	33.86	92.2 (±0.1)	**87.9** (±1.2)	82.5 (±5.1)	95.9 (±0.4)	94.6 (±0.2)
	99M	28.35	**92.7** (±0.1)	**87.3** (±1.0)	88.3 (±1.1)	**96.1** (±0.2)	**94.9** (±0.1)
	107M	22.84	**92.9** (±0.1)	**87.1** (±1.2)	**89.8** (±0.1)	**96.2** (±0.1)	**94.8** (±0.1)
Bert	69M	37.05	**93.4** (±0.1)	84.2 (±1.1)	86.8 (±0.1)	**95.0** (±0.4)	**93.7** (±0.5)
	75M	31.80	**93.7** (±0.2)	**87.4** (±0.7)	**87.3** (±0.1)	**95.0** (±0.5)	**93.7** (±0.4)
	80M	26.55	**93.6** (±0.1)	**87.9** (±0.3)	**87.6** (±0.1)	**95.1** (±0.4)	**93.8** (±0.4)
Ernie	69M	37.05	**93.3** (±0.4)	**89.1** (±0.6)	**89.4** (±0.2)	**95.8** (±0.1)	**94.1** (±0.2)
	75M	31.80	**93.3** (±0.3)	**88.7** (±1.2)	**89.2** (±0.2)	**95.8** (±0.3)	**93.8** (±0.2)
	80M	26.55	**93.8** (±0.2)	**88.1** (±0.8)	**89.6** (±0.3)	**95.9** (±0.2)	**93.4** (±0.2)
DistilBERT	44M	34.39	92.3 (±0.6)	**88.1** (±1.4)	83.2 (±1.1)	94.6 (±0.1)	**91.9** (±0.2)
	47M	28.92	**93.1** (±0.2)	**88.8** (±0.6)	84.4 (±0.5)	94.7 (±0.1)	**91.9** (±0.1)
	51M	23.45	**93.3** (±0.2)	**88.2** (±0.3)	**84.6** (±0.9)	**94.9** (±0.1)	**92.0** (±0.1)
Electra	8.9M	75.56	84.1 (±2.4)	84.3 (±0.4)	78.9 (±0.5)	88.5 (±0.9)	79.9 (±0.7)
	12M	64.75	89.7 (±0.3)	**86.0** (±0.3)	82.0 (±0.5)	92.1 (±0.8)	85.0 (±0.2)
	14M	55.94	**91.3** (±0.2)	**85.6** (±0.3)	84.3 (±0.1)	**93.7** (±0.4)	90.1 (±0.1)

Table 3.6: Results on various datasets obtained using different trainable parameters. Bold results indicate a similar or better F1-weighted value compared to the original (*unpruned*) model. The reset params column indicates the percentage of the restored pre-trained params in the model. Other results are shown in Apx.B.2

comparison metric - if not explicitly used by the comparison benchmarks - because it delivers more reliable predictions, particularly on strongly unbalanced datasets.

To fully assess KEN capabilities, we gradually increased the k value required by the algorithm, starting from a low k value and incrementally increasing it until its fine-tuned version was reached. This incremental approach allowed us to identify the critical *threshold value* whereby the compressed model obtained results similar to its fine-tuned version or when the compression value k leads to a catastrophic decline of performances, as reported in Apx. B.1. To provide a comprehensive analysis of KEN, we selected different transformer models with unique architecture, attention mechanisms, training approaches or different versions of the same model. Tab.3.5 compares the architectures of the models examined, emphasizing the number of layers and the number of parameters of each.

Model	Trainable params	`glue-sst2` Accuracy
Bert-base	109M	93.37
Hybrid	94M	93.23
HybridNT	94M	92.20
KEN	**80M**	**93.80**
Hybrid	66M	91.97
HybridNT	66M	90.71
Sajjad et al. [130]	66M	90.30
Gordon et al. [50]	66M	90.80
Flop	66M	83.20
KEN	**63M**	**92.90**

Table 3.7: Pruning algorithm comparations on SST-2 datasets

3.3.2.2 Model compression

Transformer models and other neural networks often have large file sizes, with a fine-tuned transformer potentially reaching up from 500 MB to 2GB or more in size. However, the KEN algorithm reduces this size by selecting and retaining a subset of k parameters while restoring the rest to their pre-trained values. This process creates a more concentrated model that only includes the essential k values for each matrix, resulting in significant weight reduction. To accurately assess the weight reduction achieved by KEN, we save the compressed model generated during this phase and compare it to its original, unpruned version. To ensure a fair comparison, we use the same technique to save both the compressed and original fine-tuned models. Nevertheless, KEN requires a support file, such as a dictionary, to load the k parameters saved into their appropriate positions during the loading process. This is because during loading, the k fine-tuning values must be loaded into a pre-trained model and the support file provides the necessary mapping to ensure proper placement. Sec. 3.3.3.2 provides a comprehensive overview of the compression results obtained during this analysis.

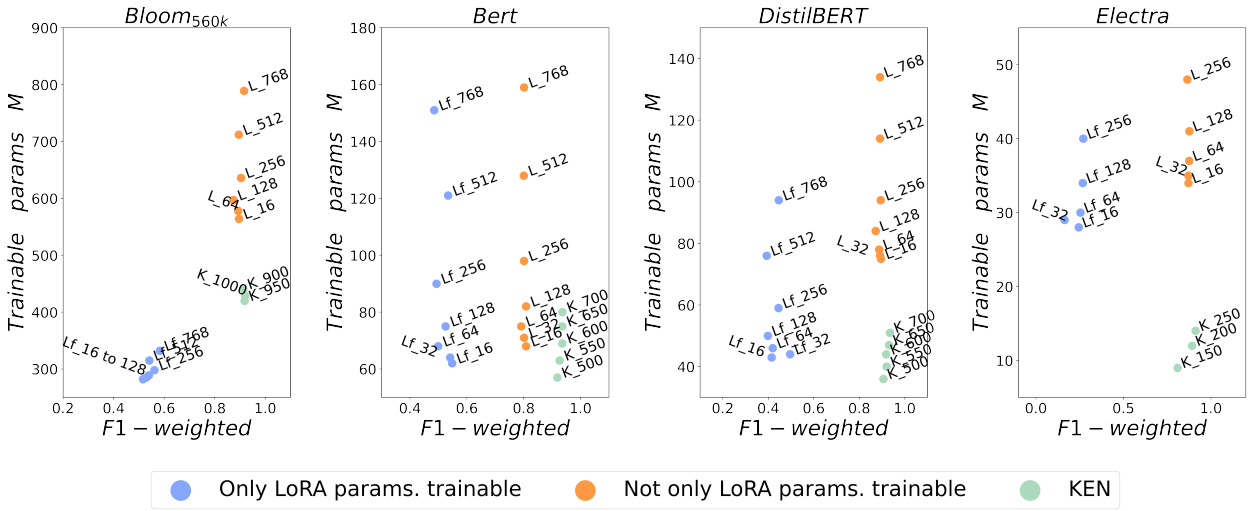


Figure 3.5: Comparison between KEN and LoRA. Labels for the LoRA marker indicate the dimension of the rank-decomposition matrix analyzed while, for KEN, the k value used

3.3.3 Results on sentiment analysis datasets

This subsection presents the results obtained for each KEN main goal. Sec. 3.3.3.1 discusses the effectiveness of KEN-pruned models compared to their unpruned counterparts, pruning benchmarks and state-of-the-art PEFT algorithm. Sec. 3.3.3.2 focuses on saving and loading the subnetwork extracted by KEN, comparing the reduced file sizes it achieves with those of the original models. Finally, Sec. 3.3.3.3 shows KEN$_{viz}$, illustrating its applications.

3.3.3.1 Experiment results

To evaluate the efficacy of KEN, we conducted a series of experiments across diverse classification and sentiment analysis datasets. For each dataset, we implemented KEN multiple times, employing varied k values and calculating the mean and standard deviation of the resulting F1-weighted scores. The complete dataset list can be found in Apx. B.2. As evidenced in Tab. 3.6, KEN successfully compresses all analyzed models without sacrificing their original, unpruned performance. We observed a remarkable reduction in overall model parameter count, ranging from a minimum of 25% to a substantial $\approx$ 70% for certain models. Intriguingly, the models with both the highest and lowest parameter counts exhibited

the most significant parameter reduction. Additionally, for each model under examination, we observed no substantial difference in performance as the percentage of reset parameters increased, maintaining a remarkable resemblance to the unpruned model performance. This observation underscores KEN exceptional generalization capability, striking a balance between performance and compression even at middle-to-high compression rates.

We compared KEN to other pruning algorithms specifically designed for transformer models, including FLOP [161], Hybrid [78] and HybridNT [78]. It is essential to note that Lagunas et al. [78] models (Hybrid and HybridNT) only prune the attention layers and not the entire model. To facilitate a comprehensive and standardized comparison of all algorithms, we recalibrated the size of their models based on our holistic perspective, ignoring any partial considerations. We combined the results obtained in their publication with those obtained from KEN and FLOP in Tab. 3.7. KEN outperformed all other compared models with a significant performance gap while utilizing fewer parameters in every instance. In addition to these findings, we conducted a thorough analysis of FLOP, which is the most complete pruning algorithm studied and, like KEN, decomposes original matrices to derive pruned ones. We conducted additional experiments on all models analyzed, using the datasets listed in Tab. 3.6. We compared the results obtained from FLOP with those of KEN, which employed fewer parameters than FLOP. As shown in Tab. 3.8, FLOP outperforms KEN in only one instance. For all other models and datasets analyzed, KEN consistently outperforms FLOP.

Although KEN belongs to the *winning ticket pruning* algorithms family, it shares similarities with Parameter Efficient Fine-tuning (PEFT) algorithms. This is because both approaches aim to identify a subset of optimal parameters within the fine-tuned model. We conducted a thorough evaluation of KEN and compared it to LoRA, wch is currently the state-of-the-art PEFT algorithm. We applied LoRA and KEN to the same layers of each model. We then trained the LoRA-based models for five times more epochs than their KEN-based counterparts. Additionally, we gradually increased the number of rank decomposition

Model	Pruning algorithm	Trainable params.	AG-NEWS	EMO	IMDB	YELP_POLARITY	glue-sst2
$BLOOM_{1B7}$	KEN	531M	**92.2** (±0.1)	**90.6** (±0.1)	**84.2** (±0.1)	**96.3** (±0.1)	**90.9** (±0.1)
	FLOP	1.1B	90.1 (±1.3)	84.0 (±1.9)	80.9 (±0.3)	85.5 (±3.5)	80.7 (±1.7)
$BLOOM_{560k}$	KEN	404M	**91.3** (±0.1)	**85.5** (±3.5)	**81.3** (±0.3)	**94.8** (±0.5)	**92.0** (±0.4)
	FLOP	408M	91.0 (±0.6)	84.0 (±2.3)	72.1 (±7.1)	87.0 (±0.5)	81.8 (±0.5)
DeBERTa	KEN	84M	**91.4** (±0.6)	**88.9** (±1.5)	**82.5** (±3.1)	**96.0** (±0.2)	**92.8** (±0.4)
	FLOP	88M	90.6 (±0.7)	83.1 (±1.7)	81.1 (±0.8)	91.4 (±0.1)	82.3 (±1.1)
Bert	KEN	57M	**91.6** (±0.7)	**86.0** (±0.5)	**84.9** (±0.8)	**93.8** (±1.6)	**92.8** (±0.5)
	FLOP	66M	90.9 (±0.9)	83.3 (±0.8)	80.5 (±0.6)	90.2 (±0.6)	83.2 (±0.2)
Ernie	KEN	57M	**91.5** (±1.4)	**88.3** (±0.4)	**87.6** (±0.6)	**95.7** (±0.1)	**94.1** (±0.4)
	FLOP	67M	89.8 (±0.4)	83.8 (±2.3)	81.1 (±0.8)	90.9 (±0.1)	83.2 (±0.9)
DistilBERT	KEN	40M	**91.9** (±0.3)	**88.2** (±1.1)	78.1 (±1.4)	**94.1** (±0.1)	**89.2** (±0.7)
	FLOP	45M	90.7 (±0.9)	83.2 (±1.2)	**81.2** (±0.9)	90.7 (±0.1)	82.4 (±1.2)
Electra	KEN	14M	**91.3** (±0.2)	**85.6** (±0.3)	**84.3** (±0.1)	**93.7** (±0.4)	**90.1** (±0.1)
	FLOP	28M	90.9 (±0.3)	83.1 (±2.1)	81.2 (±0.1)	90.5 (±0.1)	81.1 (±0.3)

Table 3.8: Comparation between KEN and FLOP pruning algorithms on different datasets. Mean and standard deviation are calculated on equal runs for each dataset and algorithm analyzed. The *Trainable params* column indicates the number of parameters used by each algorithm after the pruning phase.

matrices for each model from 16 to its original matrix size. In each LoRA-based experiment, only the LoRA-specific parameters were designated as either trainable or not. Our results, presented in Fig. 3.5, demonstrate that KEN consistently outperforms LoRA in terms of F1-measure while utilizing fewer trained parameters. However, when LoRA parameters are not the only ones trained, it produces similar results to KEN but consistently maintains a higher parameter count.

These compelling results provide strong evidence supporting our hypothesis that strategically selecting a subset of parameters and resetting the remainder offers a promising alternative to conventional pruning techniques.

3.3.3.2 Compression values

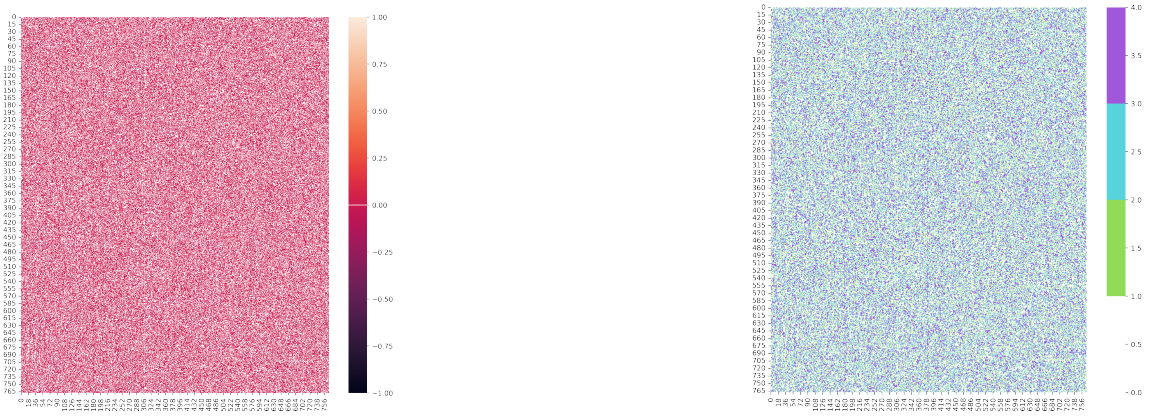
One of the primary objectives of KEN is to significantly reduce the overall size of transformer models, including their file sizes. To accomplish this goal, KEN leverages a subnetwork

Model	Total params	Original file size	# trainable params	Compressed file size (Model + support dict)
$BLOOM_{1B7}$	1.72B	7,055 MB	664M	**3,071 MB** (2,923 + 148)
			442M	**2137 MB** (2,013 + 124)
$BLOOM_{560k}$	560M	2,294 MB	429M	**2,084 MB** (1,956 + 128)
			386M	**1,842 MB** (1,731 + 111)
BERT	109M	438 MB	80M	**358 MB** (320 + 38)
			57M	**260.2 MB** (228 + 32.2)
DistilBERT	66M	266 MB	51M	**231.4 MB** (203 + 28.4)
			36M	**165 MB** (145 + 20)
DeBERTa	138M	555 MB	107M	**476.3 MB** (428 + 48.3)
			76M	**348.4 MB** (306 + 42.4)
Ernie	109M	438 MB	80M	**356.9 MB** (320 + 36.9)
			57M	**260.3 MB** (228 + 32.3)
Electra	33M	134 MB	14M	**67.01 MB** (59.1 + 7.91)
			9M	**42.58 MB** (35.5 + 7.08)

Table 3.9: Comparison of the .pt file size between the original and compressed transformer weights

comprising only k-trained parameters, allowing it to be saved and then injected into its pre-trained counterpart. This process requires a support file, like a dictionary, that specifies the precise location of each saved parameter within the pre-trained model. To ensure a fair comparison between the original and compressed model sizes, the compressed model is saved using the same techniques and format as the original model, guaranteeing consistent results. For each model, two compressed versions are generated, employing both high and low k values.

As shown in Tab. 3.9, both versions of the compressed models exhibit substantial memory savings, with their size directly proportional to the number of saved parameters. Specifically, models saved using a high k value, and thus closely mirroring the structure of the unpruned model, also conserve significant memory. This value further increases as the number of trained parameters saved diminishes. The support dictionary for parameter injection, stored using the Lempel-Ziv-Markov chain data compression algorithm, has an insignificant impact on the model final weight, which remains significantly smaller than the original. Furthermore, the time required to load the injected parameters into the pre-trained model is linear with



(a) Reset params: 47.92%

(b) Neighor counts

Figure 3.6: Output of KEN$_{viz}$ of the key attention matrix at layer 12 of a BERT model trained on `glue-sst2`. (a) show the matrix after the KEN pruning stage while (b) its neighbor counts.

the transformer architecture and the compression employed.

3.3.3.3 KEN$_{viz}$

KEN$_{viz}$ is a visualization tool that provides a clear understanding of matrices composition after the application of KEN pruning step. It offers various views to explore the pruned model, including:

1. **Single Matrix View**: It displays only the retained parameters, leaving the pruned ones blank (Fig. 3.4).

2. **Neighbor Count View**: It visualizes the number of non-zero neighbors (horizontally and vertically) for each point in a given matrix.

3. **Layer-wise View**: This iterative view applies the previous two views to each matrix per model layer.

The examples in Fig. 3.6 and Apx. B.3 both indicate that the number of non-zero neighbors for each point remains consistently high even in cases with high reset parameters. This suggests that the chosen parameters not only represent the most effective elements but also display a well-proportioned distribution within each matrix.

Chapter 4

Experiments

Accurate neoantigen presentation prediction is crucial for developing effective cancer immunotherapies. To train and evaluate models for this task, we need high-quality training and test corpora. To this end, we propose a novel approach to generate these datasets by incorporating false examples that mimic real antigens.

Our methodology involves several key steps:

- **Data Preprocessing** (Sec. 4.1.1): We begin by cleaning and filtering the CARMEN dataset (Sec. 3.1), a comprehensive resource for neoantigen analysis. This involves removing unnecessary information and ensuring data consistency (Sec. 4.1.1.1), as well as the training and test splitting (Sec. 4.1.1.2) and the analysis of the resulting datasets (Sec. 4.1.1.3).

- **False Example Generation** (Alg. 2): We introduce an algorithm to create synthetic, false antigen examples that resemble real antigens. These examples are carefully designed to challenge the model and improve its ability to distinguish between genuine and fabricated antigens.

- **Dataset Creation** (Sec. 4.1.2): We generate two corpora with varying ratios of false to real examples: 1:1 and 1:5. Each corpus is further divided into balanced and unbalanced versions to explore the impact of class imbalance on model performance.

- **Dataset Analysis** (Sec. 4.1.2.2): We conduct a thorough analysis of the generated datasets to ensure their representativeness and quality. This includes examining the distribution of peptide lengths, HLA alleles, and other relevant features.

By following these steps, we create robust and diverse datasets that are well-suited for training and evaluating models for neoantigen presentation prediction. The proposed approach addresses the challenges of limited positive examples and helps to improve the generalization capabilities of machine learning models in this domain.

After the corpora generation, this chapter delves into the experimental setup for our neoantigen presentation prediction task, focusing on the TAPE model and its configuration (Sec. 4.2). We begin by discussing the tokenization process and then explore the hyperparameters employed on the model in its training step, including: learning rate, batch size, and optimization algorithm. We also outline the training procedure, encompassing the division of data into training and validation sets, the early stopping technique, and the loss function used for model evaluation.

The final part of the chapter focuses on the KEN experiments (Sec. 4.3), discussing the strategy adopted and the dimension of the sub-networks implemented during our experiments.

4.1 Corpora Generation for Neoantigen Classification

In machine learning, a model learns features by training itself on specific datasets. Based on the training process, models can be categorized as either *pre-trained* or *fine-tuned*.

Pre-training involves initializing a model with random or null weights and training it on large and unsupervised datasets. Rather than focusing on a specific task, the goal is to learn a general-purpose embedding space. This learned representation can then be adapted to various downstream applications.

Fine-tuning, on the other hand, involves adjusting the pre-trained weights of a model on a specific task, such as classification. This allows the model to specialize in a particular domain.

TAPE (Sec. 3.2) is a pre-trained model that has been trained on a diverse set of benchmarks. Our goal is to fine-tune TAPE for *neoantigen presentation prediction* task. To create suitable training and test data for this classification task, we generate false examples that mimic real antigens. This enables the model to learn to distinguish between genuine and fabricated antigens and their corresponding HLA bindings.

We propose two corpora with varying ratios of false to real examples: 1:1 and 1:5 and, for each of them, two different versions (*balanced* and *no balanced*). Both corpora are derived from the CARMEN dataset (Sec. 3.1), which is well-suited for training models due to its characteristics.

To prepare the data for training, we perform several preprocessing steps outlined in Sec. 4.1.1. These steps involve cleaning the original CARMEN dataset and generating the training and test sets.

Additionally, Sec. 4.1.2 introduces our algorithm for generating false examples within the datasets, obtaining the 1:1 and 1:5 classification corpora already declared.

4.1.1 Data Pre-processing

This section provides a comprehensive overview of the data preparation process for our neoantigen presentation prediction task. We begin by cleaning and filtering the CARMEN dataset in Sec. 4.1.1.1. This involves removing unnecessary information to create a more streamlined dataset for subsequent analysis. In Sec. 4.1.1.2, we merge the cleaned datasets and split them into training and test sets. The final datasets contain essential features such as peptide sequences, HLA markers, and protein IDs, as illustrated in Tab. 4.3.

To ensure the quality and representativeness of the generated datasets, we conduct a thorough analysis in Sec. 4.1.1.3. This analysis includes examining the distribution of peptide lengths and HLA alleles within both the training and test sets.

4.1.1.1 Cleaning and Filtering Data

A crucial step before generating the corpora involves *pre-processing* or *cleaning* the data. This step eliminates unnecessary information, resulting in a lighter, more readable, and usable dataset. CARMEN is a collection of four different datasets (Sec. 3.1.2) and these information are spread inside three of them (`main`, `annotation-MSFragger` and `annotation-pogo`).

Filtering the Main Dataset The `main` dataset, the largest in CARMEN, required significant filtering due to its numerous unnecessary features.
In particular, we removed the following attributes:

- **Study id**: Reference to the peptide database origin
- **Sample name**: Unique identifier for each example
- **Peptide type**: Canonical, not canonical, or Nan
- **Spectral count**: Number of spectral counts
- **Assigned modifications**: String or Nan

- **Netmhcpan binder**: Indicates NetMHCpan-4.1 binding

After filtering, the main dataset retained only:

- **peptide**: Peptide string
- **Haplotype**: HLA-related genetic marker

Filtering Annotation Datasets: PoGo and MSFragger The `annotation-msfragger` and `annotation-pogo` datasets required less extensive cleaning. Both primarily contain binding information for peptide-to-genomic coordinate mapping. To determine if a peptide in the main dataset has a mapping in PoGo or MSFragger, we removed the following features from `annotation-pogo`:

- Transcript id
- Protein end
- Gene end
- Gene id
- Chromosome
- Protein start
- Gene start
- Strand

Retained just the features:

- peptide
- protein id

A similar filtering process was applied to `annotation-msfragger`, retaining only *peptide* and *Mapped protein annotation*, deleting only *protein start* and *protein end* features.

4.1.1.2 Merging Datasets and Creating Training and Test Sets

Once all datasets have been cleaned, the next step is to merge them into a comprehensive final dataset, which will then be divided into training and test sets.

Our final goal is to obtain two datasets (training and test), containing the essential information for neoantigen presentation: *Peptide*, *HLA Marker*, and the *Protein ID* to which the peptide belongs. A graphical result is shown in Tab. 4.3.

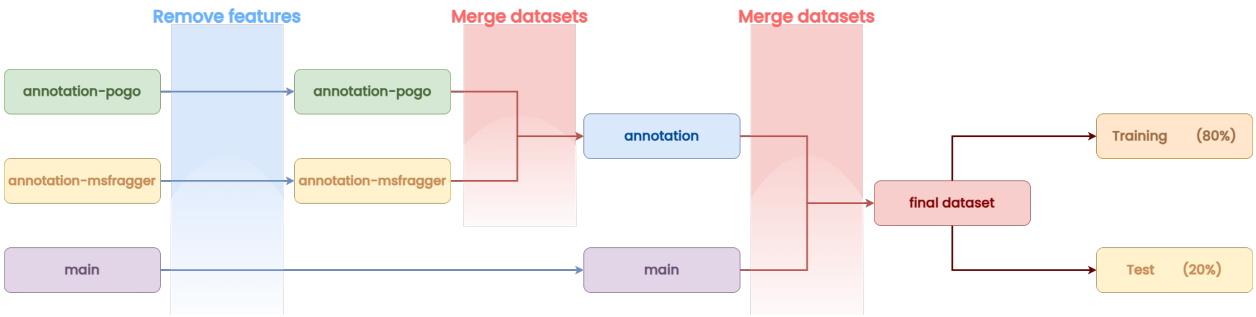


Figure 4.1: Training and test corpora generation

First, we combined the annotation datasets to create a unified dataset named `annotation`, mapping all peptides to their genomic coordinates. Each peptide may be associated with multiple HLAs, and both the `annotation-pogo` and `annotation-msfragger` datasets collect a list of HLAs for each unique peptide. After merging, we expanded this list to include every HLA for each peptide, resulting in a complete list of peptide-HLA pairs.

Next, we removed any duplicate peptide-HLA pairs from the final dataset, resulting in a total of 6,761,561 unique peptide-HLA sequences.

The final step was to merge the annotation dataset with the main dataset to obtain our final corpus. We performed this merge using a SQL join command with *Peptide* as the primary and foreign key. For each peptide in the main dataset, we added all the features from the annotation dataset where the peptide matched. This resulted in a large dataset containing 42,612,982 unique records.

We then split the final dataset into training and test sets using an 80/20 split.
To ensure that each peptide is associated with only one protein and does not appear in both

#	peptide	HLA	Protein_id
0	HYFQNTQGLIF	B*07:02	ENSP00000000233
1	YFQNTQGLIF	B*39:06	ENSP00000000233
2	HYFQNTQGLIF	B*35:03	ENSP00000000233
3	VWDVGGQDK	B*35:01	ENSP00000000233
4	HYFQNTQGLIF	B*39:01	ENSP00000000233
...	...	...	...

Table 4.1: Training dataset

#	peptide	HLA	Protein_id
0	LMLEAMNNL	A*02:17	ENSP00000473836
1	RRHMPLRLA	A*31:01	ENSP00000473836
2	RRRHMPLRL	A*32:01	ENSP00000473836
3	RRHMPLRL	A*32:01	ENSP00000473836
4	RRRHMPLRL	A*31:01	ENSP00000473836
...	...	...	...

Table 4.2: Testing dataset

Table 4.3: Fragment of the resulting datasets used to generate the classification ones

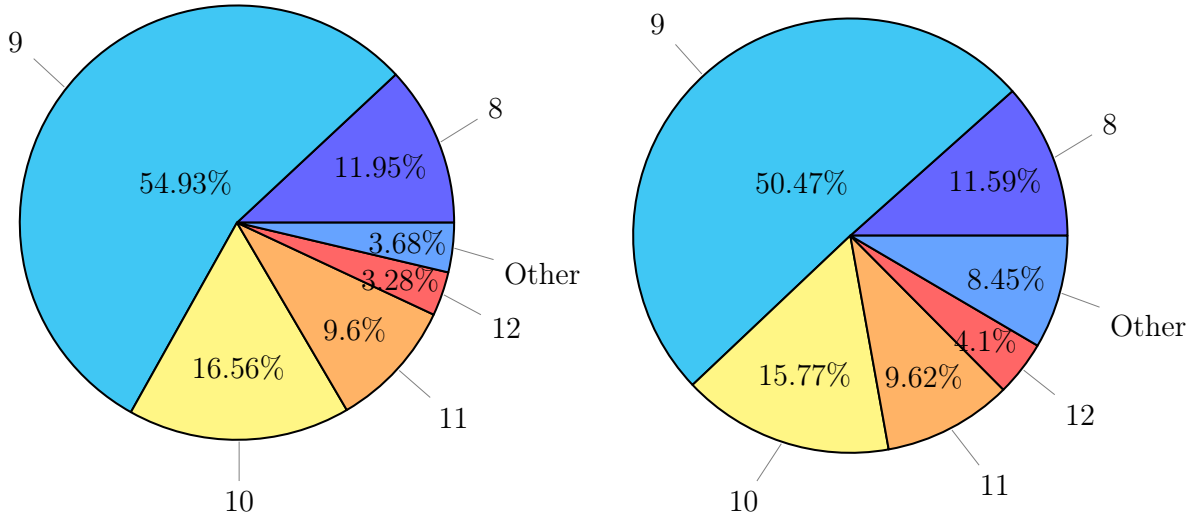


Figure 4.2: Distribution of peptide lengths in the training set (left) and test set (right). Each slice corresponds to a peptide length (e.g., 9 = length 9). Peptides shorter than 9 or longer than 12 are grouped under the *Other* category

the training and test sets, we split the dataset based on the unique values of the `Protein ID`. Specifically, we selected the first 80% of the unique protein IDs for the training set and the remaining 20% for the test set. The training set consists of 76,017 unique protein ID values, ranging from `ENSP00000000233` to `ENSP00000473833`, while the testing set contains 22,333 unique protein ID values, ranging from `ENSP00000473836` to `ENSP00000510811`. As a result of this split, the training set contains a total of 29,829,183 unique records, while the test set includes 12,783,799 unique records.

4.1.1.3 Analysis of the Resulting Datasets

To ensure the generated training and test sets accurately represent the key characteristics of the original CARMEN dataset, we analyzed the distribution of peptide lengths and HLA alleles within both sets.

Peptide Length Distribution As shown in Figure 4.2, the peptide length distributions in the training and test sets are highly similar, with comparable proportions across all length categories. Following the methodology outlined in Sec. 3.1.2 and Apx. A.0.1, we focused on

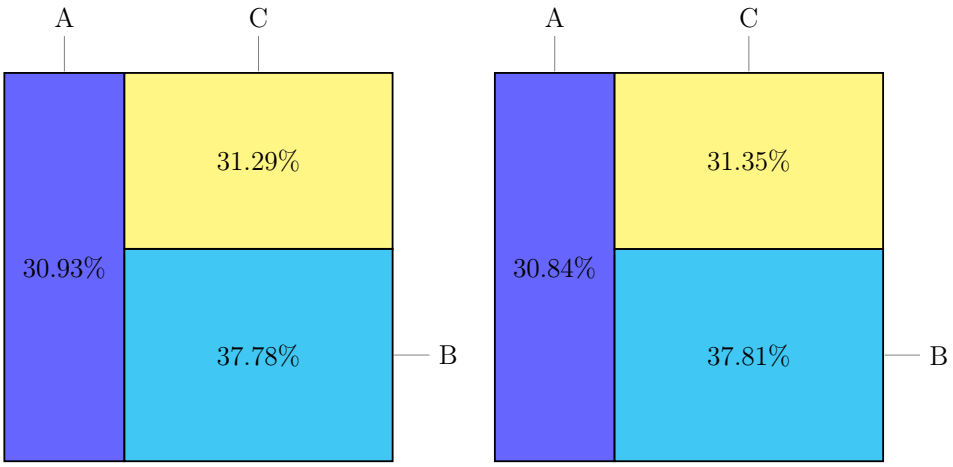


Figure 4.3: Distribution of HLA-I classes in the training set (left) and test set (right). Labels A, B, and C refer to HLA-A, HLA-B, and HLA-C, respectively

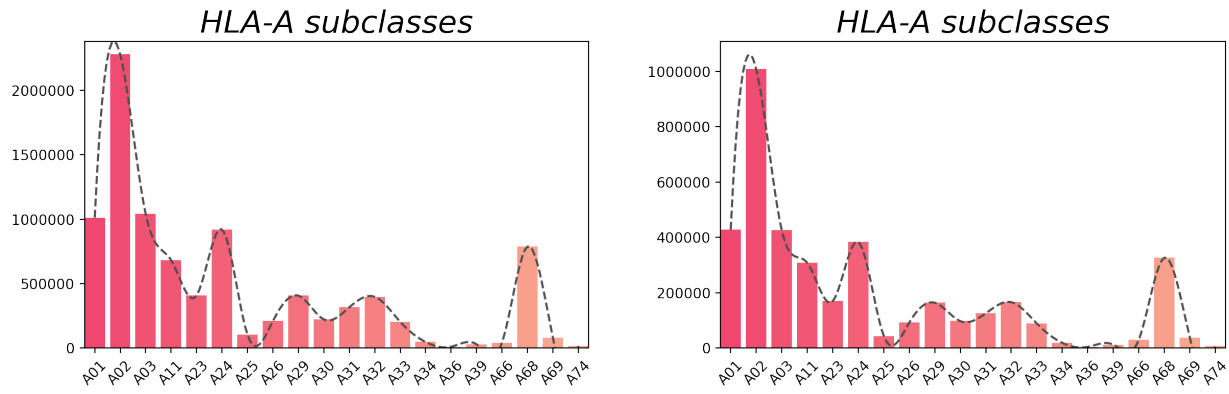
peptides with lengths between 9 and 12, grouping all others into the *Other* category.

The distribution confirms that peptide length is well preserved across both datasets, with peptides of length 9 being the most frequent, followed by lengths 10 and 11. Although the training set shows a slightly stronger bias toward length 9 compared to the test set, the overall distributions remain very close. Notably, the *Other* category — encompassing peptides shorter than 9 or longer than 12 — is slightly larger in the test set.

Overall, the pie charts demonstrate that the generated training and test sets closely mirror the original CARMEN dataset in terms of peptide length distribution, providing a representative and balanced foundation for model training and evaluation.

HLA Distribution A fundamental requirement for building unbiased models is ensuring that the distribution of HLA classes remains consistent across training and test sets. As shown in Fig. 4.3, the overall distribution of HLA-I classes (HLA-A, HLA-B, HLA-C) is well-balanced and nearly identical in both sets, enabling fair and generalizable model training. To further investigate the dataset composition, we examined the distribution of individual HLA subclasses for each class. The histograms in Figs. 4.4–4.6 show that the original distributions from the CARMEN dataset have been accurately preserved in our corpus.

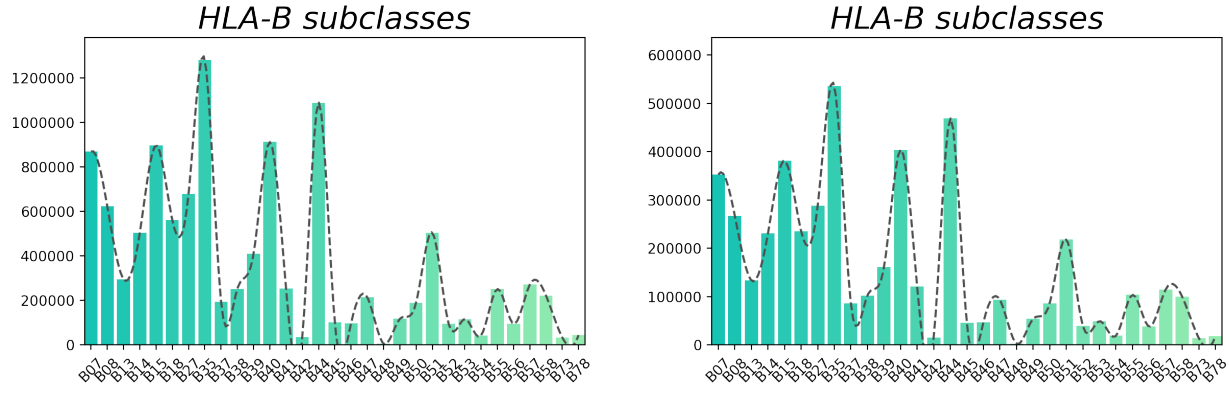
Focusing on HLA-A (Fig. 4.4), alleles such as A*02, A*68, and A*24 are the most rep-



(a) Training set (b) Test set

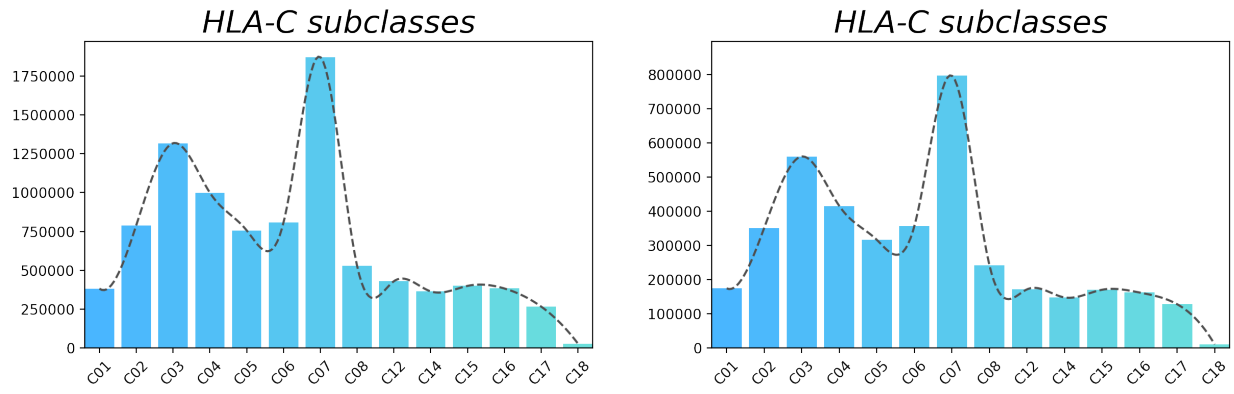
Figure 4.4: Distribution of HLA-A alleles in the training (a) and test (b) sets. Bars indicate absolute allele frequency, while a dashed curve overlays the histogram using cubic spline interpolation to reveal distribution trends

resented across both splits. Minor differences with the original distribution reported in Apx. A.0.3.1 may occur due to data cleaning steps described in Sec. 4.1, though these do not significantly affect the global balance. A similar consistency is observed for HLA-B (Fig. 4.5), with B*35, B*44, and B*40 being the most frequent alleles. HLA-B is also the most abundant class overall, as discussed in Apx. A.0.3.2. Lastly, HLA-C, the least represented class, maintains its intra-class balance across datasets (Fig. 4.6). The most common



(a) Training set (b) Test set

Figure 4.5: Distribution of HLA-B alleles in the training (a) and test (b) sets. Bars indicate absolute allele frequency, while a dashed curve overlays the histogram using cubic spline interpolation to reveal distribution trends



(a) Training set

(b) Test set

Figure 4.6: Distribution of HLA-C alleles in the training (a) and test (b) sets. Bars indicate absolute allele frequency, while a dashed curve overlays the histogram using cubic spline interpolation to reveal distribution trends

alleles are C*07, C*03, and C*04, again in line with the original dataset (Apx. A.0.3.3).

To further confirm this alignment, we generated comparison plots (see Fig. 4.7) that overlap the normalized frequency distributions of training and test sets for each HLA class. The normalization was performed using *MinMax scaling*, followed by cubic spline interpolation for smooth comparison curves. As visible in these plots, the distributions are nearly identical, highlighting the quality and fairness of our data splitting procedure.

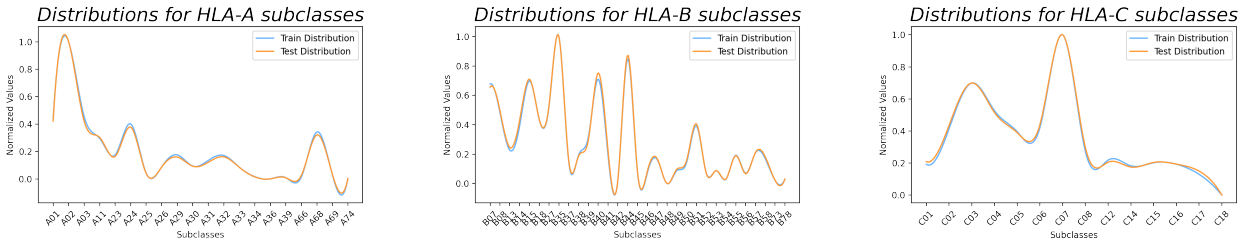


Figure 4.7: Normalized frequency distributions for each HLA-I class (HLA-A, HLA-B, and HLA-C) in the training and test sets. Each curve is obtained using cubic spline interpolation over Min-Max scaled values to allow a direct comparison between the two datasets, regardless of absolute frequency differences

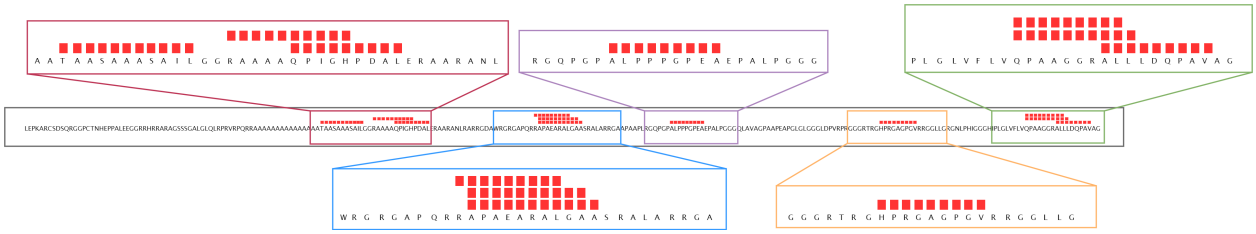


Figure 4.8: Antigens in `ENSP00000497242` protein. Red boxes highlight the locations of our peptide within the selected protein.

4.1.2 Neoantigen Classification Datasets

After preprocessing the CARMEN dataset, splitting it into training and test sets, and ensuring that both subsets preserve the original dataset characteristics, we are now prepared to transform CARMEN into a corpus for neoantigen presentation classification.

To construct a classification dataset, we introduce artificially generated negative examples, referred to as *decoys*. This process begins with a thorough analysis of the protein sequences from which the peptides in both the training and test sets originate. Our decoy generation strategy represents a novel approach not previously described in the literature.

However, we acknowledge a limitation: we cannot guarantee the biological *falsehood* of these decoy peptides. Although they are derived from the same protein sequence as their corresponding positive (true) peptides, this does not ensure biological dissimilarity. More importantly, we cannot definitively rule out the possibility that a given `<decoy-HLA>` pair might naturally occur within the human body.

Nevertheless, since these datasets are intended for fine-tuning machine learning models that have already undergone pretraining, this innovative decoy generation method could offer a fresh perspective in the field. Furthermore, the goal of both the classifier and this thesis is to accurately identify true peptides — which, thanks to CARMEN (Sec. 3.1), are known to be biologically valid and prevalent in the human body.

Protein Sequence and Peptide Validation Both the training and test datasets generated included peptide, HLA, and Protein_id information (Tab. 4.3). However, protein

sequences were absent. These sequences are crucial for determining the exact positions of antigens within proteins. In addition, this information is essential for mapping all the (true) antigens present in the protein and making and verifying the positioning of its respective (false) decoys (Fig. 4.8). Thus, we employed an external dataset to map each `Protein_id` to its corresponding full sequence.

To prevent false examples from being actual neoantigens, we created an *oracle* dictionary listing all possible true peptides within each dataset.

False Example Generation With complete peptide lists in both the datasets and corresponding protein sequences, we can generate false examples. To ensure the validity of decoys, we excluded those containing `X`, `U`, or insertion/deletion symbols (`*`) as some pretrained models, like MHCFlurry or NetMHCpan [122], might not process these characters accurately.

For each true peptide, we randomly selected a non-overlapping substring of equal length within its protein sequence. If this substring met the character restriction and was not in the oracle dictionary, it was considered a suitable decoy. The corresponding HLA ID was then assigned to the decoy, and the decoy was labeled as false.

Partial Overlap and Dataset Complexity To increase the complexity and biological realism of the dataset, we introduced partial overlaps between true (antigenic) and false (decoy) peptides. In this setup, a decoy peptide may share a segment of its amino acid sequence with a true peptide within the same protein. This reflects a realistic scenario, as such overlaps naturally occur in biological systems due to conserved or similar regions within proteins as graphically shown in Fig. 4.9. Importantly, the overlap is only partial—decoy peptides never exactly replicate a true peptide but may contain subsequences derived from them.

We ensured that the classification task was more challenging and representative of real-world conditions by including peptides with partial overlap. This increased the difficulty

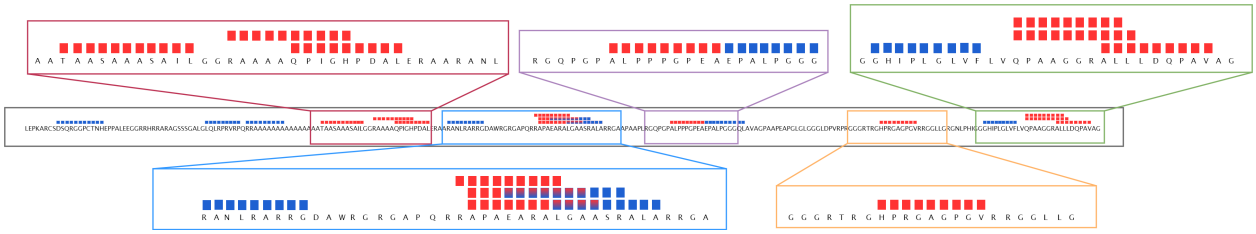


Figure 4.9: Antigens and decoys in `ENSP00000497242` protein. Red boxes indicate true peptides (antigens), while blue boxes represent decoys. Overlapping regions between red and blue boxes illustrate partial sequence overlap, where a decoy shares a segment with a true peptide. This setup reflects the biological complexity and realistic challenges introduced in the dataset

for the model in distinguishing between true and false peptides, ultimately improving its generalization ability and performance on unseen data.

1:1 and 1:5 Datasets To investigate the impact of decoy-to-antigen ratios on model performance, we generated two datasets: 1:1 and 1:5.

In the 1:1 dataset, we generated a single decoy for each antigen. Conversely, the 1:5 dataset included five different decoys for each selected antigen. This approach allowed us to evaluate how varying the number of false examples influenced the model ability to discriminate between true and false peptides.

For a comprehensive understanding of the dataset generation process, Alg. 2 outlines the steps involved. Additionally, Fig. 4.8 illustrates the positions of antigens within the `ENSP00000497242` protein, while Fig. 4.9 visualizes the positions of both antigens and generated decoys within this protein.

4.1.2.1 Balanced datasets

In the realm of Big Data, data collection necessitates careful consideration of the *5 Vs*: volume, value, variety, velocity, and veracity. One of the primary challenges encountered during the creation of 1:1 and, especially, 1:5 datasets is ensuring data variety. As illustrated in Alg. 2, decoy generation involves verifying their distinctiveness from the (true) antigens extracted from the CARMEN dataset. However, the intrinsic nature of decoys, generated

Algorithm 2 Decoys Generation

Data: P_s, /* Protein sequence */
p_s, /* Antigen sequence */
h_p, /* HLA sequence */
$oracle$ /* Oracle dictionary */

Result: d_s: /* Decoy sequence */
h_d, /* HLA decoy sequence */

$i \leftarrow \text{find}(p_s[0], P_s)$ /* Find antigen start in P_s */
$j \leftarrow \text{find}(p_s[-1], P_s)$ /* Find antigen end in P_s */

while True **do**
  $decoy\_start \leftarrow \text{random}(0, n)$ /* Generate decoy start index */
  if $decoy\_start \notin [i, j]$ **then** /* Check antigen no overlap index */
    break
  end if
end while
$decoy\_end \leftarrow decoy\_start + \text{len}(p_s)$ /* Generate decoy end index */
$d_s \leftarrow P_s[decoy\_start : decoy\_end]$ /* Suitable decoy */

while True **do**
  if $\{\texttt{X}, \texttt{U}, *\} \notin d_s$ **then** /* Check trigger values */
    break
  end if
  loop: *generate new index*
end while

while True **do**
  if d_s in oracle **then** /* Check if not a real antigen */
    loop: *generate new index*
  else
    break
  end if
end while
$h_d \leftarrow h_p$ /* Set the decoy HLA */
return d_s, h_d

randomly with lengthy protein sequences, does not guarantee their absolute uniqueness.

Moreover, as discussed in Sec. 4.1.1.1, the CARMEN dataset is partitioned by protein ID, and duplicate pair <antigens, HLA> are eliminated during the cleaning process. However, as shown in Tab.4.3, the same peptide is not deleted if linked with a different HLA. To

		Train		**Test**	
Version	Type	*Antigens*	*Decoys*	*Antigens*	*Decoys*
No Balanced	1:1	29,492,292	29,492,292	4,632,103	4,632,103
	1:5	29,492,292	147,461,460	4,632,103	23,160,515
Balanced	1:1	401,973	400,983	130,293	130,293
	1:5	401,973	1,979,438	130,293	651,465

Table 4.4: Neoantigens classification datasets composition

address these concerns, we developed *balanced versions* of 1:1 and 1:5. These datasets are derived from the original datasets by removing all duplicates, retaining only a single instance in the test set and detecting any subsequent occurrences. This approach ensures that the balanced versions contain exclusively unique antigens and decoys, making them true subsets of the original datasets.

4.1.2.2 Analysis of the obtained datasets

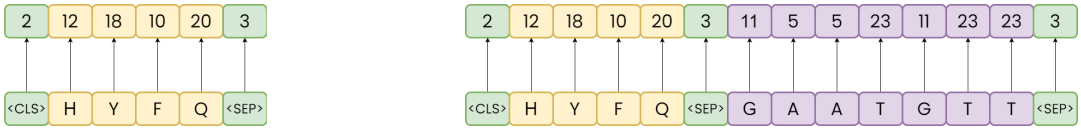
Tab. 4.4 presents the dimensions of the generated datasets. As outlined in previous sections, the *no balanced* versions are substantial, with the 1:1 training dataset alone comprising 58 million records and an additional 9 million for testing. For the 1:5 configuration, these figures increase to approximately 176 million for training and 27 million for testing.

In contrast, the *balanced* versions exhibit significantly smaller dimensions. This reduction is primarily due to the removal of duplicate peptides and decoys during dataset generation focusing solely on unique peptides without considering the pair <peptide, HLA>.

The cleaning steps occur at different stages, starting with removing duplicate antigens during data import and culminating in another comparison and cleaning step in the final dataset. If an antigen is deleted, it is not replaced.

As described in Sec. 4.1.2.1, if a duplicated peptide or decoy is found, all occurrences in the training data are deleted, leaving only one in the test set. Consequently, the dataset reduction is more concentrated in training, with, for example, just one thousand more true antigens in the 1:1 version compared to the test set.

While the no-balanced versions maintain a training-to-test ratio of 86:14, the balanced



(a) Peptide tokenization (b) `< Peptide, HLA >` tokenization

Figure 4.10: Tokenization strategies: (a) peptide-only input, (b) peptide-HLA input

versions have a more balanced ratio of 75:25.

The no-balanced datasets aim to assess whether machine learning models can effectively identify true antigens within a large and relatively homogeneous dataset. Conversely, the balanced datasets focus on evaluating model performance under conditions where duplicates and potential biases have been minimized. By comparing the results from both versions, we can gain insights into the impact of dataset balance on model performance.

4.2 TAPE Setup Experiments

This section outlines the experimental configurations for our TAPE-based experiments (Section 3.2). Before diving into the model architecture and hyperparameters, we begin with tokenization.

As a BERT family model, TAPE inherits its input processing requirements, including tokenization. For tokenization, TAPE employs the IUPAC vocabulary, a standardized lexicon specific to chemistry. The IUPAC vocabulary, developed and maintained by the International Union of Pure and Applied Chemistry (IUPAC), provides consistent terminology for chemical elements, compounds, reactions, properties, and units. Its use ensures clarity and avoids misunderstandings in scientific communication.

The tokenizer assigns numerical values to peptide sequences. Special characters, such as `<pad>`, `<mask>`, `<cls>`, `<sep>`, and `<unk>`, are assigned values 0 to 4. To represent peptide sequences numerically, the tokenizer converts each character to its corresponding numerical value. Additionally, it adds a `<cls>` token at the beginning of each sequence and a `<sep>` token to separate sequences.

We conduct two experiments: one training a TAPE model using only the peptide sequence as input to assess its ability to distinguish real antigens from decoys, and another using both peptide and HLA sequences to investigate whether HLA sequences, as important immune response indicators, can enhance model performance.

Consequently, the tokenization process for the input of these experiments differs slightly. In the second experiment, we need to include the HLA sequences. To achieve this, we tokenize the peptide and HLA sequences separately and concatenate them, omitting the `<cls>` token as it is already included during peptide transformation.

Fig. 4.10 illustrates the tokenization process for both experiments.

After the tokenization step, we ensure that all inputs have the same length by padding. Padding is a machine learning technique that involves adding blank characters to make the model input uniform. In both experiments, we set the maximum sequence length to match the longest sequence in the input.

After splitting the training data into training and validation sets using an 80-20 ratio, the input is ready to train the model with a batch size of 128.

For the model configuration, we used the TAPE base configuration, consisting of 12 layers like a BERT-base architecture. We added a dropout of 0.1 to avoid overfitting and a linear layer as a classification layer.

As output of TAPE and input for the classification layer, we extract the hidden states of the `<cls>` token from the pooled output of the BERT model for each input sequence in the batch. These hidden states are then used as input to the subsequent layers for the classification task.

The `<cls>` token is often used as a representation of the entire input sequence. By extracting its hidden states, the model can capture the relevant semantic information from the input data and use it to make predictions.

For the training step, we trained the model for a maximum of 10 epochs using an early stopping technique to stop the model training if the difference between the previous validation

Hyperparameter	Value	Description
Batch size	128	Number of samples processed in each training iteration
Epochs	10	How many time the model is trained on the same dataset
Maximum sequence length	Determinated by the longest sequence in the input	Maximum length of the input sequences
Learning rate	$2e^{-5}$	Step size for updating model parameters
Dropout rate	0.1	Probability of dropping neurons during training to prevent overfitting
Early stop patience	2	Number of epochs without improvement before early stopping
Early stop Δ	0.002	The minimum validation accuracy improvement needed to not stop the training step
Criterion	Cross Entropy	Function to measure the loss function
Optimizer	AdamW	Mathematical procedure to minimize the loss function

Table 4.5: Hyperparameters used in the TAPE training experiments

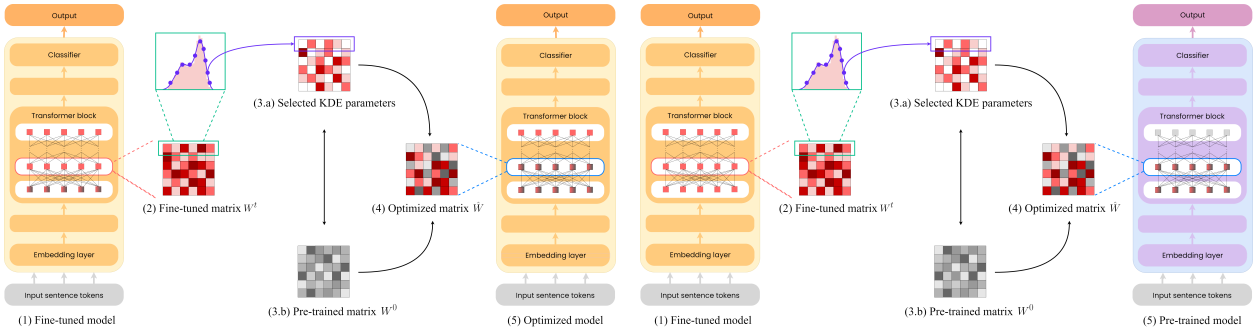
accuracy and the current one is lower than 0.002 ($\Delta = 0.002$) for a maximum of two rounds (patience = 2).

For the loss function calculation, we used Cross Entropy, which is a commonly used loss function for classification tasks, especially when dealing with categorical data. It calculates the negative log-likelihood of the correct class given the model-predicted probability distribution.

The loss function is crucial for guiding the model learning process by penalizing incorrect predictions and rewarding accurate ones.

As an optimizer, we used AdamW [84], a popular optimization algorithm that combines the benefits of Adam and weight decay with a learning rate $\alpha = 2e^{-5}$.

A brief recap of all the parameters used, is shown in Tab. 4.5.



(a) KEN *extract* mode

(b) KEN *inject* mode

Figure 4.11: KEN dual approach *extract* vs *inject*. Both the extract and inject methods in KEN involve the same process of identifying the optimal sub-network within a fine-tuned model. However, they differ in how the sub-network is integrated into the final model: in extract (a), the non-selected parameters are reset within the fine-tuned model to their pre-trained values, while in inject (b) the optimized sub-network is injected into a pre-trained matrix

4.3 KEN Setup Experiments

The KEN experiments aim to determine if an optimal sub-network can be identified that closely mimics the performance of the original network. Additionally, we sought to understand how the model sub-network varies across different trained models based on their input information. For instance, one research question was to investigate whether TAPE models trained on peptide-only and peptide-HLA data would share the same optimal sub-network or if one would achieve the original fine-tuned performance more rapidly than the other.

To conduct these experiments, we first trained models as described in Sec. 4.2. We then saved the weights of each experiment for subsequent use in KEN. Essentially, these experiments were extensions of the work presented in Sec. 3.3.2.1.

After importing the model and its corresponding fine-tuned weights, we began to eliminate non-essential fine-tuned parameters. We employed a straightforward approach: the KEN algorithm was configured in two different modes: *extract* and *inject*. In the extract mode, KEN identifies the optimal sub-network within a fine-tuned model and uses it for testing. In the inject mode, KEN does not extract the optimal sub-network but instead

identifies its location and injects it into a pre-trained model for testing. While there is no functional difference between these two methods, we opted for the inject mode in our experiments to facilitate repeated iterations and explore whether KEN consistently extracts different optimal sub-networks. Using the inject mode, we preserved the integrity of the fine-tuned model and could simply update parameters within a pre-trained model.

With this configuration in place, we can now delve into the specific experiments. As outlined in Sec. 3.3, KEN utilizes a parameter k that determines the number of optimal parameters to retain in each layer. For example, if layer l contains n parameters, KEN will preserve k fine-tuned parameters and reset the remaining $n - k$ ones.

We initiated our experiments with a low k value (35 million parameters) and gradually increased it until we reached the original fine-tuned dimension (92 million). During the pruning process, KEN identifies the optimal sub-network across all layers, from the embedding layer to the classification layer. This makes the pruning process non-homogeneous, as setting a fixed k value for all layers can lead to disproportionate pruning in larger layers like the embedding layer while smaller layers, such as bias layers, may be pruned less. However, this is one of KEN strengths, as its row-based approach ensures universal applicability without requiring specific layer considerations.

Once the optimal sub-network was selected within the fine-tuned model using a k value, we injected it into a pre-trained (non-fine-tuned) TAPE model and evaluated its performance on our neoantigen classification testing datasets.

Chapter 5

Results

This chapter delves into the performance of TAPE on the corpora we created. In Sec. 5.1.1, we thoroughly analyze the training phase across eight case studies, emphasizing training performance and outcomes for each instance. In each case, the models were trained using both peptide-only and peptide+HLA inputs to evaluate the influence of HLA sequences. TAPE consistently delivers exceptional results in all scenarios, revealing some surprising trends, particularly regarding the impact of HLA sequences.

To delve deeper into this trend, Sec. 5.1.2 presents the testing outcomes, concentrating specifically on the role of HLA sequences during this phase. Although TAPE impressive capabilities are confirmed, integrating HLA sequences as input does not improve performance compared to using peptide sequences alone. Indeed, as the ROC curves indicate, the performance is largely consistent across all eight models, suggesting that peptide-only inputs are sufficient to achieve comparable results.

To enhance our understanding of TAPE performance and how our generated corpora affect neoantigen presentation, we performed additional experiments with MHCFlurry and TransPHLA. As described in Sec. 5.1.3.1, we evaluated MHCFlurry using our balanced datasets and observed that TAPE consistently outperforms it across all test cases. Conversely, TransPHLA, detailed in Sec. 5.1.3.2, was evaluated on the Pfam dataset. While

TAPE performance matches that of TransPHLA, our generated corpora offer better properties due to the absence of overlapping data, unlike Pfam, which may suffer from redundancy and repeated entries.

Finally, we implemented the KEN pruning algorithm to create optimal subnetworks for TAPE and examined the effect of pruning on models trained with and without HLA sequences. As demonstrated in Sec. 5.2, KEN efficiently prunes the model, achieving performance comparable to unpruned models while utilizing approximately 52% of the training parameters. Interestingly, in contrast to the results observed in full-scale models, HLA sequences prove crucial for improved performance in pruned models, enabling them to reach baseline performance earlier and more reliably compared to peptide-only input.

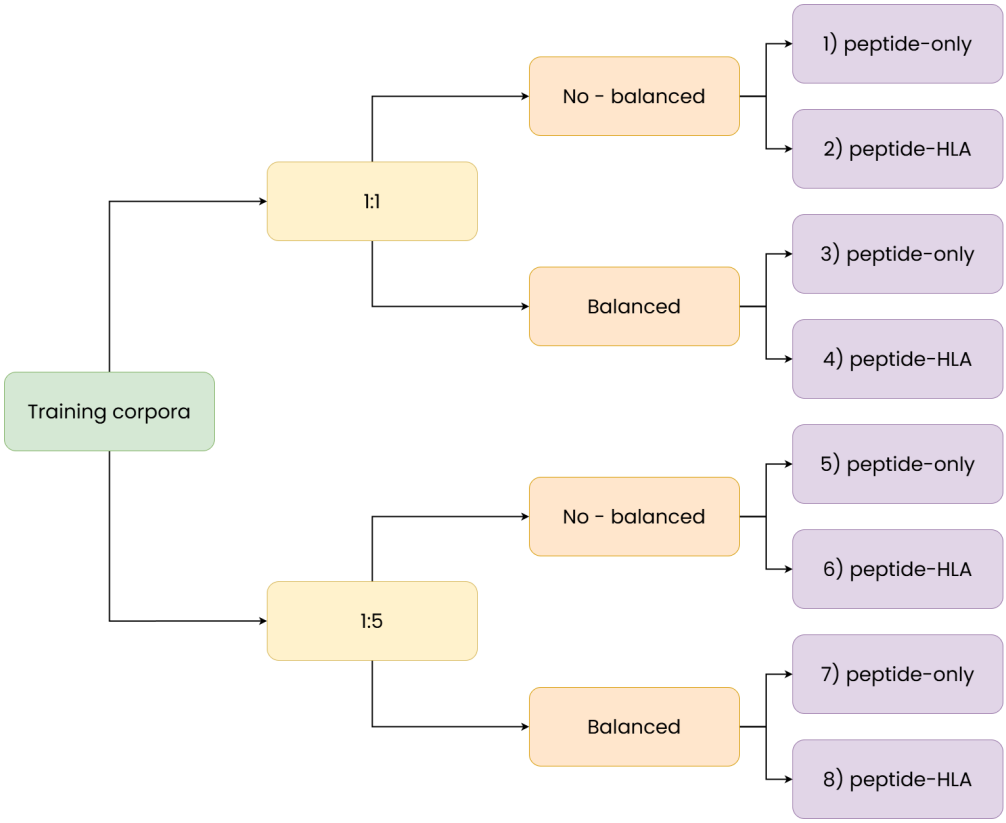


Figure 5.1: TAPE experiments provided based on the corpora and the input given

5.1 TAPE Results

5.1.1 Training results

To evaluate TAPE performance on our generated datasets, we fine-tuned our model using the hyperparameters outlined in Tab. 4.5. We trained eight distinct TAPE models, varying the training corpus and input data. Details of these experiments can be found in the workflow shown in Fig.5.1 or extensively described in Sec. 4.2.

Due to the implementation of early stopping, none of the eight trained models reached the maximum of 10 epochs. This occurred because the validation accuracy did not improve by at least 0.002 during the final two epochs.

All models demonstrated impressive performance, with both training and validation accuracies consistently exceeding 90%. However, a deeper analysis revealed some unexpected re-

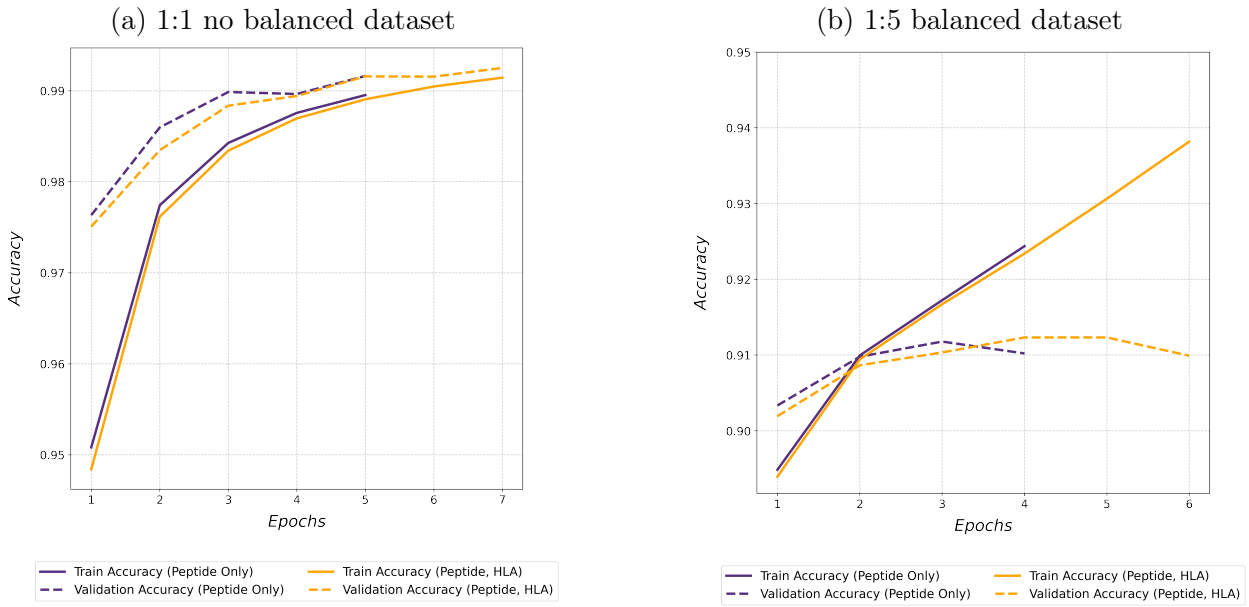


Figure 5.2: Training results

sults: the peptide-only models consistently stopped training earlier than their HLA-inclusive counterparts. This was counterintuitive, as we had expected the additional HLA data would facilitate faster convergence.

Surprisingly, when comparing the performance of the peptide-only models with the peptide-HLA models across different corpora, we found their performance curves were nearly identical. This suggests that the inclusion of HLA had minimal impact on the training process, as the performance of the model closely mirrored that of the peptide-only model from the very first epoch. Fig. 5.2 illustrates this finding with two representative examples, comparing balanced (1:5) and unbalanced (1:1) datasets. In both cases, the training and validation curves for the two model types were virtually indistinguishable. While the peptide-HLA model in Fig. 5.2b appears to slightly outperform the peptide-only model, the difference in accuracy is negligible (approximately 0.015).

In conclusion, although TAPE demonstrated excellent performance in all eight scenarios, the results were unexpected. Including HLA data did not yield the anticipated benefits regarding training speed or accuracy.

Dataset	Type	Precision	Recall	F1 true label	AROC
1:1	w/o HLA	0.982	0.982	0.980	0.983
	with HLA	0.984	0.984	0.984	0.984
1:5	w/o HLA	0.965	0.970	0.968	0.997
	with HLA	0.965	0.970	0.968	0.997
Balanced					
1:1	w/o HLA	0.838	0.899	0.867	0.938
	with HLA	0.871	0.878	0.874	0.946
1:5	w/o HLA	0.758	0.656	0.703	0.941
	with HLA	0.773	0.740	0.756	0.953

Table 5.1: Testing results on the eight cases provided

5.1.2 TAPE test results

After training eight TAPE models on various versions of our corpora, we evaluated their performance on testing sets to determine if the training phase irregularities persisted. As shown in Tab. 5.1, the testing results are consistently high across all metrics and test cases.

A deeper analysis confirmed the training phase trends: the performance difference between models using peptide-only input and peptide-HLA pair input was minimal in all cases. While Precision, Recall, and F1-score were generally higher when HLA information was included, the improvement was negligible.

To better understand these subtle differences, we examined ROC curves. These curves visualize a model performance across different classification thresholds, with higher and more left-shifted curves indicating better performance. By analyzing ROC curves for different testing datasets, we gained further insights. The ROC curves for unbalanced datasets (1:1 and 1:5 ratios) are shown in Fig. 5.3. The curves for models with and without HLA data exhibit significant overlap, further confirming that HLA inclusion offers minimal benefit. The AROC values in Table 5.1 reinforce this observation, with differences between the two model types being negligible. To eliminate potential biases introduced by peptide repetition, we conducted further evaluations using balanced datasets. These datasets were designed to ensure no overlap between training and testing peptides, providing a more reliable assessment.

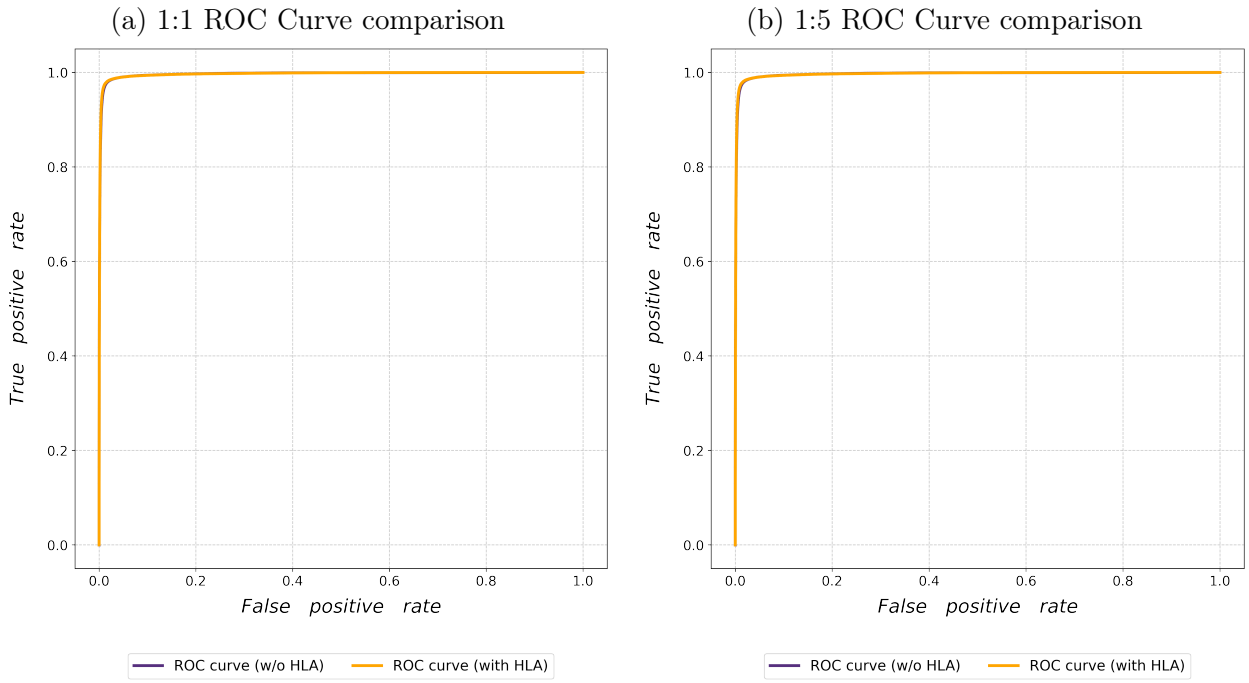


Figure 5.3: ROC curves using 1:1 and 1:5 unbalanced datasets

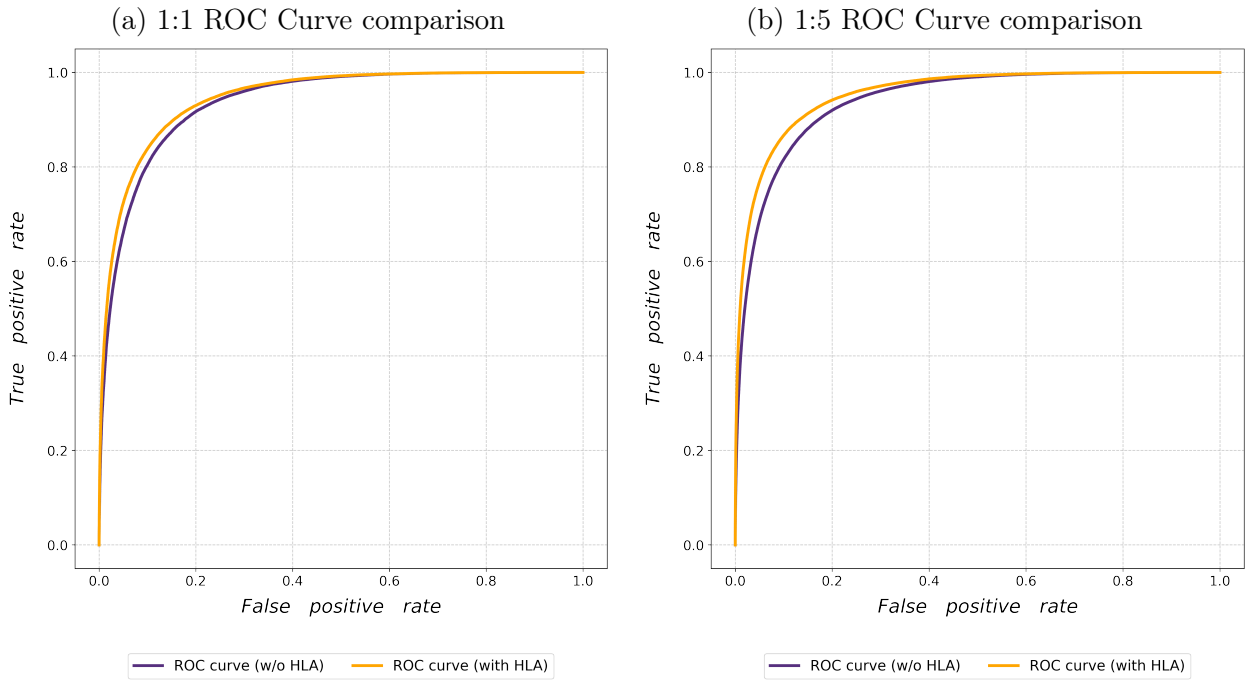


Figure 5.4: ROC curves using 1:1 and 1:5 balanced datasets

As illustrated in Fig. 5.4, models incorporating HLA sequences showed only marginal improvements compared to peptide-only models. These findings align with the training phase results, reaffirming that peptide-only inputs are sufficient for achieving comparable performance. The inclusion of HLA data did not significantly enhance the predictive capabilities of the models, as differences in AROC values were minimal (≈ 0.01).

Overall, the test results of the TAPE experiments provide valuable insights. While the models consistently achieved high performance across all scenarios, the inclusion of HLA information did not yield the anticipated benefits. These findings suggest that peptide-only inputs are sufficient for effective prediction, reaffirming the hypothesis that peptide data alone can deliver equivalent predictive capabilities.

5.1.3 Additional Analysis

To further assess the robustness and generalizability of our model and dataset generation strategy, we conducted two complementary analyses. First, we compared our results against predictions made by the well-established model MHCflurry. This comparison allows us to understand the performance of a widely-used tool when applied to our generated datasets and evaluate how it fares against our own TAPE-based model (Sec. 5.1.3.1). Second, we benchmarked our approach against the state-of-the-art TransPHLA model, trained on the popular Pfam dataset, in order to highlight how training on curated but overlapping datasets may lead to inflated performance and reduced generalization. Our goal in both analyses is to validate the effectiveness of our dataset generation process and emphasize the limitations of commonly adopted benchmark datasets in the literature.

5.1.3.1 MHCflurry Evaluation

MHCflurry [112] is an open-source machine learning framework designed to predict peptide–MHC class I binding affinity and antigen presentation likelihood. It plays a pivotal role in computational immunology and proteomics pipelines, especially in contexts such as cancer

neoantigen discovery and vaccine design. The platform is composed of three independent predictive modules:

- **Binding Affinity (BA)**: a regression model that estimates the binding strength (often expressed as IC50) between a peptide and a specific MHC class I allele.
- **Antigen Processing (AP)**: a deep learning model that approximates intracellular biological processes involved in antigen presentation, such as proteasomal cleavage, TAP transport, and ER aminopeptidase trimming.
- **Presentation Score (PS)**: a logistic regression that integrates the outputs of the BA and AP models to generate a final score representing the likelihood that a peptide will be presented on the cell surface in complex with a class I MHC molecule.

One of MHCflurry main strengths lies in its ability to generalize to a broad range of peptide sequences and HLA alleles, including rare or non-canonical variants. This is made possible by training on large experimental datasets that include both binding assays and mass spectrometry-derived peptide identifications. MHCflurry also offers a user-friendly interface (both CLI and Python API), which has contributed to its widespread adoption in the proteomics community.

Despite its flexibility, MHCflurry requires both a peptide and an associated HLA allele as input. This constraint makes it incompatible with some of our previously used datasets, which did not include HLA annotations. Therefore, in our analysis, we only used datasets classified as *Type: with HLA*, where both peptide and HLA identifiers are available. Moreover, in order to challenge the model while ensuring fair evaluation, we used only the *Balanced* versions of our datasets. As demonstrated in Tab. 3.2, these datasets are more challenging due to a more realistic positive-to-negative ratio and reduced redundancy, which make model overfitting less likely. As shown in Tab. 5.2, our TAPE-based model significantly outperforms MHCflurry across all evaluation metrics, including precision, recall, F1-score, and AUROC. In particular, MHCflurry consistently exhibits low recall, which strongly affects its

Dataset	Model	Precision	Recall	F1 (True Label)	AUROC
1:1 (Balanced)	TAPE	0.871	0.878	0.874	0.946
	MHCflurry	0.890	0.226	0.360	0.844
1:5 (Balanced)	TAPE	0.773	0.740	0.756	0.953
	MHCflurry	0.618	0.226	0.331	0.844

Table 5.2: Performance comparison between TAPE and MHCflurry on balanced datasets with HLA annotations

F1-score, despite achieving relatively high precision. This suggests that while MHCflurry is confident in its positive predictions, it fails to identify a substantial proportion of true binders, especially in datasets that were not seen during its training. In contrast, TAPE shows both high precision and high recall, indicating a better balance between sensitivity and specificity. These results highlight the effectiveness of contextualized representations and the importance of using clean, non-overlapping training data.

5.1.3.2 TransPHLA and the Pfam Dataset

To complement the above analysis, we evaluated the performance of our model against TransPHLA [26], a Transformer-based architecture designed specifically for peptide-HLA binding prediction. TransPHLA was trained on a curated subset of the Pfam database [101], which has become a widely-used benchmark in neoantigen prediction studies.

The TransPHLA model architecture is composed of four main components:

1. Amino acid embedding layers with positional encodings,
2. Two separate Transformer encoder blocks for peptides and HLA sequences,
3. Fully connected layers for feature extraction and optimization,
4. A final joint encoder block followed by a softmax output layer to produce binding probability predictions.

A unique characteristic of TransPHLA is the use of distinct encoders for peptide and HLA inputs, which are later merged to capture complex interaction patterns before final

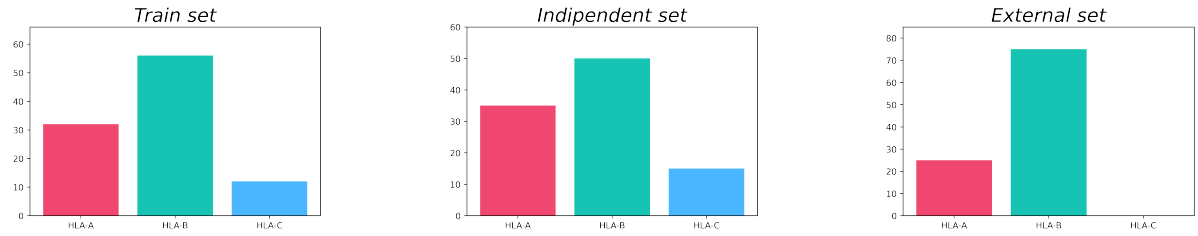


Figure 5.5: [Pfam sub-datasets composition]HLA class composition in the training, independent, and external subsets of the Pfam sub-datasets

classification.

While the Pfam subset used for training and evaluation of TransPHLA is balanced and covers a wide range of HLA alleles (Fig. 5.5), a closer inspection reveals a critical issue: *data overlap*. Approximately 30% of the peptides found in the training set are also present in the independent test set, and about 15% are repeated in the external test set. These overlaps are concerning, as they may result in overestimation of model performance due to data leakage.

In contrast, our generated datasets, especially the *Balanced* variants, were carefully curated to avoid any such overlap. This ensures a higher degree of generalizability and a more rigorous evaluation of model performance. Despite the fact that TransPHLA achieved strong results on its benchmark dataset (accuracy: 0.98, AUROC: 0.95, F1-score: 0.97), these numbers must be interpreted with caution due to the underlying data leakage. Our TAPE-based model, trained and evaluated on overlap-free, challenging datasets, not only achieves competitive results but also demonstrates a stronger capacity to generalize to unseen data.

Taken together, the analyses provided in Sec. 5.1.3.1 and Sec. 5.1.3.2 confirm the value of our dataset generation strategy and the robustness of our fine-tuned model. By avoiding overlapping training examples and leveraging contextualized sequence representations, we provide a more realistic and generalizable corpora for modeling peptide-HLA interactions in real-world scenarios.

Subset	Epitopes	Decoys
Train	359,166	1,795,830
Independent	85,876	85,562
External	51,984	51,881

Table 5.3: Epitope and decoy distribution in the Pfam subsets used for TransPHLA training and evaluation.

5.2 Optimized models using KEN

The TAPE model, akin to its BERT-like architecture, comprises 92 million parameters. As demonstrated in Sec. 5.1, TAPE excels in both training and testing phases, achieving high accuracy during training and impressive precision, recall, F1-score, and AROC scores during test.

Intriguingly, our analysis revealed that the HLA sequence, often considered crucial, has a surprisingly minimal impact on model performance. In fact, results obtained without including HLA were either comparable or superior to those achieved with HLA as input.

Motivated by this finding, we sought to identify the core components of TAPE performance and determine whether a smaller subnetwork could replicate its capabilities. To this end, we employed the KEN pruning algorithm (Sec. 3.3) with the configuration specified in Sec. 4.3.

With the *inject* configuration of KEN, we extracted subnetworks from each of the eight TAPE fine-tuned models (Fig. 5.1) by systematically increasing the parameter threshold k, from a small initial value to a value approaching 92 million.

Each optimized subnetwork was evaluated on the same test set used for its full-sized counterpart, and the results are presented in Tab. 5.1. KEN leverages Kernel Density Estimators to select parameters, a statistical method inherently subject to slight variations across iterations. To mitigate this variability, we repeated each experiment 10 times and compared the mean testing performance of each subnetwork to that of the unpruned TAPE model, taking into account the standard deviation obtained.

Dataset	Type	Trainable params							Baseline
		35M	39M	43M	48M	52M	56M	61M	92M
1:1	w/o HLA	0.844 (± 0.015)	0.893 (± 0.011)	0.923 (± 0.003)	**0.939 (± 0.002)**	**0.951 (± 0.001)**	**0.959 (± 0.001)**	**0.963 (± 0.001)**	0.981
	with HLA	0.869 (± 0.004)	0.899 (± 0.002)	0.919 (± 0.001)	**0.934 (± 0.001)**	**0.944 (± 0.001)**	**0.953 (± 0.001)**	**0.958 (± 0.001)**	0.982
1:5	w/o HLA	0.795 (±0.001)	0.797 (±0.003)	0.808 (±0.002)	0.789 (±0.002)	0.803 (±0.001)	0.830 (±0.003)	0.875 (±0.001)	0.989
	with HLA	0.849 (± 0.04)	**0.879 (± 0.013)**	**0.896 (± 0.011)**	**0.902 (± 0.006)**	**0.913 (± 0.004)**	**0.915 (± 0.002)**	**0.919 (± 0.001)**	0.920
Balanced									
1:1	w/o HLA	0.461 (± 0.095)	**0.831 (± 0.011)**	0.790 (± 0.052)	**0.822 (± 0.021)**	**0.848 (± 0.007)**	**0.858 (± 0.003)**	**0.862 (± 0.003)**	0.865
	with HLA	**0.816** (±0.012)	**0.815** (±0.037)	**0.836** (±0.017)	**0.853** (±0.002)	**0.854** (±0.005)	**0.860** (±0.001)	**0.862** (±0.001)	0.865
1:5	w/o HLA	0.785 (±0.002)	0.787 (±0.002)	0.791 (±0.001)	0.777 (±0.001)	0.787 (±0.001)	0.803 (±0.003)	0.838 (±0.001)	0.979
	with HLA	0.785 (±0.002)	0.787 (±0.002)	0.791 (±0.001)	0.777 (±0.001)	0.787 (±0.001)	0.803 (±0.003)	**0.905** (±0.001)	0.907

Table 5.4: KEN pruning results obtained using the eight fine-tuned models provided. The metric used to compare the models is the F1-weighted. Mean and standard deviation are calculated on 5 runs per experiment. Bold results indicate a difference with the baseline model of less than 0.05.

The results of the KEN pruning process are summarized in Tab. 5.4. In nearly all eight analyzed cases, KEN successfully achieved performance comparable to the original model while utilizing approximately half of its fine-tuned parameters (48 million).

Specifically, for the non-balanced configuration, the most promising results were obtained with the 48-million parameter model, matching the performance of the unpruned counterpart, which had 92 million parameters. In the 1:5 configuration, the model incorporating HLA sequences reached baseline performance (defined as the unpruned model performance) more rapidly, requiring only 39 million parameters. In contrast, the model without HLA failed to attain baseline performance, even with increasing numbers of parameters.

Generally, we observed that HLA sequences facilitated the subnetwork convergence to the baseline F1-weighted score more quickly than the peptide-only configuration. For instance, in the 1:1 dataset, both models eventually reached similar performance to the baseline, but the peptide + HLA configuration (defined as *with HLA*) consistently outperformed the peptide-only configuration (*w/o HLA*). In the 1:5 dataset, the peptide + HLA configuration achieved comparable performance with just 39 million parameters, representing only 42% of the trained parameters.

For a more visual representation of these results, Fig. 5.6 presents the data from Tab. 5.4 as a line chart, highlighting the region where the KEN-pruned models exhibit performance

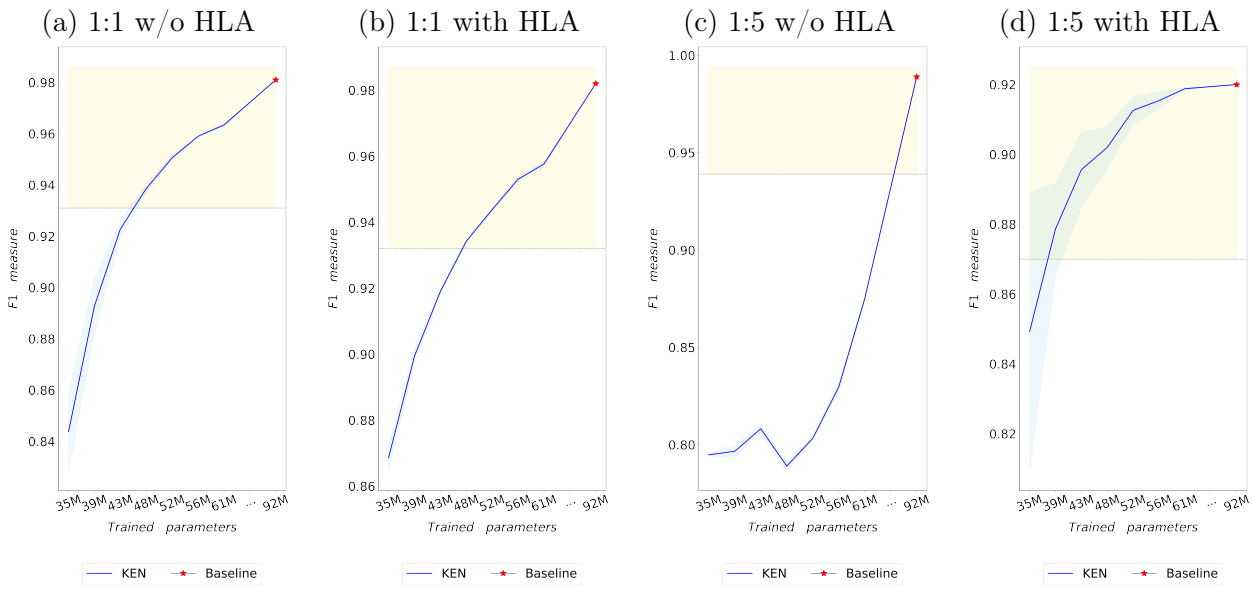


Figure 5.6: KEN pruning results on non-balanced configurations: The line charts display the F1-weighted scores achieved by the optimized subnetworks with different trainable parameters. The shaded blue area represents the standard deviation across 5 runs at each step. The red star marks the performance of the baseline (unpruned) model. The yellow shaded region highlights a range of ±0.05 standard deviations around the baseline F1-weighted score, as detailed in Tab. 5.4

similar to the baseline within a 0.05 margin F1-weighted score.

For the balanced configurations, the results were consistent (see Fig. 5.7). Using the 1:1 dataset, the model that incorporated both HLA and peptide inputs achieved performance levels comparable to the non-pruned model with only 35 million parameters (approximately 38% of the full model size). In contrast, the model that did not include HLA input experienced a significant drop in performance. Similarly, for the 1:5 dataset, the model with HLA input matched the non-pruned performance with 61 million parameters (approximately 66.3%), while the model without HLA required a greater number of parameters to achieve acceptable performance.

In conclusion, our experiments demonstrate the effectiveness of KEN pruning in significantly reducing the size of TAPE models while maintaining their performance. We found that HLA sequences, which were previously considered non-essential for training the TAPE model, actually played a vital role when used in a small, optimized model. In fact, models

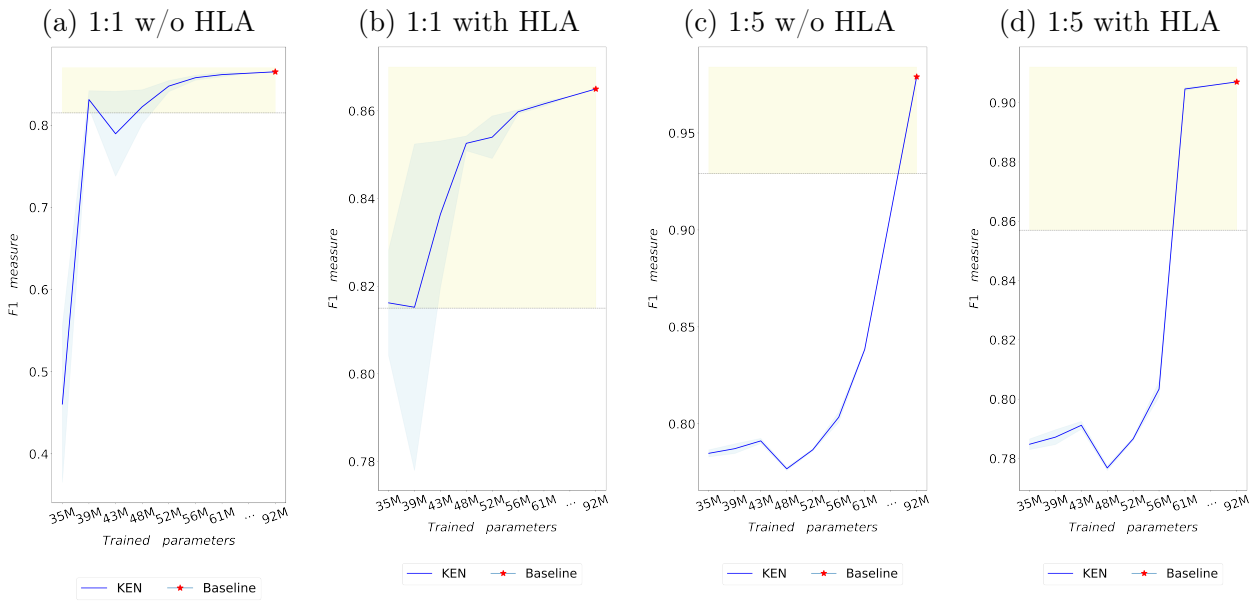


Figure 5.7: KEN pruning results on *balanced* configurations: The line charts display the F1-weighted scores achieved by the optimized subnetworks with different trainable parameters. The shaded blue area represents the standard deviation across 5 runs at each step. The red star marks the performance of the baseline (unpruned) model. The yellow shaded region highlights a range of ±0.05 standard deviations around the baseline F1-weighted score, as detailed in Tab. 5.4

trained using HLAs as input reached the same performance levels as the unpruned TAPE model sooner than those trained with peptides only, highlighting the crucial importance of HLAs in the task of neoantigen presentation classification.

In summary, across all the experiments conducted, we observed an unexpected result: incorporating HLA information did not improve model performance. Surprisingly, using only the peptide sequence led to comparable outcomes. This finding is counterintuitive, as most established models—such as NetMHCPan [122] and MHCFlurry [112]—rely heavily on HLA information either as a key input or, in the case of the CARMEN dataset, as a binding partner for peptide-HLA interactions.

Additionally, the comparison between TAPE and MHCFlurry demonstrated that TAPE outperforms MHCFlurry when both are evaluated on the same datasets. Notably, this performance gap widens when comparing MHCFlurry with TAPE in a peptide-only configuration.

However, as shown at the end of this chapter, HLA information should not be entirely

disregarded. It plays a crucial role when working with very small models. In such scenarios, including HLA data enhances model performance and accelerates convergence toward the accuracy of its non-pruned counterpart. These findings underscore the value of HLA data not only in biological contexts but also in machine learning. While its influence diminishes as model capacity and internal representation increase, it remains a powerful asset in constrained or lightweight settings.

Chapter 6

Conclusions

This thesis presents a comprehensive investigation into the application of advanced machine learning techniques for neoantigen classification, with a particular focus on the use of transformer-based architectures and model optimization strategies. The work addresses all fundamental stages of building a robust classification pipeline—from data collection and corpus construction to model evaluation and pruning—within the broader objective of enabling AI-driven support for personalized cancer therapies.

The starting point of this research was the CARMEN dataset, which aggregates peptides associated with HLA Class I alleles from reanalyzed 75 publicly available mass spectrometry datasets. From this resource, we developed eight distinct corpora specifically designed for neoantigen classification. These corpora vary across two key dimensions: (1) the ratio of decoy peptides generated per true peptide, which was either 1:1 or 1:5, and (2) the structure of peptide-HLA associations, defined as either balanced or non-balanced. In the non-balanced setting, a single peptide may appear in association with multiple HLA sequences, while in the balanced setting, each peptide is linked to exactly one HLA allele, introducing a greater level of difficulty due to reduced redundancy and increased specificity.

The decoy generation strategy implemented in this thesis represents a novel contribution to the field. Instead of sampling randomly, decoys were created by extracting alternative

sequences from the same source protein as the true peptides. While we cannot guarantee that these decoys are biologically irrelevant (since the peptide-HLA combination may potentially be valid in vivo), this approach grounds the decoys in a realistic biological context and avoids artificial randomness. Most importantly, our pipeline was designed to prevent any overlap between peptides used in training and those used in testing—an issue that affects many existing datasets, such as those derived from Pfam. This careful partitioning ensures that models are evaluated under truly generalizable conditions.

Each of the eight corpora was used to fine-tune TAPE, a transformer-based model inspired by the BERT architecture. The smallest corpus (*Balanced 1:1*) contained over 1 million peptide-HLA records, while the largest (*non-balanced 1:5*) reached nearly 232 million records. Despite the variability in size and structure, TAPE demonstrated strong and consistent performance across all configurations, with particularly interesting insights emerging when comparing input types. Notably, we observed that using the peptide alone as input, without the associated HLA allele, often led to similar performance as when the full peptide-HLA pair was used. This suggests that, in some contexts, the peptide sequence alone may encode sufficient information for accurate classification.

To explore this further, we introduced KEN, a pruning algorithm based on kernel density estimation, which allowed us to reduce model complexity without sacrificing performance. By resetting non-essential fine-tuned weights to their pretraining values, KEN enabled the pruning of more than 40% of model parameters, making the architecture lighter and more efficient for deployment in memory-constrained environments. Interestingly, pruning revealed that models trained with peptide-HLA input retained full-size model performance more effectively than those trained with peptides alone, especially under massive pruning. This indicates that HLA input contributes to model robustness, even when not strictly necessary for performance in full-scale scenarios.

Our approach was benchmarked against two state-of-the-art models: MHCFlurry and TransPHLA. MHCFlurry was directly evaluated on our corpora, revealing that TAPE consis-

tently outperformed it across all experiments conducted. On the other hand, TransPHLA—trained on Pfam subsets—showed similar accuracy but relied on overlapping training and test sets, which likely inflated its performance. Our corpora, free of such overlap, provide a more reliable ground for model assessment and reflect a higher standard of experimental rigor.

In summary, this work proposes a well-rounded pipeline for neoantigen classification that integrates new dataset creation strategies, comparative model evaluation, and an efficient pruning method. While the models developed are not intended for clinical deployment, they provide a foundation for further research toward more compact, interpretable, and effective tools in cancer immunotherapy. Conducted within the framework of the KATY project, this thesis contributes to the broader goal of leveraging artificial intelligence for personalized cancer treatment. The insights and methodologies developed here support future exploration at the intersection of machine learning and oncology, where interpretability and efficiency remain key priorities.

Bibliography

[1] Abele, R. and Tampë, R. (1999). Function of the transport complex tap in cellular immune recognition. *Biochimica et Biophysica Acta (BBA)-Biomembranes*, 1461(2):405–419.

[2] Abercrombie, G., Rieser, V., and Hovy, D. (2023). Consistency is key: Disentangling label variation in natural language processing with intra-annotator agreement. *arXiv preprint arXiv:2301.10684*.

[3] Akhtar, S., Basile, V., and Patti, V. (2021). Whose opinions matter? perspective-aware models to identify opinions of hate speech victims in abusive language detection. *arXiv preprint arXiv:2106.15896*.

[4] Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. (2019). Unified rational protein engineering with sequence-based deep representation learning. *Nature methods*, 16(12):1315–1322.

[5] AlQuraishi, M. (2019). Proteinnet: a standardized data set for machine learning of protein structure. *BMC bioinformatics*, 20:1–10.

[6] Apps, R., Meng, Z., Del Prete, G. Q., Lifson, J. D., Zhou, M., and Carrington, M. (2015). Relative expression levels of the hla class-i proteins in normal and hiv-infected cells. *The Journal of Immunology*, 194(8):3594–3600.

[7] Ba, J. and Caruana, R. (2014). Do deep nets really need to be deep? *Advances in neural information processing systems*, 27.

[8] Barbieri, F., Camacho-Collados, J., Espinosa Anke, L., and Neves, L. (2020). TweetEval: Unified benchmark and comparative evaluation for tweet classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1644–1650, Online. Association for Computational Linguistics.

[9] Basile, V., Cabitza, F., Campagner, A., and Fell, M. (2021). Toward a perspectivist turn in ground truthing for predictive computing. *arXiv preprint arXiv:2109.04270*.

[10] Bedran, G., Gasser, H.-C., Weke, K., Wang, T., Bedran, D., Laird, A., Battail, C., Zanzotto, F. M., Pesquita, C., Axelson, H., et al. (2023). The immunopeptidome from a genomic perspective: establishing the noncanonical landscape of mhc class i–associated peptides. *Cancer immunology research*, 11(6):747–762.

[11] Benbaki, R., Chen, W., Meng, X., Hazimeh, H., Ponomareva, N., Zhao, Z., and Mazumder, R. (2023). Fast as chita: Neural network pruning with combinatorial optimization. *arXiv preprint arXiv:2302.14623*.

[12] Bender, E. M., Gebru, T., McMillan-Major, A., and Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

[13] Bepler, T. and Berger, B. (2019). Learning protein sequence embeddings using information from structure. *arXiv preprint arXiv:1902.08661*.

[14] Berntsen, A., Brimnes, M. K., thor Straten, P., Svane, I. M., et al. (2010). Increase of circulating cd4+ cd25highfoxp3+ regulatory t cells in patients with metastatic renal cell carcinoma during treatment with dendritic cell vaccination and low-dose interleukin-2. *Journal of Immunotherapy*, 33(4):425–434.

[15] Blalock, D., Gonzalez Ortiz, J. J., Frankle, J., and Guttag, J. (2020). What is the state of neural network pruning? *Proceedings of machine learning and systems*, 2:129–146.

[16] Block, M. S., Suman, V. J., Nevala, W. K., Kottschade, L. A., Creagan, E. T., Kaur, J. S., Quevedo, J. F., McWilliams, R. R., and Markovic, S. N. (2011). Pilot study of granulocyte-macrophage colony-stimulating factor and interleukin-2 as immune adjuvants for a melanoma peptide vaccine. *Melanoma research*, 21(5):438–445.

[17] Bodmer, J. G., Marsh, S. G., Albert, E. D., Bodmer, W. F., Dupont, B., Erlich, H. A., Mach, B., Mayr, W. R., Parham, P., Sasazuki, T., et al. (1991). Nomenclature for factors of the hla system, 1990. *Immunobiology*, 182(3-4):334–345.

[18] Bodria, F., Panisson, A., Perotti, A., Piaggesi, S., et al. (2020). Explainability methods for natural language processing: Applications to sentiment analysis. In *CEUR Workshop Proceedings*, volume 2646, pages 100–107. CEUR-WS.

[19] Borden, E. S., Buetow, K. H., Wilson, M. A., and Hastings, K. T. (2022). Cancer neoantigens: challenges and future directions for prediction, prioritization, and validation. *Frontiers in oncology*, 12:836821.

[20] Boveri, T. (1914). *Zur frage der entstehung maligner tumoren*. Fischer.

[21] Buonaguro, L. and Tagliamonte, M. (2020). Selecting target antigens for cancer vaccine development. *Vaccines*, 8(4):615.

[22] Casás-Selves, M. and DeGregori, J. (2011). How cancer shapes evolution and how evolution shapes cancer. *Evolution: Education and outreach*, 4:624–634.

[23] Chatterjee, A., Narahari, K. N., Joshi, M., and Agrawal, P. (2019). SemEval-2019 task 3: EmoContext contextual emotion detection in text. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 39–48, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

[24] Chen, B., Khodadoust, M. S., Olsson, N., Wagar, L. E., Fast, E., Liu, C. L., Muftuoglu, Y., Sworder, B. J., Diehn, M., Levy, R., et al. (2019). Predicting hla class ii antigen presentation through integrated deep learning. *Nature biotechnology*, 37(11):1332–1343.

[25] Choi, M., Carver, J., Chiva, C., Tzouros, M., Huang, T., Tsai, T.-H., Pullman, B., Bernhardt, O. M., Hüttenhain, R., Teo, G. C., et al. (2020). Massive. quant: a community resource of quantitative mass spectrometry–based proteomics datasets. *Nature methods*, 17(10):981–984.

[26] Chu, Y., Zhang, Y., Wang, Q., Zhang, L., Wang, X., Wang, Y., Salahub, D. R., Xu, Q., Wang, J., Jiang, X., et al. (2022). A transformer-based model to predict peptide–hla class i binding and optimize mutated peptides for vaccine design. *Nature Machine Intelligence*, 4(3):300–311.

[27] Cignarella, A. T., Basile, V., Sanguinetti, M., Bosco, C., Rosso, P., and Benamara, F. (2020). Multilingual irony detection with dependency syntax and neural models. *arXiv preprint arXiv:2011.05706*.

[28] Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

[29] Cohan, A., Ammar, W., van Zuylen, M., and Cady, F. (2019). Structural scaffolds for citation intent classification in scientific publications. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3586–3596, Minneapolis, Minnesota. Association for Computational Linguistics.

[30] Comber, J. D. and Philip, R. (2014). Mhc class i antigen presentation and implications for developing a new generation of therapeutic vaccines. *Therapeutic advances in vaccines*, 2(3):77–89.

[31] CR, U. (2023). Worldwide cancer incidence statistics. *Cancer Res UK*.

[32] Craig, R., Cortens, J. P., and Beavis, R. C. (2004). Open source system for analyzing, validating, and storing protein identification data. *Journal of proteome research*, 3(6):1234–1242.

[33] Cuff, J. A. and Barton, G. J. (1999). Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins: Structure, Function, and Bioinformatics*, 34(4):508–519.

[34] Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 512–515.

[35] de Gibert, O., Perez, N., García-Pablos, A., and Cuadros, M. (2018). Hate Speech Dataset from a White Supremacy Forum. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 11–20, Brussels, Belgium. Association for Computational Linguistics.

[36] De Sousa Silveira, T., Uszkoreit, H., and Ai, R. (2019). Using aspect-based analysis for explainable sentiment predictions. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 617–627. Springer.

[37] Deutsch, E. W., Eng, J. K., Zhang, H., King, N. L., Nesvizhskii, A. I., Lin, B., Lee, H., Yi, E. C., Ossola, R., and Aebersold, R. (2005). Human plasma peptideatlas. *Proteomics*, 5(13):3497–3500.

[38] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[39] Dong, X., Chen, S., and Pan, S. (2017). Learning to prune deep neural networks via layer-wise optimal brain surgeon. *Advances in neural information processing systems*, 30.

[40] Efremova, M., Finotello, F., Rieder, D., and Trajanoski, Z. (2017). Neoantigens generated by individual mutations and their role in cancer immunity and immunotherapy. *Frontiers in immunology*, 8:1679.

[41] Elnaggar, A., Heinzinger, M., Dallago, C., Rehawi, G., Wang, Y., Jones, L., Gibbs, T., Feher, T., Angerer, C., Steinegger, M., et al. (2021). Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 44(10):7112–7127.

[42] Fang, X., Guo, Z., Liang, J., Wen, J., Liu, Y., Guan, X., and Li, H. (2022). Neoantigens and their potential applications in tumor immunotherapy. *Oncology Letters*, 23(3):1–9.

[43] Fell, M., Akhtar, S., and Basile, V. (2021). Mining annotator perspectives from hate speech corpora. In *NL4AI@ AI* IA*.

[44] Flomenberg, N., Baxter-Lowe, L. A., Confer, D., Fernandez-Vina, M., Filipovich, A., Horowitz, M., Hurley, C., Kollman, C., Anasetti, C., Noreen, H., et al. (2004). Impact of hla class i and class ii high-resolution matching on outcomes of unrelated donor bone marrow transplantation: Hla-c mismatching is associated with a strong adverse effect on transplantation outcome. *Blood*, 104(7):1923–1930.

[45] Frankle, J. and Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

[46] Frenda, S., Pedrani, A., Basile, V., Lo, S. M., Cignarella, A. T., Panizzon, R., Marco, C., Scarlini, B., Patti, V., Bosco, C., and Bernardi, D. (2023). EPIC: Multi-perspective annotation of a corpus of irony. In Rogers, A., Boyd-Graber, J., and Okazaki, N., editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13844–13857, Toronto, Canada. Association for Computational Linguistics.

[47] Gatenby, R. A. and Gillies, R. J. (2008). A microenvironmental model of carcinogenesis. *Nature Reviews Cancer*, 8(1):56–61.

[48] Generali, E., Bose, T., Selmi, C., Voncken, J. W., and Damoiseaux, J. G. (2018). Nature versus nurture in the spectrum of rheumatic diseases: Classification of spondyloarthritis as autoimmune or autoinflammatory. *Autoimmunity reviews*, 17(9):935–941.

[49] Gong, Y., Liu, L., Yang, M., and Bourdev, L. (2014). Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

[50] Gordon, M. A., Duh, K., and Andrews, N. (2020). Compressing bert: Studying the effects of weight pruning on transfer learning. *arXiv preprint arXiv:2002.08307*.

[51] Gulli, A. (2005). Ag's corpus of news articles.

[52] Gurulingappa, H., Rajput, A. M., Roberts, A., Fluck, J., Hofmann-Apitius, M., and Toldo, L. (2012). Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of Biomedical Informatics*, 45(5):885 – 892. Text Mining and Natural Language Processing in Pharmacogenomics.

[53] Han, S., Pool, J., Tran, J., and Dally, W. (2015). Learning both weights and connections for efficient neural network. *Advances in neural information processing systems*, 28.

[54] Hanahan, D. and Weinberg, R. A. (2011). Hallmarks of cancer: the next generation. *cell*, 144(5):646–674.

[55] Hanson, S. and Pratt, L. (1988). Comparing biases for minimal network construction with back-propagation. *Advances in neural information processing systems*, 1.

[56] Hassibi, B. and Stork, D. (1992). Second order derivatives for network pruning: Optimal brain surgeon. *Advances in neural information processing systems*, 5.

[57] Havens, L., Bach, B., Terras, M., and Alex, B. (2022). Beyond explanation: A case for exploratory text visualizations of non-aggregated, annotated datasets. In *Proceedings of the 1st Workshop on Perspectivist Approaches to NLP@ LREC2022*, pages 73–82.

[58] He, P., Liu, X., Gao, J., and Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

[59] He, Y., Rappuoli, R., De Groot, A. S., and Chen, R. T. (2010). Emerging vaccine informatics. *BioMed Research International*, 2010(1):218590.

[60] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[61] Hooshmand, M. N. and Maserat, E. (2024). Application of machine learning and deep learning for cancer vaccine (rapid review). *Multimedia Tools and Applications*, 83(17):51211–51226.

[62] Hou, J., Adhikari, B., and Cheng, J. (2018). Deepsf: deep convolutional neural network for mapping protein sequences to folds. *Bioinformatics*, 34(8):1295–1303.

[63] Hu, Z., Ott, P. A., and Wu, C. J. (2018). Towards personalized, tumour-specific, therapeutic vaccines for cancer. *Nature Reviews Immunology*, 18(3):168–182.

[64] Huang, G., Liu, S., Van der Maaten, L., and Weinberger, K. Q. (2018). Condensenet: An efficient densenet using learned group convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2752–2761.

[65] Institute, W. T. S. (2024). *Cancer Genome Project*. Hinxton, Cambridge CB10 1SA, UK.

[66] Janowsky, S. A. (1989). Pruning versus clipping in neural networks. *Physical Review A*, 39(12):6600.

[67] Johnson, D. R. (2000). Differential expression of human major histocompatibility class i loci: Hla-a,-b, and-c. *Human immunology*, 61(4):389–396.

[68] Jones, P., Côté, R. G., Martens, L., Quinn, A. F., Taylor, C. F., Derache, W., Hermjakob, H., and Apweiler, R. (2006). Pride: a public repository of protein and peptide identifications for the proteomics community. *Nucleic acids research*, 34(suppl_1):D659–D663.

[69] Kalaora, S., Nagler, A., Nejman, D., Alon, M., Barbolin, C., Barnea, E., Ketelaars, S. L., Cheng, K., Vervier, K., Shental, N., et al. (2021). Identification of bacteria-derived hla-bound peptides in melanoma. *Nature*, 592(7852):138–143.

[70] Kallor, A. A., Waleron, M., Bedran, G., Eugénio, P., Pesquita, C., Faria, D., Zanzotto, F. M., Battail, C., Rajan, A., and Alfaro, J. (2023). Carmen: A pan-hla and pan-cancer proteogenomic database on antigen presentation to support cancer immunotherapy. *Cancer Research*, 83(7_Supplement):6577–6577.

[71] Kaushik, A. C., Li, M., Mehmood, A., Dai, X., and Wei, D.-Q. (2021). Acps: An accurate bioinformatics tool for precision-based anti-cancer peptide generation via omics data. *Chemical Biology & Drug Design*, 97(2):372–382.

[72] Keogh, E., Fikes, J., Southwood, S., Celis, E., Chesnut, R., and Sette, A. (2001). Identification of new epitopes from four different tumor-associated antigens: recognition of naturally processed epitopes correlates with hla-a 0201-binding affinity. *The Journal of Immunology*, 167(2):787–796.

[73] Keung, P., Lu, Y., Szarvas, G., and Smith, N. A. (2020). The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

[74] Kim, Y. and Rush, A. M. (2016). Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

[75] Kisselev, A. F., Akopian, T. N., Woo, K. M., and Goldberg, A. L. (1999). The sizes of peptides generated from protein by mammalian 26 and 20 s proteasomes: implications for understanding the degradative mechanism and antigen presentation. *Journal of Biological Chemistry*, 274(6):3363–3371.

[76] Klausen, M. S., Jespersen, M. C., Nielsen, H., Jensen, K. K., Jurtz, V. I., Soenderby, C. K., Sommer, M. O. A., Winther, O., Nielsen, M., Petersen, B., et al. (2019). Netsurfp-2.0: Improved prediction of protein structural features by integrated deep learning. *Proteins: Structure, Function, and Bioinformatics*, 87(6):520–527.

[77] Kumar, N., Hendriks, B. S., Janes, K. A., de Graaf, D., and Lauffenburger, D. A. (2006). Applying computational modeling to drug discovery and development. *Drug discovery today*, 11(17-18):806–811.

[78] Lagunas, F., Charlaix, E., Sanh, V., and Rush, A. M. (2021). Block pruning for faster transformers. *arXiv preprint arXiv:2109.04838*.

[79] LeCun, Y., Denker, J., and Solla, S. (1989). Optimal brain damage. *Advances in neural information processing systems*, 2.

[80] Lemoine, F. M., Cherai, M., Giverne, C., Dimitri, D., Rosenzwajg, M., Trebeden-Negre, H., Chaput, N., Barrou, B., Thioun, N., Gattegnio, B., et al. (2009). Massive expansion of regulatory t-cells following interleukin 2 treatment during a phase i-ii dendritic cell-based immunotherapy of metastatic renal cancer. *International journal of oncology*, 35(3):569–581.

[81] Li, X. and Roth, D. (2002). Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

[82] Lin, Z., Akin, H., Rao, R., Hie, B., Zhu, Z., Lu, W., Smetanin, N., Verkuil, R., Kabeli, O., Shmueli, Y., et al. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130.

[83] Loeb, L. A. and Harris, C. C. (2008). Advances in chemical carcinogenesis: a historical review and prospective. *Cancer research*, 68(17):6863.

[84] Loshchilov, I. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

[85] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., and Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

[86] Machaca, V., Goyzueta, V., Cruz, M. G., Sejje, E., Pilco, L. M., López, J., and Túpac, Y. (2024). Transformers meets neoantigen detection: a systematic literature review. *Journal of Integrative Bioinformatics*, 21(2):20230043.

[87] Malach, E., Yehudai, G., Shalev-Schwartz, S., and Shamir, O. (2020). Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pages 6682–6691. PMLR.

[88] Manning, C. D., Clark, K., Hewitt, J., Khandelwal, U., and Levy, O. (2020). Emergent linguistic structure in artificial neural networks trained by self-supervision. *Proceedings of the National Academy of Sciences*, 117(48):30046–30054.

[89] Martens, L., Hermjakob, H., Jones, P., Adamski, M., Taylor, C., States, D., Gevaert, K., Vandekerckhove, J., and Apweiler, R. (2005a). Pride: the proteomics identifications database. *Proteomics*, 5(13):3537–3545.

[90] Martens, L., Van Damme, P., Van Damme, J., Staes, A., Timmerman, E., Ghesquière, B., Thomas, G. R., Vandekerckhove, J., and Gevaert, K. (2005b). The human platelet proteome mapped by peptide-centric proteomics: a functional protein profile. *Proteomics*, 5(12):3193–3204.

[91] Mason, P. and Parham, P. (1998). Hla class i region sequences, 1998. *Tissue antigens*, 51(4):417–466.

[92] Mastromattei, M., Basile, V., Zanzotto, F. M., et al. (2022a). Change my mind: how syntax-based hate speech recognizer can uncover hidden motivations based on different viewpoints. In *1st Workshop on Perspectivist Approaches to Disagreement in NLP, NLPerspectives 2022 as part of Language Resources and Evaluation Conference, LREC 2022 Workshop*, pages 117–125. European Language Resources Association (ELRA).

[93] Mastromattei, M., Ranaldi, L., Fallucchi, F., and Zanzotto, F. M. (2022b). Syntax and prejudice: Ethically-charged biases of a syntax-based hate speech recognizer unveiled. *PeerJ Computer Science*, 8:e859.

[94] Mastromattei, M. and Zanzotto, F. M. (2024a). Less is KEN: a universal and simple non-parametric pruning algorithm for large language models. In Ku, L.-W., Martins, A., and Srikumar, V., editors, *Findings of the Association for Computational Linguistics ACL 2024*, pages 11361–11374, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

[95] Mastromattei, M. and Zanzotto, F. M. (2024b). Linguistic fingerprint in transformer models: How language variation influences parameter selection in irony detection. In Abercrombie, G., Basile, V., Bernadi, D., Dudy, S., Frenda, S., Havens, L., and Tonelli, S., editors, *Proceedings of the 3rd Workshop on Perspectivist Approaches to NLP (NLPerspectives) @ LREC-COLING 2024*, pages 123–130, Torino, Italia. ELRA and ICCL.

[96] McCandless, D., Evans, T., and Barton, P. (2024). The rise and rise of a.i. large language models (llms) and their associated bots like chatgpt. *Information is Beautiful*.

[97] McCarty, M. and Avery, O. T. (1946). Studies on the chemical nature of the substance inducing transformation of pneumococcal types. *Journal of Experimental Medicine*, 83(2):97–104.

[98] Michele, M., Basile, V., Zanzotto, F. M., et al. (2022). Change my mind: How syntax-based hate speech recognizer can uncover hidden motivations based on different viewpoints. In *1st Workshop on Perspectivist Approaches to Disagreement in NLP, NLPerspectives 2022 as part of Language Resources and Evaluation Conference, LREC 2022 Workshop*, pages 117–125. European Language Resources Association (ELRA).

[99] Mieleszczenko-Kowszewicz, W., Kanclerz, K., Bielaniewicz, J., Oleksy, M., Gruza, M., Wozniak, S., Dziecioł, E., Kazienko, P., and Kocon, J. (2023). Capturing human perspectives in nlp: Questionnaires, annotations, and biases. In *The ECAI 2023 2nd Workshop on Perspectivist Approaches to NLP. CEUR Workshop Proceedings*.

[100] Miłkowski, P., Gruza, M., Kanclerz, K., Kazienko, P., Grimling, D., and Kocoń, J. (2021). Personal bias in prediction of emotions elicited by textual opinions. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 248–259.

[101] Mistry, J., Chuguransky, S., Williams, L., Qureshi, M., Salazar, G. A., Sonnhammer, E. L., Tosatto, S. C., Paladin, L., Raj, S., Richardson, L. J., et al. (2021). Pfam: The protein families database in 2021. *Nucleic acids research*, 49(D1):D412–D419.

[102] Moult, J., Fidelis, K., Kryshtafovych, A., Schwede, T., and Tramontano, A. (2018). Critical assessment of methods of protein structure prediction (casp)—round xii. *Proteins: Structure, Function, and Bioinformatics*, 86:7–15.

[103] Mozer, M. C. and Smolensky, P. (1989). Using relevance to reduce network size automatically. *Connection Science*, 1(1):3–16.

[104] Mukherjee, S. (2010). *The emperor of all maladies: a biography of cancer*. Simon and Schuster.

[105] Müller, M., Gfeller, D., Coukos, G., and Bassani-Sternberg, M. (2017). 'hotspots' of antigen presentation revealed by human leukocyte antigen ligandomics for neoantigen prioritization. *Frontiers in immunology*, 8:1367.

[106] Nesta, A. V., Tafur, D., and Beck, C. R. (2021). Hotspots of human mutation. *Trends in Genetics*, 37(8):717–729.

[107] Nielsen, N. P., Salgado, S., and Berntsen, D. (2020). Using virtual reality to examine emotional hotspots and intrusions in the trauma film paradigm. *Journal of Applied Research in Memory and Cognition*, 9(3):370–380.

[108] Nukaya, I., Yasumoto, M., Iwasaki, T., Ideno, M., Sette, A., Celis, E., Takesako, K., and Kato, I. (1999). Identification of hla-a24 epitope peptides of carcinoembryonic antigen which induce tumor-reactive cytotoxic t lymphocyte. *International journal of cancer*, 80(1):92–97.

[109] Omenn, G. S. (2004). The human proteome organization plasma proteome project pilot phase: reference specimens, technology platform comparisons, and standardized data submissions and analyses. *Proteomics*, 4(5):1235–1240.

[110] Ouyang, X., Liu, Y., Zhou, Y., Guo, J., Wei, T.-T., Liu, C., Lee, B., Chen, B., Zhang, A., Casey, K. M., et al. (2021). Antitumor effects of ipsc-based cancer vaccine in pancreatic cancer. *Stem cell reports*, 16(6):1468–1477.

[111] O'Connor, C. M., Adams, J. U., and Fairman, J. (2010). Essentials of cell biology. *Cambridge, MA: NPG Education*, 1(54):5.

[112] O'Donnell, T. J., Rubinsteyn, A., and Laserson, U. (2020). Mhcflurry 2.0: improved pan-allele prediction of mhc class i-presented peptides by incorporating antigen processing. *Cell systems*, 11(1):42–48.

[113] Pang, B. and Lee, L. (2005). Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. *arXiv preprint cs/0506075*.

[114] Paola Cruz-Tapias, J. and Anaya, J.-M. (2013). Major histocompatibility complex: Antigen processing and presentation. *Autoimmunity: From Bench to Bedside; NIH: Bethesda, MD, USA*.

[115] Pappalardo, F., Chiacchio, F., and Motta, S. (2013). Cancer vaccines: state of the art of the computational modeling approaches. *BioMed research international*, 2013(1):106407.

[116] Peng, M., Mo, Y., Wang, Y., Wu, P., Zhang, Y., Xiong, F., Guo, C., Wu, X., Li, Y., Li, X., et al. (2019). Neoantigen vaccine: an emerging tumor immunotherapy. *Molecular cancer*, 18:1–14.

[117] Piazuelo, M. B., Epplein, M., and Correa, P. (2010). Gastric cancer: an infectious disease. *Infectious Disease Clinics*, 24(4):853–869.

[118] Prince, J. T., Carlson, M. W., Wang, R., Lu, P., and Marcotte, E. M. (2004). The need for a public proteomics repository. *Nature biotechnology*, 22(4):471–472.

[119] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

[120] Rao, R., Bhattacharya, N., Thomas, N., Duan, Y., Chen, P., Canny, J., Abbeel, P., and Song, Y. (2019). Evaluating protein transfer learning with tape. *Advances in neural information processing systems*, 32.

[121] Reits, E. A., Vos, J. C., Gromme, M., and Neefjes, J. (2000). The major substrates for tap in vivo are derived from newly synthesized proteins. *Nature*, 404(6779):774–778.

[122] Reynisson, B., Alvarez, B., Paul, S., Peters, B., and Nielsen, M. (2020). Netmhcpan-4.1 and netmhciipan-4.0: improved predictions of mhc antigen presentation by concurrent

motif deconvolution and integration of ms mhc eluted ligand data. *Nucleic acids research*, 48(W1):W449–W454.

[123] Rives, A., Meier, J., Sercu, T., Goyal, S., Lin, Z., Liu, J., Guo, D., Ott, M., Zitnick, C. L., Ma, J., et al. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118.

[124] Robinson, J., Halliwell, J. A., Hayhurst, J. D., Flicek, P., Parham, P., and Marsh, S. G. (2015). The ipd and imgt/hla database: allele variant databases. *Nucleic acids research*, 43(D1):D423–D431.

[125] Rocklin, G. J., Chidyausiku, T. M., Goreshnik, I., Ford, A., Houliston, S., Lemak, A., Carter, L., Ravichandran, R., Mulligan, V. K., Chevalier, A., et al. (2017). Global analysis of protein folding using massively parallel design, synthesis, and testing. *Science*, 357(6347):168–175.

[126] Rogers, A., Kovaleva, O., and Rumshisky, A. (2020). A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

[127] Ruzzetti, E. S., Ranaldi, F., Logozzo, F., Mastromattei, M., Ranaldi, L., and Zanzotto, F. M. (2023). Exploring linguistic properties of monolingual BERTs with typological classification among languages. In Bouamor, H., Pino, J., and Bali, K., editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14447–14461, Singapore. Association for Computational Linguistics.

[128] Ruzzetti, E. S., Ranaldi, L., Mastromattei, M., Fallucchi, F., Scarpato, N., and Zanzotto, F. M. (2022). Lacking the embedding of a word? look it up into a traditional dictionary. In Muresan, S., Nakov, P., and Villavicencio, A., editors, *Findings of the As-*

sociation for Computational Linguistics: ACL 2022, pages 2651–2662, Dublin, Ireland. Association for Computational Linguistics.

[129] Sahin, U. and Türeci, Ö. (2018). Personalized vaccines for cancer immunotherapy. *Science*, 359(6382):1355–1360.

[130] Sajjad, H., Dalvi, F., Durrani, N., and Nakov, P. (2020). Poor man's bert: Smaller and faster transformer models. *arXiv preprint arXiv:2004.03844*, 2(2).

[131] Samek, W. and Müller, K.-R. (2019). Towards explainable artificial intelligence. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 5–22.

[132] Samek, W., Wiegand, T., and Müller, K.-R. (2017). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296*.

[133] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

[134] Sarah, W. (2023). What are neoantigens? *Cancer Research*.

[135] Sarkisyan, K. S., Bolotin, D. A., Meer, M. V., Usmanova, D. R., Mishin, A. S., Sharonov, G. V., Ivankov, D. N., Bozhanova, N. G., Baranov, M. S., Soylemez, O., et al. (2016). Local fitness landscape of the green fluorescent protein. *Nature*, 533(7603):397–401.

[136] Schrader, A., Varga, Z., Hegele, A., Pfoertner, S., Olbert, P., and Hofmann, R. (2006). Second-line strategies for metastatic renal cell carcinoma: classics and novel approaches. *Journal of cancer research and clinical oncology*, 132:137–149.

[137] Schubert, U., Anton, L. C., Gibbs, J., Norbury, C. C., Yewdell, J. W., and Bennink, J. R. (2000). Rapid degradation of a large fraction of newly synthesized proteins by proteasomes. *Nature*, 404(6779):770–774.

[138] Schumacher, T. N. and Schreiber, R. D. (2015). Neoantigens in cancer immunotherapy. *Science*, 348(6230):69–74.

[139] Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons.

[140] Shen, H., Shao, H.-W., Chen, X.-H., Wu, F.-L., Wang, H., Huang, Z.-L., Shen, J., Wang, T., Zhang, W.-F., and Huang, S.-L. (2013). Identification of a novel hla-a2-restricted mutated survivin epitope and induction of specific anti-hcc ctls that could effectively cross-recognize wild-type survivin antigen. *Cancer Immunology, Immunotherapy*, 62:393–403.

[141] Sheng, E. and Uthus, D. (2020). Investigating societal biases in a poetry composition system. In *Proceedings of the Second Workshop on Gender Bias in Natural Language Processing*, pages 93–106, Barcelona, Spain (Online). Association for Computational Linguistics.

[142] Singh, S. P. and Alistarh, D. (2020). Woodfisher: Efficient second-order approximation for neural network compression. *Advances in Neural Information Processing Systems*, 33:18098–18109.

[143] Smith, C. C., Selitsky, S. R., Chai, S., Armistead, P. M., Vincent, B. G., and Serody, J. S. (2019). Alternative tumour-specific antigens. *Nature Reviews Cancer*, 19(8):465–478.

[144] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

[145] Spiliotis, E. T., Osorio, M., Zúñiga, M. C., and Edidin, M. (2000). Selective export of mhc class i molecules from the er after their dissociation from tap. *Immunity*, 13(6):841–851.

[146] Stratton, M. R., Campbell, P. J., and Futreal, P. A. (2009). The cancer genome. *Nature*, 458(7239):719–724.

[147] Sun, X., Zhou, X., Wang, Q., and Sharples, S. (2022). Investigating the impact of emotions on perceiving serendipitous information encountering. *Journal of the Association for Information Science and Technology*, 73(1):3–18.

[148] Sun, Y., Wang, S., Li, Y., Feng, S., Tian, H., Wu, H., and Wang, H. (2020). Ernie 2.0: A continual pre-training framework for language understanding. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8968–8975.

[149] Sze, V., Chen, Y.-H., Yang, T.-J., and Emer, J. S. (2017). Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE*, 105(12):2295–2329.

[150] Tang, M.-s., Lee, H.-W., Weng, M.-w., Wang, H.-T., Hu, Y., Chen, L.-C., Park, S.-H., Chan, H.-w., Xu, J., Wu, X.-R., et al. (2022). Dna damage, dna repair and carcinogenicity: Tobacco smoke versus electronic cigarette aerosol. *Mutation Research/Reviews in Mutation Research*, 789:108409.

[Technology] Technology, G. C. Attention mechanism: Overview. 2023.

[152] Tourimpampa, A., Drigas, A., Economou, A., and Roussos, P. (2018). Perception and text comprehension. it'sa matter of perception! *International Journal of Emerging Technologies in Learning (Online)*, 13(7):228.

[153] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. (2023). Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

[154] van Endert, P. M., Saveanu, L., Hewitt, E. W., and Lehner, P. J. (2002). Powering the peptide pump: Tap crosstalk with energetic nucleotides. *Trends in biochemical sciences*, 27(9):454–461.

[155] Vaswani, A. (2017). Attention is all you need. *arXiv preprint arXiv:1706.03762*.

[156] Vilone, G. and Longo, L. (2021). Notions of explainability and evaluation approaches for explainable artificial intelligence. *Information Fusion*, 76:89–106.

[157] Von Hansemann, D. (1890). Ueber asymmetrische zelltheilung in epithelkrebsen und deren biologische bedeutung. *Virchow's Arch. Path. Anat*, 119:299.

[158] Walter, S., Weinschenk, T., Stenzl, A., Zdrojowy, R., Pluzanska, A., Szczylik, C., Staehler, M., Brugger, W., Dietrich, P.-Y., Mendrzyk, R., et al. (2012). Multipeptide immune response to cancer vaccine ima901 after single-dose cyclophosphamide associates with longer patient survival. *Nature medicine*, 18(8):1254–1261.

[159] Wang, E. and Aifantis, I. (2020). Rna splicing and cancer. *Trends in cancer*, 6(8):631–644.

[160] Wang, Z. and Cao, Y. J. (2020). Adoptive cell therapy targeting neoantigens: a frontier for cancer research. *Frontiers in immunology*, 11:176.

[161] Wang, Z., Wohlwend, J., and Lei, T. (2019). Structured pruning of large language models. *arXiv preprint arXiv:1910.04732*.

[162] Watson, J. D. and Crick, F. H. (1953). Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. *Nature*, 171(4356):737–738.

[163] Williams, A. P., Peh, C. A., Purcell, A. W., McCluskey, J., and Elliott, T. (2002). Optimization of the mhc class i peptide cargo is dependent on tapasin. *Immunity*, 16(4):509–520.

[164] Workshop, B., Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., et al. (2022). Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

[165] Xie, N., Shen, G., Gao, W., Huang, Z., Huang, C., and Fu, L. (2023). Neoantigens: promising targets for cancer therapy. *Signal transduction and targeted therapy*, 8(1):9.

[166] Xompero, G., Mastromattei, M., Salman, S., Giannone, C., Favalli, A., Romagnoli, R., and Zanzotto, F. M. (2022). Every time I fire a conversational designer, the performance of the dialogue system goes down. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 137–145, Marseille, France. European Language Resources Association.

[167] Yamanluirt, G., Berns, E. J., Xue, A., Lee, A., Bagheri, N., Mrksich, M., and Mirkin, C. A. (2020). Exploration of the nanomedicine-design space with high-throughput screening and machine learning. In *Spherical Nucleic Acids*, pages 1687–1716. Jenny Stanford Publishing.

[168] Yang, T.-J., Chen, Y.-H., and Sze, V. (2017). Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5687–5695.

[169] Yoshimura, K., Minami, T., Nozawa, M., and Uemura, H. (2013). Phase i clinical trial of human vascular endothelial growth factor receptor 1 peptide vaccines for patients with metastatic renal cell carcinoma. *British journal of cancer*, 108(6):1260–1266.

[170] Yu, F., Koltun, V., and Funkhouser, T. (2017). Dilated residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 472–480.

[171] Zaidi, N., Soban, M., Chen, F., Kinkead, H., Mathew, J., Yarchoan, M., Armstrong, T. D., Haider, S., and Jaffee, E. M. (2020). Role of in silico structural modeling in predicting immunogenic neoepitopes for cancer vaccine development. *JCI insight*, 5(17).

[172] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

[173] Zhang, X.-M., Huang, Y., Li, Z.-S., Lin, H., and Sui, Y.-F. (2010). Prediction and analysis of hla-a2/a24-restricted cytotoxic t-lymphocyte epitopes of the tumor antigen mage-n using the artificial neural networks method on netctl1. 2 server. *Oncology Letters*, 1(6):1097–1100.

[174] Zhao, A. and Yu, Y. (2021). Knowledge-enabled bert for aspect-based sentiment analysis. *Knowledge-Based Systems*, 227:107220.

[175] Zhao, W. and Sher, X. (2018). Systematically benchmarking peptide-mhc binding predictors: From synthetic to naturally processed epitopes. *PLoS computational biology*, 14(11):e1006457.

[176] Zhu, C., Han, S., Mao, H., and Dally, W. J. (2016). Trained ternary quantization. *arXiv preprint arXiv:1612.01064*.

[177] Zhu, M. and Gupta, S. (2017). To prune, or not to prune: exploring the efficacy of pruning for model compression. *arXiv preprint arXiv:1710.01878*.

[178] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.

Appendices

Appendix A

CARMEN dataset composition

This chapter delves deeper into the CARMEN dataset, initially introduced in Sec. 3.1. Specifically, all analyses presented are based on the `main` dataset (Sec. 3.1, Tab. 3.1), which was used to create the classification corpora (Sec. 4.1.2) and for all its applications.

The following sections provide an in-depth exploration of key aspects of the CARMEN dataset, such as the distribution of peptide lengths (Apx. A.0.1), spectral count analysis (Apx. A.0.2), and the prevalence of HLA alleles (Apx. A.0.3).

A.0.1 Peptide lenght

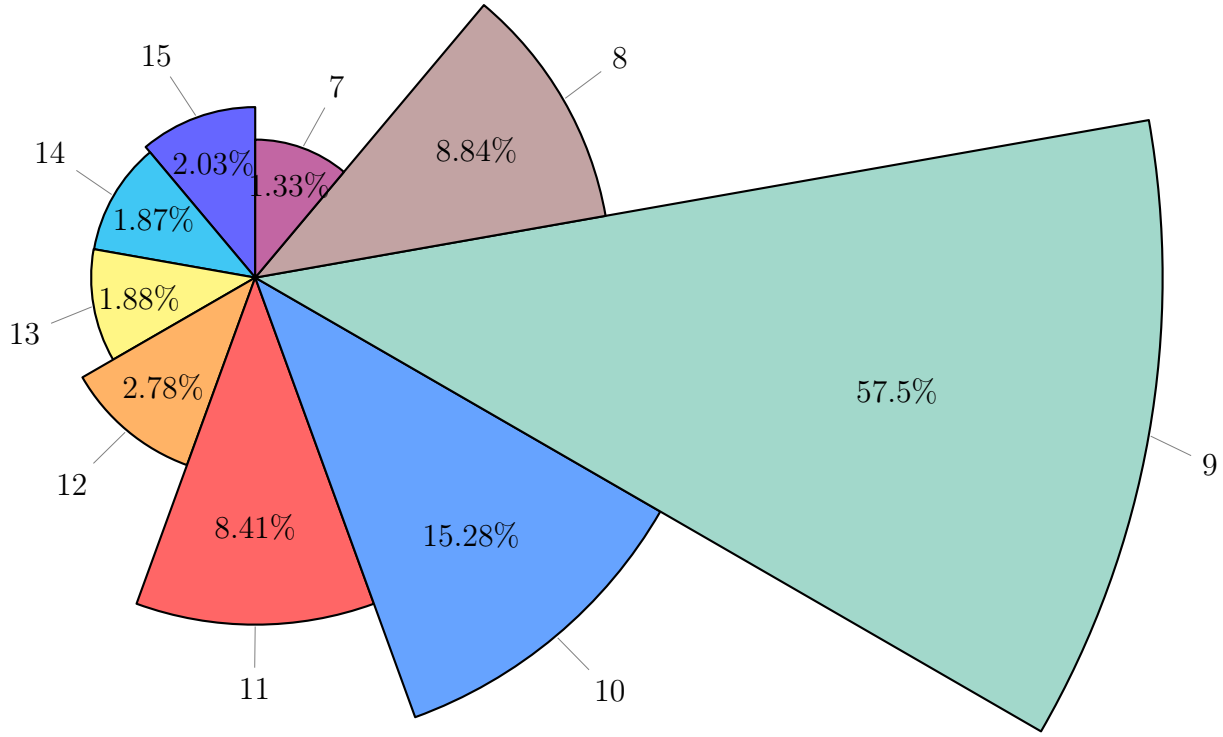


Figure A.1: Peptide length within CARMEN dataset

It is crucial to understand the sizes of peptides formed by proteasomes during protein breakdown to comprehend their degradative processes and the subsequent stages in protein turnover and the creation of major histocompatibility complex class I antigenic peptides [75].

Upon examining the 7,896,364 peptides in the primary CARMEN dataset, we found that all the peptides fall within the 7 to 25 range, in accordance with Kisselev et al. [75]. The average length distribution is 9.58 with a standard deviation (δ) of 1.44. This means that most of the peptides are within the range [8, 11], which accounts for more than half of the entire dataset, as depicted in Fig. A.1. Additionally, the range [20, 25] is less concentrated, comprising fewer than 100 examples in total.

A.0.2 Spectral counts

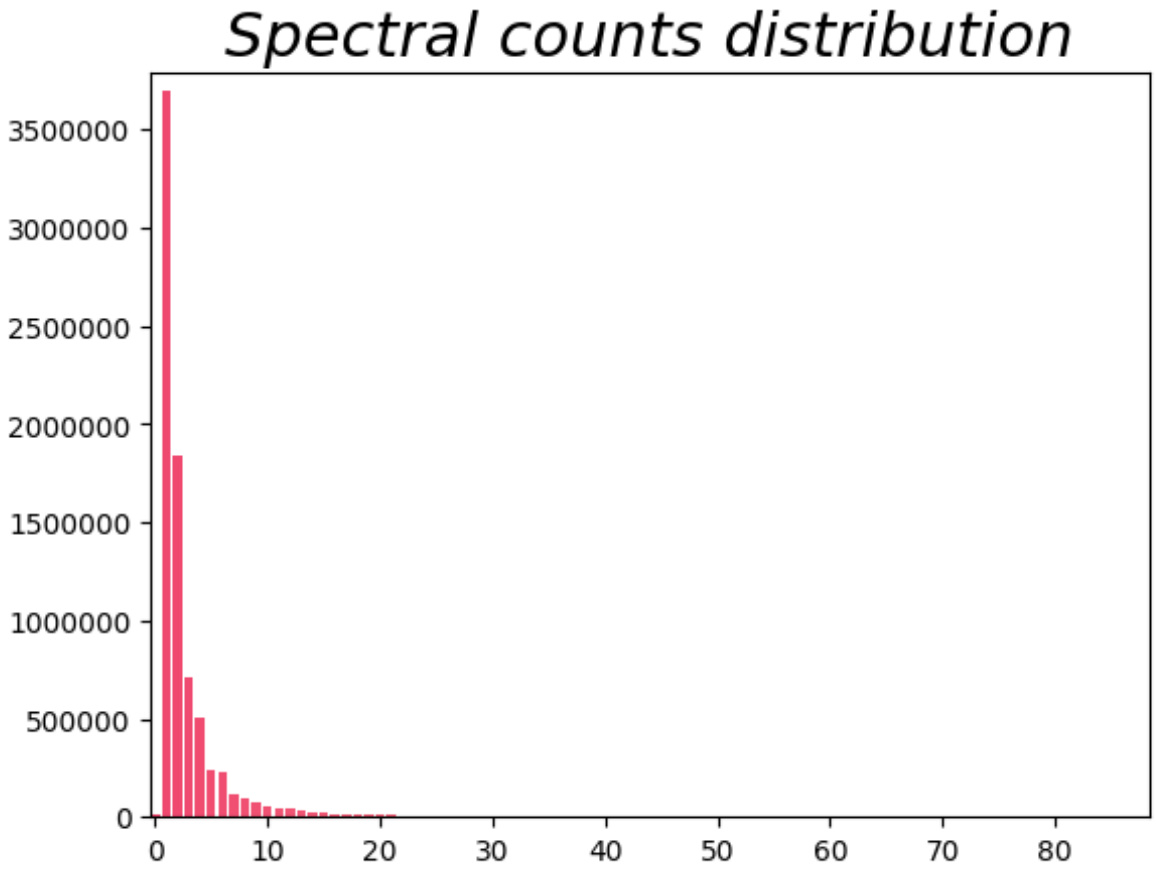


Figure A.2: Spectral counts distribution

Spectral count in proteomics refers to the frequency with which a specific peptide or protein is identified in a mass spectrometry (MS) experiment. This method is simple yet effective for label-free quantification of proteins. The steps to count how many spectral are in a peptide are the following:

1. **Protein Digestion**: Proteins in a sample are broken down into smaller peptides using enzymes like trypsin.

2. **Liquid Chromatography (LC)**: Peptides are separated based on their size and charge using LC.

3. **Mass Spectrometry (MS)**: Peptides are ionized and analyzed based on their mass-to-charge ratio.

4. **Spectral Identification**: The MS data is searched against a protein database to identify the peptides and their corresponding proteins.

5. **Spectral Counting**: For each identified protein, the number of times its peptides appear in the MS spectra is counted, which is known as the spectral count.

In the dataset, we found that the spectral counts range from 0 to 1300, with an average of 2.99 and a standard deviation (δ) of 5.70. In detail, Fig. A.2 illustrates that, despite the wide range, the spectral counts are mostly concentrated at 0 (no spectral counts), 1, and 2.

A.0.3 HLA distribution

A.0.3.1 HLA-A

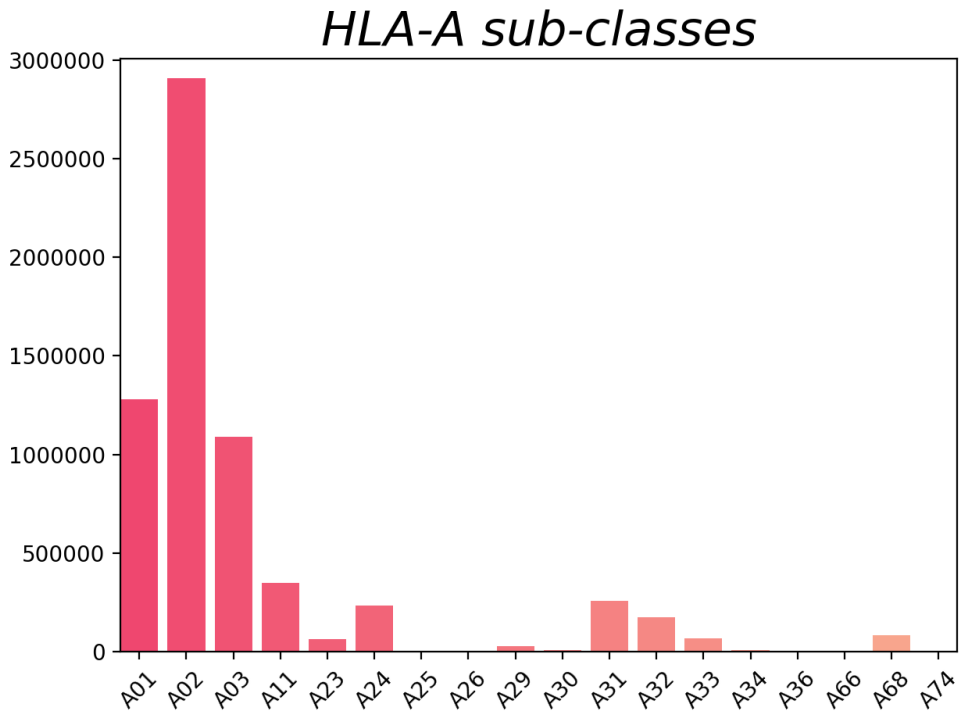


Figure A.3: HLA-A sub-classes distribution

As noted in Sec. 3.1.2 and Fig. 3.2, HLA-A alleles dominate the dataset, accounting for 83% of all peptides. HLA-C and HLA-B follow with 1.54% and 1.45%, respectively, excluding peptides not binding by NetMHCpan (13.69%).

Here, we delve deeper into the composition of each HLA class and the distribution of their respective subclasses.

Within the HLA-A subset, HLA-A:02 is the most prevalent allele, representing 44.26% of the sub-dataset with 2,907,292 binding peptides. HLA-A:01 and HLA-A:03 follow with 19.46% and 16.61%, respectively.

In contrast, the least common HLA-A subclasses are HLA-A:74 (0.06%), HLA-A:25 (0.05%), HLA-A:36 (0.05%), HLA-A:66 (0.04%), and HLA-A:26 (0.04%).

A.0.3.2 HLA-B

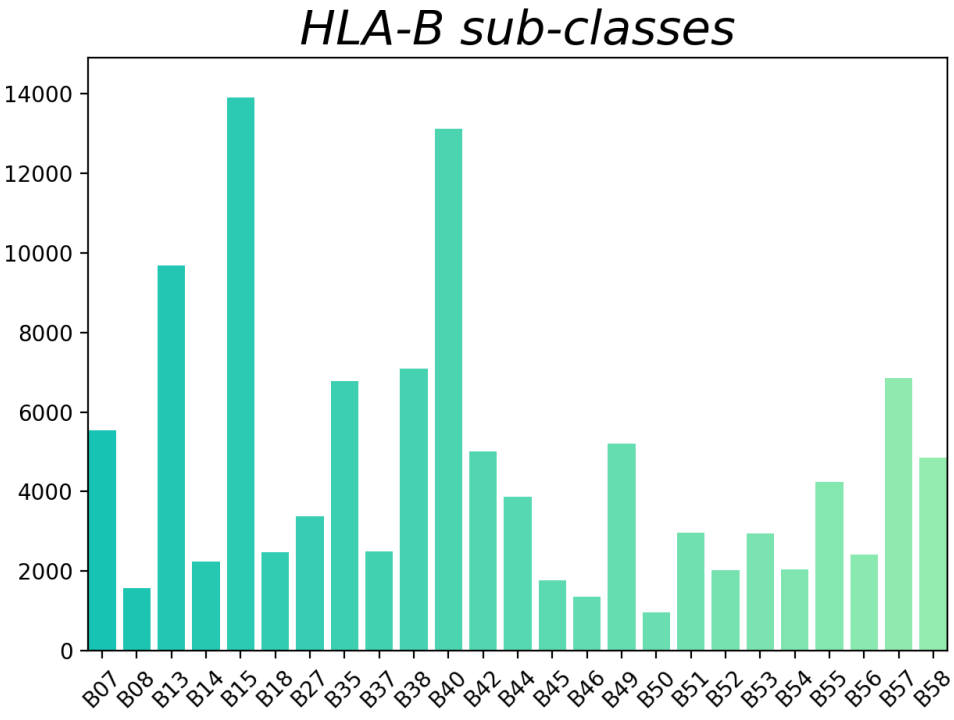


Figure A.4: HLA-B sub-classes distribution

The HLA-B subset has the highest number of different alleles. HLA-B:15 is the most prevalent, representing 12.12% of the sub-dataset with 13,918 binding peptides, followed

by HLA-B:40 (11.42%) and HLA-B:13 (8.44%). The distribution of sub-classes is generally harmonic, with the least common sub-classes close to 1% of the entire peptide sub-class. Specifically, HLA-B:50 is the least common allele, with 954 peptides (0.83%), followed by HLA-B:46 (1.18%) and HLA-B:08 (1.38%).

A.0.3.3 HLA-C

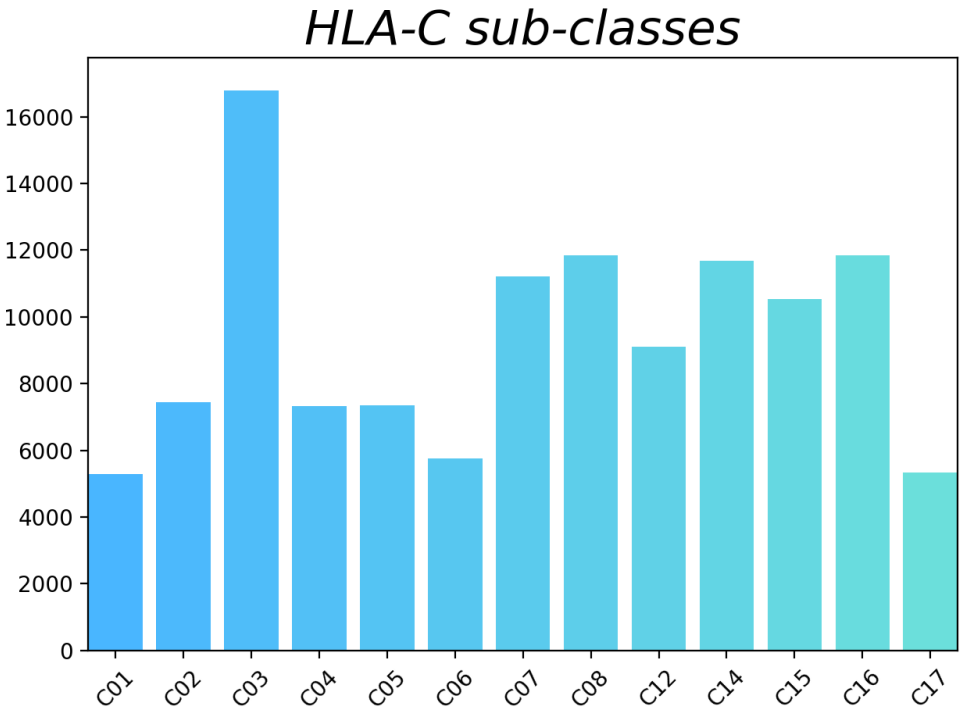


Figure A.5: HLA-C sub-classes distribution

The HLA-C subset is the most well-balanced dataset and has the fewest subclasses (13). This means that all of them have a good frequency in the dataset, with HLA-C:03, HLA-C:16, and HLA-C:08 being the most common ones (16776 - 13.81%, 11840 - 9.75% and 11835 - 9.74%). This is also reflected to the lowest subclasses that have more than 4% of alleles in the dataset: HLA-C:01 (5284, 4.35%), HLA-C:17 (5325, 4.38%) and HLA-C:06 (5759, 4.74%).

Appendix B

KEN pruning algorithm additional experiments

In this Appendix, we present supplementary findings related to the KEN pruning algorithm as introduced in [94]. Specifically, Apx.B.1 demonstrates the superiority of KEN's KDE-based parameter selection over random choices. Apx. B.2 provides additional results using the same methodologies described in Sec. 3.3.3. Finally, Apx. B.3. showcases new examples of the KEN visualization tool (KEN_{viz}), illustrating how the key attention matrix evolves across layers 0 and 12 of a BERT model trained on the `glue-sst2` dataset. For each layer, we present both *single matrix view* and *neighbor count view*.

B.1 How to prove the importance of selected parameters

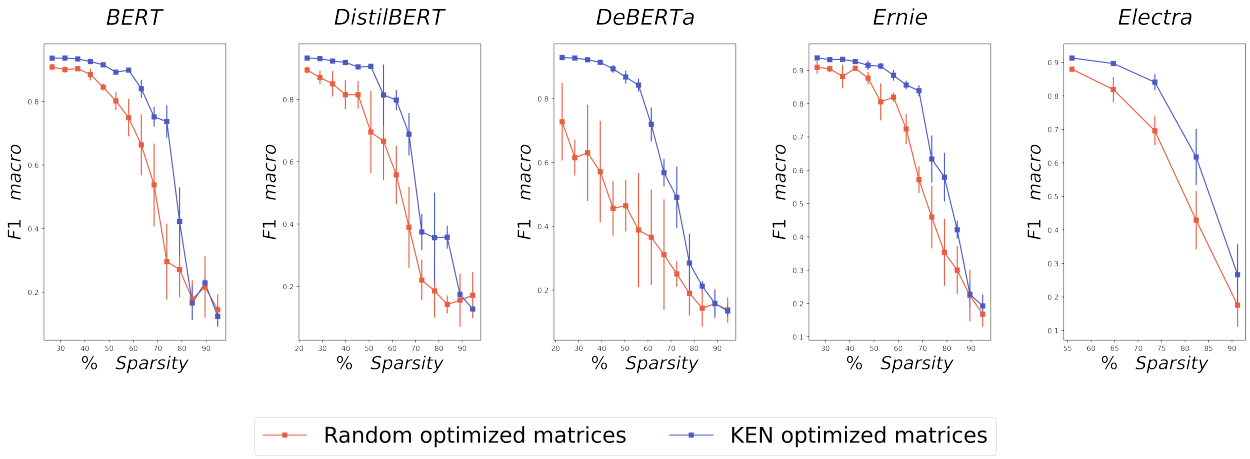


Figure B.1: Performance variation on `AG-NEWS` dataset with different reset parameters percentage value. All the experiments were conducted using KEN *full* configuration

To assess the effectiveness of KEN core idea, which involves selecting parameters based on their distribution using Kernel Density Estimation (KDE), we conducted parallel experiments. In these experiments, KEN randomly chose parameters to either retain or reset to their pre-trained values. This randomized approach allowed us to compare KEN KDE-based selection strategy against random parameter pruning.

Formally, for each matrix in a generic model, the optimized matrix $\hat{W}$ contained k randomly selected fine-tuned parameters. Our goal is to determine whether the parameters introduced into a generic transformer model by KEN constituted an optimal subnetwork or if equivalent results could be achieved by randomly selecting the same number of parameters. To address this question, we performed an experiment using the `AG-NEWS` dataset, comparing the performance differences between extracting $\hat{W}$ matrices using KEN and using k random values for each matrix row.

The results, illustrated in Fig. B.1, consistently show that KEN outperforms its random counterpart. KEN achieves a lower error rate and a smaller performance gap at reasonable compression levels. It is important to note that, in all cases and for all models examined,

Dataset	Reference
trec	Li and Roth [81]
AG-NEWS	Gulli [51]
rotten tomatoes	Pang and Lee [113]
IMDB	Maas et al. [85]
ade_corpus_v2	Gurulingappa et al. [52]
glue-sst2	Socher et al. [144]
YELP POLARITY	Zhang et al. [172]
hate_speech_offensive	Davidson et al. [34]
hate_speech18	de Gibert et al. [35]
EMO	Chatterjee et al. [23]
scicite	Cohan et al. [29]
amazon_reviews_multi	Keung et al. [73]
poem sentiment	Sheng and Uthus [141]
tweet_eval-emoji	Barbieri et al. [8]
tweet_eval-hate	Barbieri et al. [8]
tweet_eval-irony	Barbieri et al. [8]
tweet_eval-offensive	Barbieri et al. [8]
tweet_eval-feminist	Barbieri et al. [8]

Table B.1: Analyzed datasets

there exists a threshold value beyond which the model performance inevitably declines. The KEN algorithm effectively compresses models while preserving high performance and minimizing error rates. However, if the reset parameters exceed a certain threshold (specific to each model), its performance suffers a catastrophic decline. When random values are used, this threshold is reached earlier, resulting in a larger performance gap and higher error rate. Nevertheless, the upper limit obtained with random selection is always lower than or equal to the average value obtained with KEN.

Furthermore, when using KEN, the error rate remains minimal within the threshold. This suggests that the subnetwork derived from KEN is not random; rather, it consistently selects the most effective portion of the original network.

Dataset	$BLOOM_{1B7}$	$BLOOM_{560k}$	Bert	DistilBert	DeBERTa	Ernie	Electra
trec	61.46%	23.26%	26.55%	23.45%	22.84%	26.55%	55.94%
rotten_tomatoes	69.17%	24.80%	26.55%	34.39%	44.88%	42.29%	55.94%
hate_speech_offensive	61.46%	23.26%	26.55%	34.39%	22.84%	26.55%	55.94%
hate_speech18	61.46%	23.26%	26.55%	23.45%	33.86%	31.80%	64.75%
scicite	61.46%	23.26%	37.05%	28.92%	22.84%	31.80%	55.94%†
ade_corpus_v2	69.17%	24.80%	52.78%	45.32%	44.88%	63.28%	73.56%
amazon_reviews_multi	69.17%	24.80%	31.80%	34.39%	22.84%	31.80%	55.94%†
poem_sentiment	74.31%	26.34%	58.03%	45.32%	22.84%	47.54%	73.56%
tweet_eval-emoji	74.31%	23.26%	63.28%	23.45%	44.88%	79.02%	55.94%
tweet_eval-hate	61.46%	23.26%	26.55%	61.73%	44.88%	47.54%	55.94%
tweet_eval-irony	61.46%	23.26%	26.55%	23.45%	22.84%	26.55%	64.75%
tweet_eval-offensive	61.46%	23.26%	26.55%†	34.39%	28.35%	31.80%	55.94%
tweet_eval-femminist	61.46%	23.26%	26.55%	39.05%	22.84%	37.05%	64.75%

Table B.2: Results obtained from the analysis of additional datasets not shown in Tab.3.6. The values presented in this table correspond to the lowest percentage of reset parameters that KEN achieved without impacting the model performance. The † symbol denotes a reset parameter rate that falls below the minimum value reported in Tab. 3.6

B.2 Additional results

In this appendix, we show additional results obtained using KEN not shown in Tab. 3.6. Tab. B.1 provides a comprehensive overview of all datasets analyzed in the paper, while Tab. B.2 displays the additional results included. Unlike Tab. 3.6, Tab. B.2 focuses on the highest percentage of reset parameters for each model on each dataset where KEN F1-weighted score matches or surpasses the performance of the original unpruned model. This highlights the exceptional compression capabilities of KEN, enabling it to achieve comparable or even improved performance while significantly reducing the model parameter count.

B.3 KEN$_{viz}$ examples

KEN$_{viz}$ generates visual representations of the model pruning after the KEN application. Here, we focus on key matrices from layers 0 and 12 of a BERT model trained on the `glue-sst2` dataset (details in Sec. 3.3.3.1). For each layer, we present both a single matrix view and a neighbor count view, as described in Sec. 3.3.3.3.

BERT was chosen for this experiment due to its exceptional performance across a range of k values during testing (Tab. 3.6 and Tab. B.2). To comprehensively explore how parameter selection patterns evolve, we employed three different k values, representing varying degrees of parameter selection. This allowed us to observe how parameter choices shift as the amount of parameter resetting increases.

Fig. B.2 and Fig. B.3 consistently reveal a uniform distribution of parameters within each matrix row across all configurations and layers. This implies an absence of well-defined clusters of selected parameters. Furthermore, the number of neighbors for each parameter remains consistent regardless of the chosen k value.

(a) Parameter reset 21.87%

(b) Parameter reset 47.91%

(c) Parameter reset 73.95%

Figure B.2: KEN$_{viz}$ visualization of the key attention matrix at layer 0 of a BERT model trained on the `glue-sst2` dataset. The left-hand figures depict the matrix after undergoing the KEN pruning stage, while the right-hand ones showcase the corresponding neighbor counts

(a) Parameter reset 21.87%

(b) Parameter reset 47.91%

(c) Parameter reset 73.95%

Figure B.3: KEN$_{viz}$ visualization of the key attention matrix at layer 12 of a BERT model trained on the `glue-sst2` dataset. The left-hand figures depict the matrix after undergoing the KEN pruning stage, while the right-hand ones showcase the corresponding neighbor counts

Appendix C

Linguistic Fingerprint in Transformer Models

Mastromattei and Zanzotto [95] explore the correlation between linguistic diversity, sentiment analysis, and transformer model architectures. The authors investigate how different English variations impact transformer-based models for irony detection. To conduct their study, they used the EPIC corpus to extract five diverse English variation-specific datasets and applied the KEN pruning algorithm on five different architectures. Their results reveal several similarities between optimal subnetworks, which provide insights into the linguistic variations that share strong resemblances and those that exhibit greater dissimilarities. The authors discovered that optimal subnetworks across models share at least 60% of their parameters, emphasizing the significance of parameter values in capturing and interpreting linguistic variations. This study highlights the inherent structural similarities between models trained on different variants of the same language and also the critical role of parameter values in capturing these nuances.

C.1 Introduction

Sentiment analysis datasets, particularly those annotated on crowdsourcing platforms, may contain biases due to the lack of information about the cultural backgrounds of the annotators. This can lead to machine learning models trained on this data amplifying these biases, affecting how people perceive and label sentiment. Although these models can capture general sentiment, they often fail to capture the nuances experienced by different groups.

This paper examines the impact of linguistic diversity on transformer models designed for irony detection. Using the EPIC corpus [46], we created five subsets tailored to different variations of English. We trained different transformer models and used the KEN pruning algorithm [94] to extract the minimum subset of optimal parameters that maintain the original performance of the model. We conducted this experimental process across five transformer architectures, revealing a minimum parameter overlap of 60% among resulting subnetworks. We then performed a comprehensive analysis to identify subnetworks with the highest and lowest similarity. Additionally, we used KEN_{viz} for a visual examination of pattern similarities. Our results show that the linguistic variation is closely related to the individual values of each parameter within the models. This suggests that the diversity among linguistic variation is not just a structural aspect, but is deeply rooted in the specific values contained in the model. These insights can help create models that better capture the richness of linguistic variation and address bias effectively.

C.2 Background and related work

Artificial intelligence (AI) models impact our daily lives in many ways. Some applications go beyond just processing data and strive to understand the intricate human elements and cultural nuances of our world. For instance, sentiment analysis requires a deeper understanding of implicit phrases and cultural differences to accurately interpret emotions [152, 147]. This

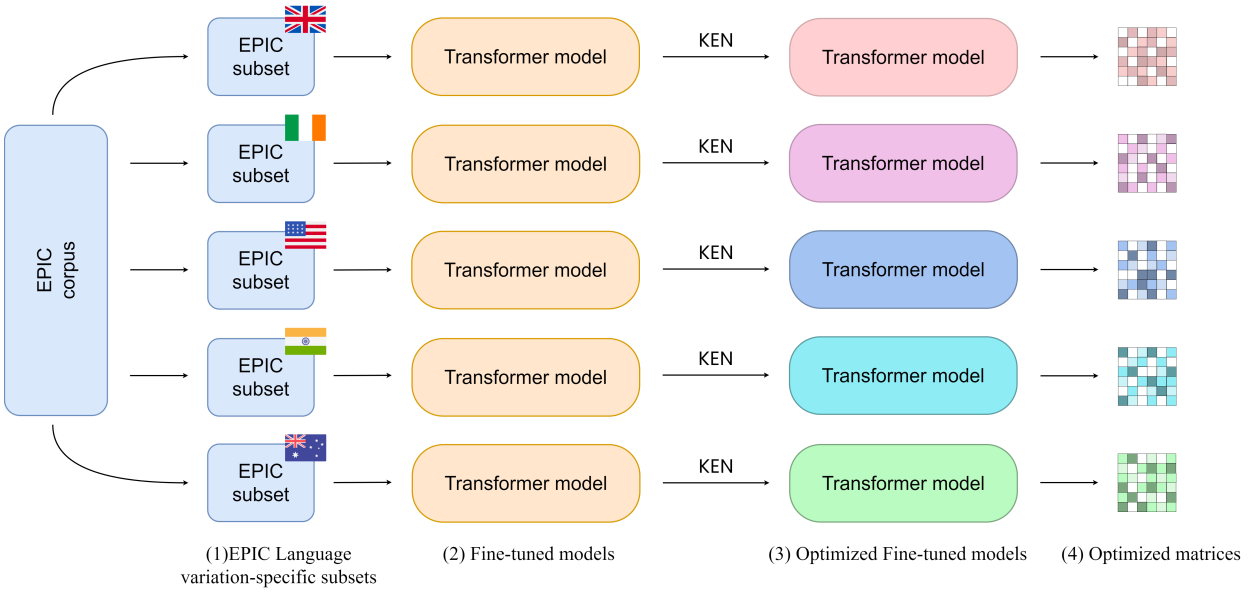


Figure C.1: Workflow overview. Specific language variations are selected from the EPIC corpus (1). For each unique language subset, a dedicated transformer model is trained (2). This ensures that each model specializes in the intricacies of its assigned language variation. Finally, the KEN pruning algorithm is applied to optimize the trained models (3). This involves efficient and lightweight architectures for each language variant (4).

is why rigorous studies are essential before deploying data and models in real-world settings. When creating data, it is crucial to incorporate different perspectives evaluation standards, such as "golden standards" [9], incorporating criteria for evaluating annotators [100, 2, 99], grouping them according to potential bias factors [43] or using text visualization techniques to analyze annotated datasets [57]. On the model level, explainable AI (XAI) techniques [132, 131, 156] are being used to demystify complex models and ensure transparency. Many neural interpretability models rely on attention-based techniques [18], utilizing auxiliary tasks [36], or external knowledge integration [174]. Moreover, attention-based models exhibit a grasp of the syntactic structure of analyzed sentences [88]. Consequently, the role of syntax in model interpretation is being extensively studied across various domains, including irony [27] and hate speech [93, 92]. This multifaceted exploration contributes to a richer understanding of the interplay between language, culture and model interpretability to achieve increasingly inclusive AI models.

C.3 Methods

This section introduces the core components of our research: the EPIC corpus and the KEN pruning algorithm. Sec. C.3.1 provides an in-depth exploration of the EPIC corpus, explaining its composition and the diverse language varieties it encompasses. Sec. C.3.2 analyzes the KEN pruning algorithm, emphasizing its key role in transformer model optimization.

C.3.1 EPIC Corpus

The EPIC [46] corpus consists of 3,000 conversations from social media platforms. It covers five different varieties of English, including Australian (AU), British (GB), Irish (IE), Indian (IN) and American (US). The corpus offers valuable insights into how cultural and linguistic factors shape the perception of irony, giving a comprehensive analysis of it from different perspectives.

To ensure the authenticity of the data, EPIC sources its content from Twitter and Reddit, capturing informal communication across different regions and demographic areas. Rigorous data curation guarantees the inclusion of potential ironies while maintaining a balanced distribution across language varieties, mitigating selection bias. Native speakers from each country independently label instances as ironic or non-ironic, using a multi-perspective annotation process. This ensures a robust and nuanced understanding of cultural humor. Annotators possess robust language skills and familiarity with online communication styles, reinforcing the reliability of their judgments. The inclusive approach in both data collection and annotation facilitates the development of *perspective-aware* models [3] that account for cultural and linguistic variations.

C.3.2 KEN algorithm

KEN (**K**ernel density **E**stimator for **N**eural network compression) [94], is a pruning algorithm designed to extract the most essential subnetwork from transformer models. It exploits the

Model	AU	GB	IE	IN	US
Bert	47.54	58.03	58.03	58.03	58.03
DistilBert	56.26	34.39	50.79	50.79	56.26
DeBerta	44.88	55.91	55.91	55.91	55.91
Ernie	58.03	47.54	58.03	58.03	58.03
Electra	91.18	91.18	64.75	91.18	82.37

(a) Pertentage of parameter reset after the KEN pruning step for all the models on each language variation subsets analyzed. The percentage indicates the number of parameters reset to their pre-trained value in the entire model

Model	AU	GB	IE	IN	US
Bert	+2.0	+2.1	+5.5	+4.6	+0.0
DistilBert	+0.6	+0.0	+3.5	+2.4	+0.0
DeBerta	+1.3	+2.9	+7.2	+1.4	+0.0
Ernie	+0.0	+0.0	+0.0	+13.5	+0.0
Electra	+5.2	+0.7	+1.5	+0.1	+2.1

(b) Variation of the F1-weighted measure across all the language variation subsets after the KEN pruning step. Positive values indicate a score improvement compared to the unpruned version

Table C.1: Result obtained during our experiment: Tab. C.1a shows the percentage of parameter reset of each model in all language variation subsets analyzed while Tab. C.1b presents per F1-weighted performance variation obtained.

winning ticket lottery hypothesis [45], according to which an optimal subset of fine-tuned parameters maintains the same performance as the original one.

KEN leverages Kernel Density Estimations (KDEs) to generalize point distributions for each row of a transformer matrix, resulting in a streamlined version of the original fine-tuned model. By pinpointing the k most representative parameters within each distribution, KEN effectively prunes the network, preserving them while reverting the remaining parameters to their pre-trained state. KEN archives minimum parameter reduction between 25% and 60% for specific models, maintaining equivalent or better performance than their unpruned counterparts. The resultant subnetwork can be seamlessly archived and reintegrated into its pre-trained configuration for diverse downstream applications. This approach not only significantly reduces model size but also enhances efficiency and flexibility across various tasks.

C.4 Experiments

This section provides a detailed explanation of the entire process we followed during our experiment. The process began with the variant-specific datasets extraction to the optimal subnetworks search and the transformer architecture tested.

The EPIC corpus contains approximately 3,000 sentences annotated by multiple annotators, resulting in 14,172 records. To create language-variant-specific datasets, we distilled unique sentences from the corpus and applied majority voting based on annotations, with ties resolved by labeling records as "irony." This meticulous process yielded well-balanced datasets, each comprising approximately 600 records.

Five models, each specializing in a single language variant, were trained using the same transformer architecture. After fine-tuning, we used the KEN pruning algorithm to extract the smallest and most efficient subnetwork in each model. This process involves incrementally increasing the number of fine-tuning parameters retained and decreasing those restored to pre-training values, starting from a minimal subset of parameters and expanding it until the pruned model performance matches or exceeds its unpruned counterpart. Using these optimized subnetworks, we analyzed the internal structures of the models and measured the similarities between the optimized subnetworks across different language variants. For each layer, we extracted the corresponding matrices and conducted a meticulous analysis of the positions of the optimal parameters within each optimal subnetwork. This involved an *"in-breadth" analysis*, which identified the parameters present in all optimal models examined and *pairwise comparisons* between models to identify the language variants with the greatest and least similarity, regardless of the model architecture. We conducted these analyses for each architecture under examination on the layers that constitute the attention mechanism or similar structures, as these layers concentrate most of the arithmetic operations of the model and are a strength of the transformer model core structure.

We replicate this experiment across five distinct transformer model architectures, including Bert [38], DistilBERT [133], DeBERTa [58], Ernie [148] and Electra [28]. The provided Fig. C.1 visually depicts the entire workflow, starting with language variety subset extraction to the resulting optimized subnetworks obtained.

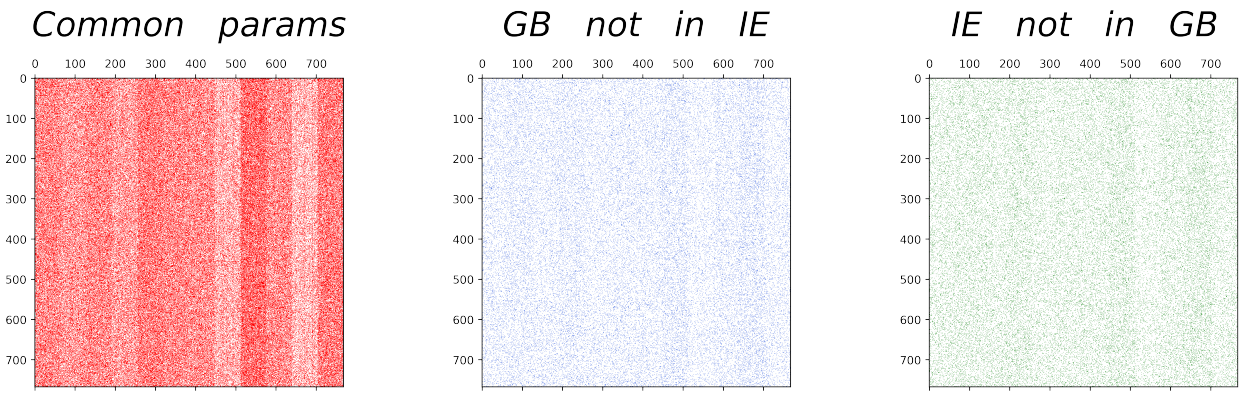


Figure C.2: Comparison of the optimal subnetworks of two DeBERTa models (layer 0, attention output matrix) trained on British (GB) and Irish (IE) linguistic variation, respectively. The matrix on the left shows the number of common parameters between the two matrices (subnetwork overlap), while the middle one shows the location of the optimal parameters of the GB subnetwork not present in IE, and on the right the exact opposite. Blank values refer to the values not belonging to the optimal network and thus the collection of points that the KEN algorithm has reset to their pre-training value. Additional results are shown in Apx. C.7

C.5 Results

The KEN algorithm is an effective method for selecting the best model parameters for each language variation. The rate at which these parameters are reset varies across different architectures, as shown in Tab. C.1a. However, this resetting rate consistently exceeds 50% on average. Surprisingly, despite the substantial resetting, performance actually improves in most cases, as demonstrated by the F1-weighted scores in Tab. C.1b. Notably, these results were achieved through tuning steps on relatively small data sets, with only 600 examples per variation. It is essential to note that our primary goal was not to establish new state-of-the-art (SoTa) models, but rather to investigate the impact of language variations on model parameters within each architecture examined. From this perspective, the results are encouraging and demonstrate a positive impact. Additionally, the varying percentages of parameter resets among linguistic variations using the same architecture contribute to a more nuanced understanding of the optimal subnetworks and their comparison. After examining subnetwork structures, it was discovered that two optimal subnetworks share at least 60% of

Subnet A	Subnet B	BERT	DeBERTa	DistilBERT	Ernie	Electra
AU	GB	69.73	69.94	61.69	69.81	89.49
AU	IE	69.79	69.94	75.22	82.72	23.15
AU	IN	69.73	69.94	75.17	83.22	87.6
AU	US	69.73	69.94	83.42	83.22	29.09
GB	IE	83.02	82.74	69.38	69.76	23.15
GB	IN	82.59	82.71	69.38	69.81	86.95
GB	US	82.59	82.71	61.66	69.81	29.06
IE	IN	82.6	82.86	85.85	82.39	23.15
IE	US	82.6	82.86	75.17	82.39	69.68
IN	US	>90.0	>90.0	76.22	>90.0	29.45

Table C.2: Similarity percentages between subnetworks specific to language variation. Percentages are obtained by comparing for each model the number of non-reset parameters within each attention (or similarity) layers

their parameters. This percentage, however, does not take into account parameters reset by KEN, which could significantly impact the final result. Tab. C.2 indicates that Indian (IN) and American (US) variations have the highest overlap, with more than 90% in three out of five models. British (GB) and Irish (IE) also have considerable overlap across all models, which is highly desirable. Despite extensive analysis, identifying the most distinct variants remains challenging, as the percentage difference between pairs of language variations across all models is relatively small.

In addition to tabular descriptions, we have graphically presented the results obtained. Through KEN_{viz}, three different types of results are visualized: (1) the subnetwork overlap of two language variations within the same selected matrix layer, (2) fine-tuned parameters chosen for the linguistic variation A but not for B and (3) the reverse. Fig. C.2 showcases one of the obtained results, while Apx. C.7 provides more case studies by analyzing results across all models in their last attention layer for specific linguistic variations. These graphical representations offer insights into the precise placement of optimal parameters and the shared or differing structures between models.

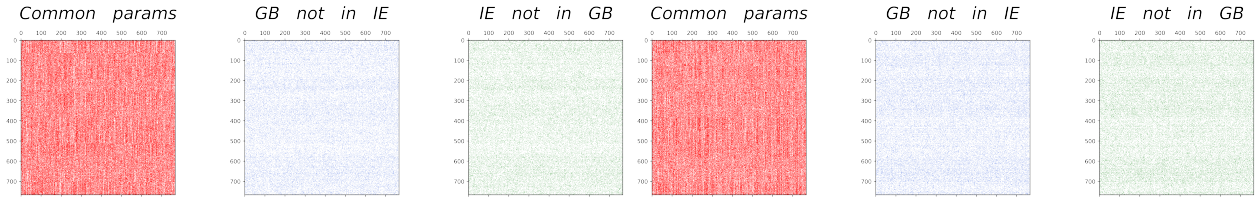
C.6 Conclusion

This study conducted a thorough analysis of different transformer models to discover their divergences in detecting irony when trained on different linguistic variants. We uesd the EPIC corpus and created language-variant-specific datasets for five English variations (American, British, Indian, Irish and Australian). Using the KEN pruning algorithm, we extracted optimal subnetworks from five transformer architectures (BERT, DistilBERT, DeBERTa, Ernie and Electra) tailored to each language variation. Our study revealed that different linguistic variations share a remarkable number of parameters, regardless of the architecture used. We provided insights into the similarity of each pair of optimized subnetwork linguistic variations by reporting the percentage of common parameters. However, we found it challenging to rank the dissimilarity since the shared parameter percentage remained consistently high in all cases. To enhance our understanding of how linguistic diversity manifests in the models, we used KEN_{viz} to provide a graphical view of the specific locations of shared and distinct parameters across models and language variations. Despite limitations like dataset size, our study shows that training transformer models and adapting them to linguistic variations produce highly similar output models, illustrating that their differences are rooted in their parameter values.

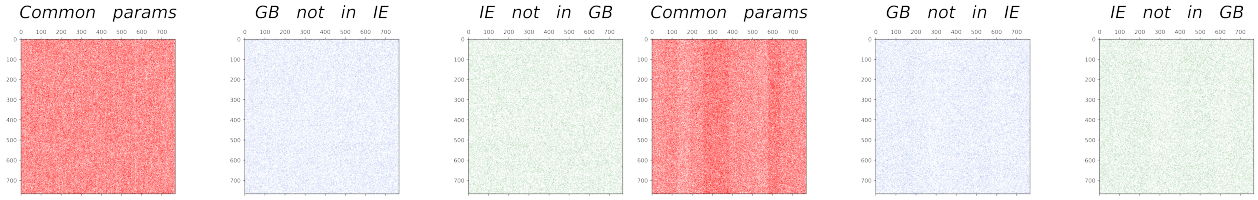
C.7 Additional results

We present some graphical results obtained using KEN_{viz} by analyzing the output of attention matrices in the last levels for each model analyzed. We selected several pairs of linguistic variations for each model that showed the most interesting results based on the findings in Tab. C.2. These visual results highlight the commonalities found within the optimal subnetworks and show the difficulty of finding differences between them. However, in some cases, parameter selection focuses more on certain areas than others.

C.7.1 DeBerta model fingerprints comparison



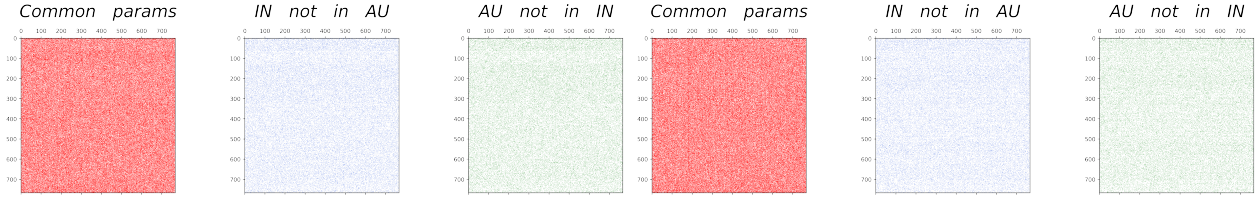
(a) `q_proj` matrices (b) `pos_proj` matrices



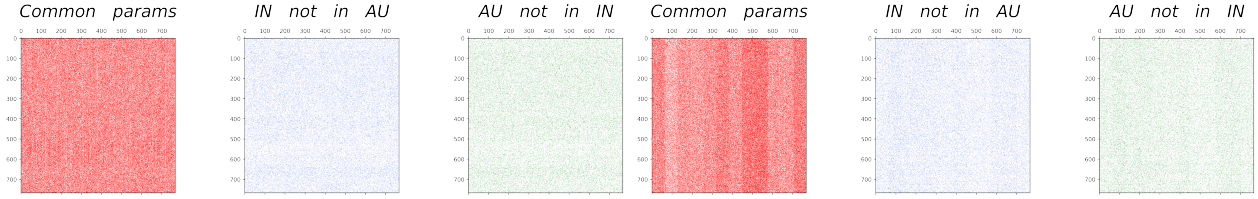
(c) `in_proj` matrices (d) Output matrices

Figure C.3: Comparison of optimal parameter selection in Layer 12 of DeBERTa across two language-specific subnetworks. Each subfigure (a–d) shows a different projection matrix, with three corresponding heatmaps: shared parameters (left), A-exclusive (center), B-exclusive (right). White areas denote pruned weights

C.7.2 Ernie model fingerprints comparison



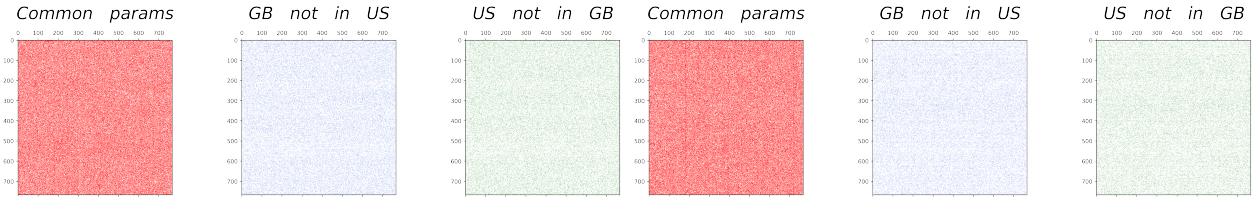
(a) Key matrices (b) Query matrices



(c) Value matrices (d) Output matrices

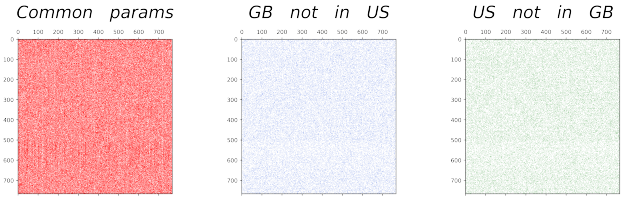
Figure C.4: Comparison of optimal parameter selection in Layer 11 of Ernie across two language-specific subnetworks. Each subfigure (a–d) shows a different projection matrix, with three corresponding heatmaps: shared parameters (left), A-exclusive (center), B-exclusive (right). White areas denote pruned weights

C.7.3 BERT model fingerprints comparison



(a) Key matrices

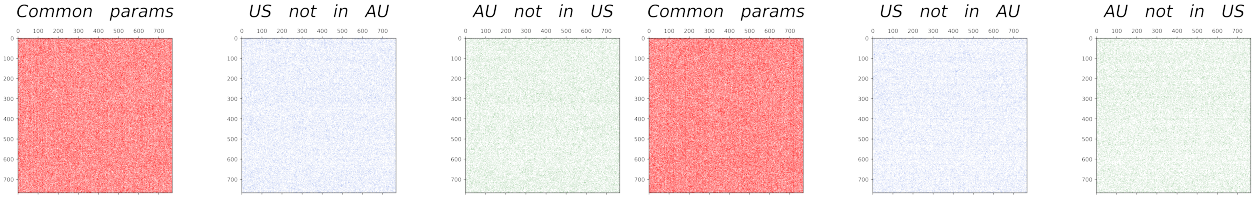
(b) Query matrices



(c) Value matrices

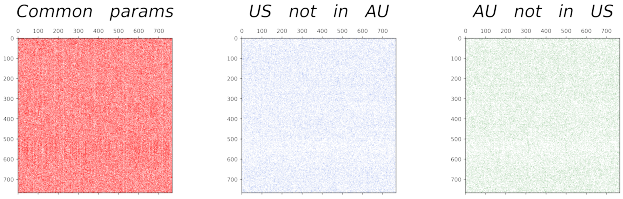
Figure C.5: Comparison of optimal parameter selection in Layer 12 of BERT across two language-specific subnetworks. Each subfigure (a–c) shows a different projection matrix, with three corresponding heatmaps: shared parameters (left), A-exclusive (center), B-exclusive (right). White areas denote pruned weights

C.7.4 DistilBERT model fingerprints comparison



(a) `k_lin` matrices

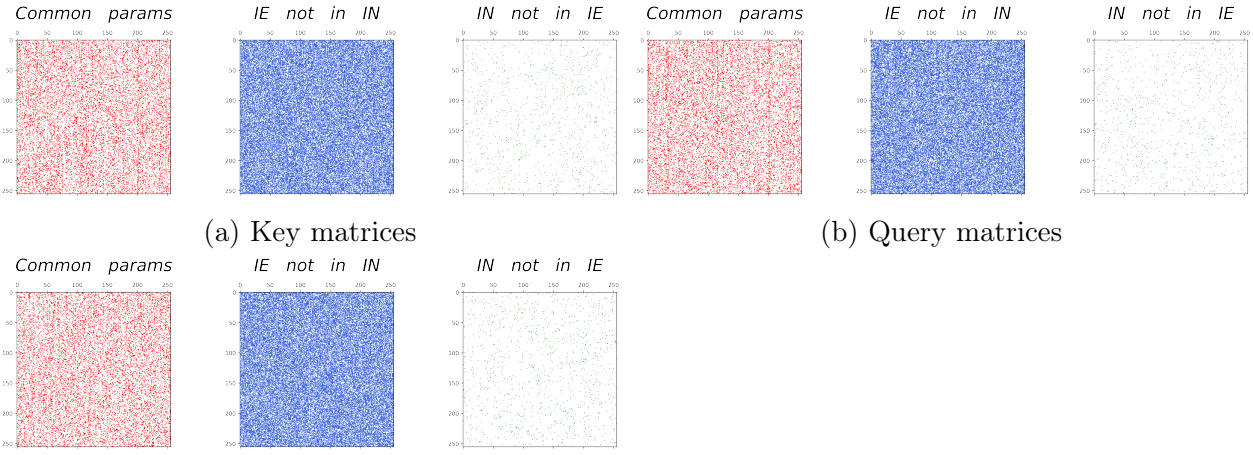
(b) `q_lin` matrices



(c) `v_lin` matrices

Figure C.6: Comparison of optimal parameter selection in Layer 5 of DistilBERT across two language-specific subnetworks. Each subfigure (a–c) shows a different projection matrix, with three corresponding heatmaps: shared parameters (left), A-exclusive (center), B-exclusive (right). White areas denote pruned weights

C.7.5 Electra model fingerprints comparison



(a) Key matrices

(b) Query matrices

(c) Value matrices

Figure C.7: Comparison of optimal parameter selection in Layer 12 of Electra across two language-specific subnetworks. Each subfigure (a–c) shows a different projection matrix, with three corresponding heatmaps: shared parameters (left), A-exclusive (center), B-exclusive (right). White areas denote pruned weights

Appendix D

Other Works

This section highlights three of my research projects published during my PhD, which explore syntax-based modeling and hybrid systems to address the challenges of hate speech detection and dialogue systems. While these studies are not the central focus of this thesis, they contribute valuable insights into the development of AI models that are robust, explainable, and ethical, enriching the broader discourse on AI research.

The first paper *Syntax and Prejudice: Unveiling the Ethically Charged Biases of a Syntax-Based Hate Speech Recognizer* [93] (Apx. D.1), investigates the potential of syntax-based hate speech recognizers to mitigate biases in AI models. Through the implementation of KERM-HATE, a syntax-based system, this work uncovers both strengths and limitations, emphasizing the ethical implications of biased models and advocating for greater transparency and accountability in AI design.

The second study *Change My Mind: A Syntax-Based Approach to Understanding Perspective-Driven Hate Speech* [92] (Apx. D.2), examines how cultural backgrounds shape perceptions of hate speech. By employing explainable syntax-based models and introducing a novel contrasting tree analysis, this research demonstrates how linguistic nuances and annotator

perspectives influence model predictions, underscoring the importance of culturally aware datasets in hate speech detection.

The third paper *Every Time I Fire a Conversational Designer, the Performance of the Dialog System Goes Down* [166] (Apx. D.3) explores the integration of human expertise in neural dialogue systems. The study introduces CLINN, a hybrid system that combines neural networks with semi-logical rules crafted by conversational designers. It highlights the significant role of domain-specific knowledge in reducing data requirements and enhancing system performance.

Then, *Lacking the Embedding of a Word? Look it up into a Traditional Dictionary* [128] (Apx. D.4) proposes two novel methods, DefiNNet and DefBERT, which use traditional dictionary definitions to derive embeddings for rare or out-of-vocabulary (OOV) words. The study shows these methods outperform models like FastText and BERT for OOV words, demonstrating the value of external lexical resources in improving word embeddings.

Finally, *Exploring Linguistic Properties of Monolingual BERTs with Typological Classification among Languages* [127] (Apx. D.5), investigates how monolingual BERT models encode syntactic features, revealing that syntactic typological similarity correlates with weight matrix similarities in BERT's middle layers. The research highlights the influence of domain adaptation in enhancing these linguistic representations.

These articles illustrate the wide-ranging applications of machine learning models across various domains. Although their focus diverges from the primary objective of this thesis, the findings summarized in the following sections emphasize critical goals for AI systems—achieving high performance, leveraging human conversational insights for model training, and addressing essential aspects such as fairness, explainability, and inclusivity.

D.1 Syntax and Prejudice: Unveiling the Ethically Charged Biases of a Syntax-Based Hate Speech Recognizer

Hate speech recognizers (HSRs) can be the panacea for containing hate in social media or can result in the biggest form of prejudice-based censorship hindering people to express their true selves. In this paper, we hypothesized how massive use of syntax can reduce the prejudice effect in HSRs. To explore this hypothesis, we propose Unintended-bias Visualizer based on Kermit modeling (KERM-HATE): a syntax-based HSR, which is endowed with syntax heat parse trees used as a post-hoc explanation of classifications. KERM-HATE significantly outperforms BERT-based, RoBERTa-based and XLNet-based HSR on standard datasets. Surprisingly this result is not sufficient. In fact, the post-hoc analysis on novel datasets on recent divisive topics shows that even KERM-HATE carries the prejudice distilled from the initial corpus. Therefore, although tests on standard datasets may show higher performance, syntax alone cannot drive the "attention" of HSRs to ethically-unbiased features.

Key Contributions

- **Unveiling Biases in Syntax-Based Models**: The paper illustrates how syntax-based models, while effective in capturing nuanced linguistic cues, can inadvertently amplify existing societal biases.
- **Analyzing the Role of Linguistic Features**: The authors investigate the impact of specific linguistic features, including syntactic constructions and lexical choices, on model predictions. They highlight how certain features, often linked to marginalized groups, can lead to biased classifications.
- **Proposing Ethical Guidelines**: The study offers a set of ethical guidelines for the development and deployment of hate speech detection systems. These guidelines stress the importance of transparency, accountability, and fairness.

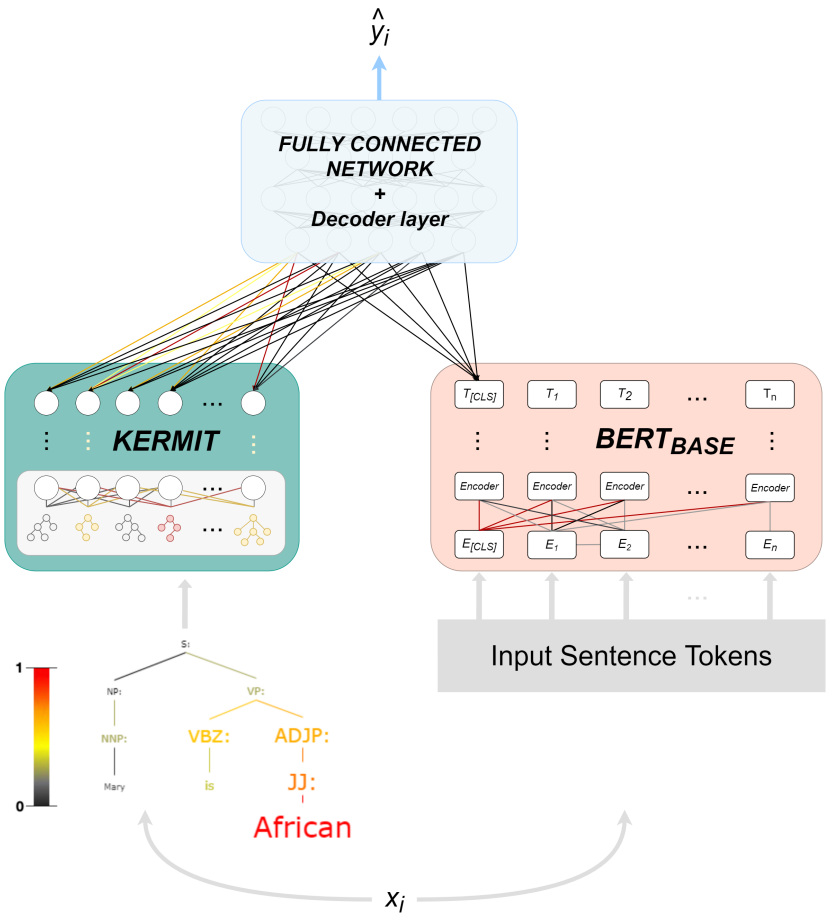


Figure D.1: KERM-HATE architecture

Findings

- **Sensitivity to Linguistic Variation**: Syntax-based models may be sensitive to subtle linguistic variations, potentially resulting in unintended biases. For example, a model might misclassify a sarcastic comment as hate speech if it fails to recognize the intended tone.

- **Impact of Training Data**: The selection of training data significantly affects the model's performance and fairness. Biases present in the training data can be perpetuated or intensified by the model.

- **Ethical Implications of Biased Models**: Biased hate speech detection systems can

have severe consequences, including the suppression of free speech, marginalization of vulnerable groups, and erosion of public trust in AI.

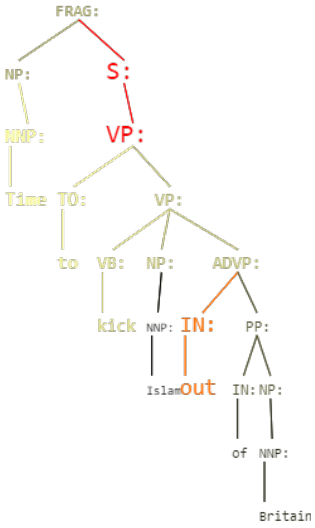
Conclusion

The paper highlights the critical importance of ethical considerations in the development and deployment of AI-powered hate speech detection systems. By understanding the potential biases inherent in syntax-based models, researchers and practitioners can strive to create fairer and more equitable systems. The study advocates for a multidisciplinary approach that merges technical expertise with ethical considerations to ensure that AI serves to promote good rather than cause harm.

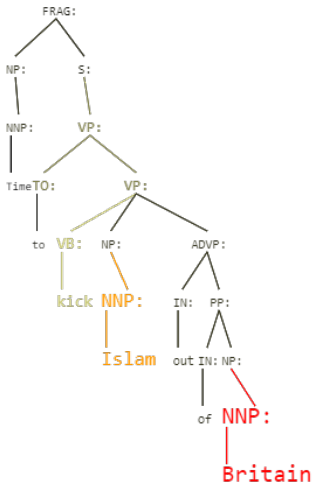
D.2 Change My Mind: A Syntax-Based Approach to Understanding Perspective-Driven Hate Speech

Hate speech recognizers may mislabel sentences by not considering the different opinions that society has on selected topics. In this paper, we show how explainable machine learning models based on syntax can help to understand the motivations that induce a sentence to be offensive to a certain demographic group. To explore this hypothesis, we use several syntax-based neural networks, which are equipped with syntax heat analysis trees used as a post-hoc explanation of the classifications and a dataset annotated by two different groups having dissimilar cultural backgrounds. Using particular *contrasting trees*, we compared the results and showed the differences. The results show how the keywords that make a sentence offensive depend on the cultural background of the annotators and how this differs in different fields. In addition, the syntactic activations show how even the sub-trees are very relevant in the classification phase.

Sentence: *Time to kick Islam out of Britain*



(a) Tree obtained subtracting from KERMIT$_C$ activation values those of KERMIT$_T$



(b) Tree obtained subtracting from KERMIT$_T$ activation values those of KERMIT$_C$

Figure D.2: Contrasting trees

Key Contributions

1. **Polarized Dataset**: This study uses a dataset annotated by two groups with distinct cultural backgrounds, allowing for the exploration of perspective-driven hate speech.

2. **Explainable Models**: We employ explainable syntax-based models, such as KERMHATE, to gain insights into their decision-making processes.

3. **Contrasting Tree Analysis**: We introduce a novel technique to compare the activation patterns of different models applied to the same sentence, emphasizing the syntactic structures that lead to varied classifications.

Findings

- **Cultural Influence on Perception**: The results show that the cultural backgrounds of the annotators significantly affect their perception of hate speech. This leads to variations in identifying offensive keywords and syntactic structures.
- **Syntactic Nuances**: The analysis reveals that not only individual words but also the underlying syntactic structures play a crucial role in detecting hate speech.
- **Model Limitations**: Traditional hate speech recognition models, which often depend on simple keyword matching, tend to overlook nuanced linguistic cues and cultural contexts.

Conclusion

This study emphasizes the importance of considering diverse perspectives and the limitations associated with relying solely on common hate speech corpora. By utilizing syntax-based explainable models and a culturally diverse dataset, we present a promising approach to understanding and addressing the challenges of hate speech detection in a complex and multifaceted society.

D.3 Every Time I Fire a Conversational Designer, the Performance of the Dialog System Goes Down

Incorporating explicit domain knowledge into neural-based task-oriented dialogue systems is an effective way to reduce the need of large sets of annotated dialogues. In this paper, we investigate how the use of explicit domain knowledge of conversational designers affects the performance of neural-based dialogue systems. To support this investigation, we propose the Conversational-LogicInjection-in-Neural-Network system (CLINN) where explicit knowledge

is coded in semilogical rules. By using CLINN, we evaluated semi-logical rules produced by a team of differently-skilled conversational designers. We experimented with the Restaurant topic of the MultiWOZ dataset. Results show that external knowledge is extremely important for reducing the need of annotated examples for conversational systems. In fact, rules from conversational designers used in CLINN significantly outperform a state-of-the-art neural-based dialogue system.

Key Contributions

1. **Human Expertise Matters**: The paper highlights the invaluable insights and intuition of human designers in enhancing dialogue system performance.

2. **CLINN**: A Hybrid Approach: The proposed Conversational-Logic-Injection-in-Neural-Network (CLINN) system combines neural network models with explicit domain knowledge encoded in semi-logical rules. This hybrid approach leverages the strengths of both data-driven and knowledge-driven techniques.

3. **Improved Performance**: CLINN, which incorporates rules developed by conversational designers, demonstrates superior performance compared to state-of-the-art neural-based dialogue systems, particularly when trained on smaller datasets.

Findings

- **Human-engineered rules can significantly enhance the performance of neural-based dialogue systems**. By explicitly encoding domain knowledge, CLINN can improve the system ability to reason, plan, and generate appropriate responses.

- **CLINN effectively leverages the strengths of both data-driven and knowledge-driven approaches**. By combining the power of neural networks with the precision of logical rules, CLINN can achieve a better balance between flexibility and control.

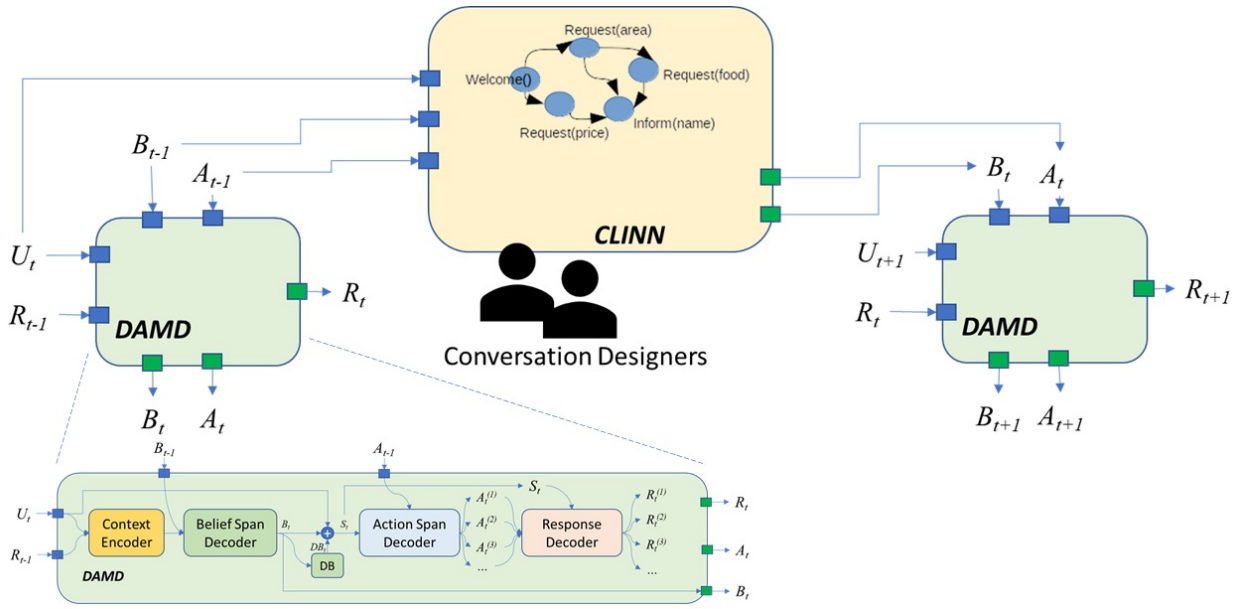


Figure D.3: Injecting external handwritten domain scripts with the Conversational-Logic-Injection-in-Neural-Network (CLINN) and the architecture of the Domain Aware Multi-Decoder (DAMD) network

- **Incorporating domain-specific knowledge can mitigate the limitations of data-driven models**. By providing the model with explicit knowledge about the domain, CLINN can generate more accurate and informative responses, even when trained on limited data.

Conclusion

The paper emphasizes the critical importance of incorporating human expertise in the field of conversation design. By combining human intuition with machine learning techniques, we can create more advanced dialogue systems. Recognizing the value of human insight and domain knowledge enables us to develop conversational agents that are both robust and engaging. The findings of this research have significant implications for the future development of dialogue systems, especially in areas where data is limited and where high levels of accuracy and reliability are essential.

D.4 Lacking the embedding of a word? look it up into a traditional dictionary

Word embeddings are foundational for capturing semantic and syntactic relationships in natural language processing (NLP). However, standard word embedding models often fail when encountering out-of-vocabulary (OOV) words—terms absent from their training data—which can significantly degrade downstream task performance. OOV issues are particularly common for rare, technical, or low-resource language words. To address these challenges, Ruzzetti et al. [128] propose leveraging dictionary-based definitions as a rich semantic resource to generate embeddings for OOV words. They introduce two innovative models, *Definition Neural Network* (DefiNNet) and *Define BERT* (DefBERT), which enhance word embedding spaces by incorporating the compositional structure of dictionary definitions.

The core idea is inspired by human language acquisition: when encountering an unknown word, people often rely on its dictionary definition to infer its meaning. By mimicking this process computationally, DefiNNet and DefBERT create meaningful embeddings for OOV words, thereby overcoming traditional embedding limitations. These models are designed not as standalone alternatives but as complementary enhancements to existing embedding spaces like Word2Vec and BERT, enriching their semantic coverage.

Key Contributions

The key contributions of this research can be summarized as follows:

- **Dictionary-Informed OOV Embeddings**: DefiNNet and DefBERT utilize dictionary-based definitions to create high-quality embeddings for rare and unseen words. By analyzing the semantic and compositional structure of definitions, the models generate embeddings that capture subtle word relationships.

- **Two complementary approaches:**
  - DefiNNet: A neural architecture that processes definitions in two steps. First, a DefAnalyzer extracts key elements such as the supertype (hypernym) and head modifier from the definition. These elements are then combined with existing Word2Vec embeddings to create a refined vector for the target word.
  - DefBERT: This model leverages BERT's contextual power to process entire definitions. Two variations are explored:
    * DefBERT$_{[CLS]}$: Uses the [CLS] token embedding, representing the entire definition.
    * DefBERT$_{Head}$: Focuses on the hypernym (supertype) within the definition, enhancing semantic precision.
- **Experimental Framework and Benchmarks:** The models are evaluated on several benchmark datasets, including WordNet-based word similarity and relatedness tasks, to assess their effectiveness in handling OOV words and generating indirect embeddings. The performance is compared against established baselines, such as additive models and standard BERT embeddings.

Findings

The experiments yield several key findings, demonstrating the advantages of dictionary-informed embedding approaches:

- **Enhanced semantic representation:** DefiNNet and DefBERT achieve superior performance on both word similarity and word relatedness tasks, as evidenced by higher cosine similarity and Spearman's correlation scores on benchmark datasets. These improvements highlight the models' ability to capture nuanced semantic relationships, especially for rare words.

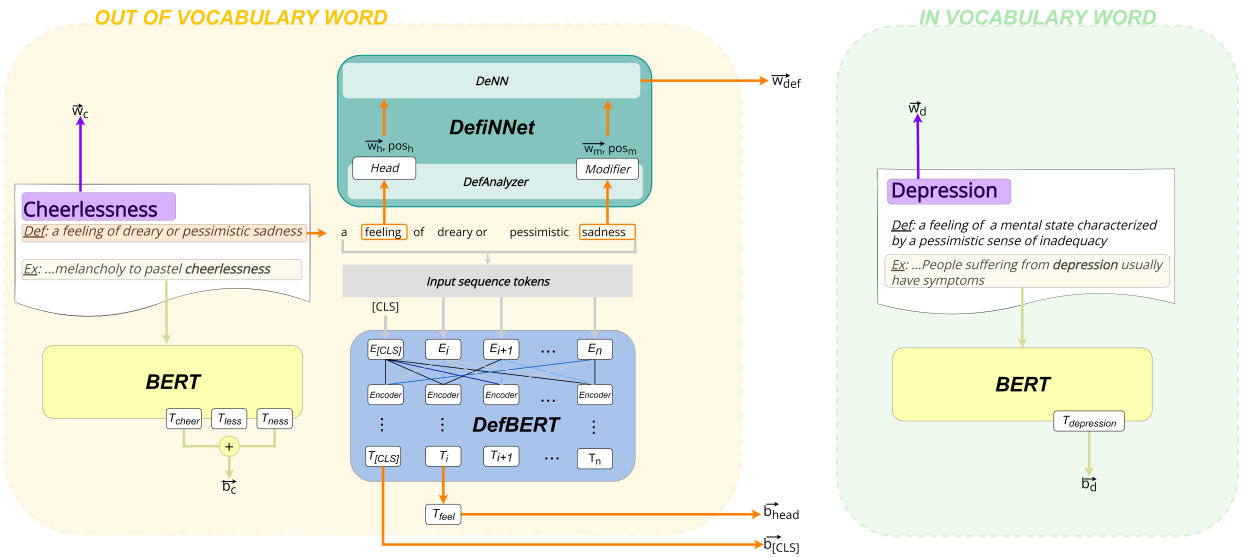


Figure D.4: Exploiting definitions of out-of-vocabulary (OOV) words: DefiNNet and DefBERT models

- **Strong correlation with WordNet similarity measures:** The models exhibit particularly strong alignment with the WordNet-based *res* similarity measure, which leverages information content to evaluate word relatedness. This suggests that dictionary definitions effectively encode the semantic nuances required for robust word embeddings.

- **OOV handling capability:** By leveraging dictionary definitions, DefiNNet and DefBERT can generate embeddings for OOV words that closely resemble their in-vocabulary counterparts. This is crucial for improving performance in NLP applications such as machine translation, text summarization, and question-answering systems.

- **Complementarity with existing embedding spaces:** Rather than replacing traditional word embeddings, DefiNNet and DefBERT enhance their semantic scope by providing additional embeddings for previously unseen words. This allows for better generalization in downstream NLP tasks.

- **Robustness to low-resource settings:** Dictionary-based approaches are particu-

larly beneficial in low-resource settings, where corpus-based embeddings may suffer due to limited training data. By exploiting definitions, the proposed models can compensate for the scarcity of linguistic data.

Conclusion

This research demonstrates the potential of dictionary-informed embeddings to address the longstanding OOV problem in NLP. By incorporating dictionary definitions into the word embedding process, DefiNNet and DefBERT significantly enhance the semantic understanding and interpretability of language models. The models' ability to generate high-quality embeddings for rare and unseen words opens up new possibilities for improving low-resource language processing and building more robust, semantically aware NLP systems. Future work could explore integrating these dictionary-based techniques with multilingual embeddings, probing deeper into the relationship between definitions and universal sentence embeddings, and applying the models to specific NLP tasks, such as cross-lingual transfer learning or domain-specific adaptation.

D.5 Exploring Linguistic Properties of Monolingual BERTs with Typological Classification among Languages

This paper embarked on a detailed exploration of transformer-based language models, specifically Monolingual BERTs, with the goal of elucidating how these models encode the structural properties of natural language. Driven by the significant advancements in Natural Language Processing (NLP) attributed to transformer architectures, this research aimed to move beyond surface-level performance metrics and delve into the internal representations constructed by these models. A core motivation was to understand the mechanisms through which these models capture and process linguistic properties, particularly across the diverse spectrum of human languages. To this end, the authors employed typological classifica-

tion, a methodology leveraging linguistic information about languages, to assess the extent to which BERT models comprehend and represent these inherent linguistic features. This investigation sought to illuminate the cross-linguistic capabilities of these models, examining how they handle variations and similarities across different language families, thereby contributing to a more nuanced understanding of their linguistic comprehension.

Key Contributions:

- **Application of typological classification for BERT analysis**: The authors' primary contribution was the innovative application of typological classification as a methodological tool to scrutinize the linguistic properties encoded within Monolingual BERTs. This approach facilitated a systematic and quantifiable assessment of how effectively these models capture and represent various linguistic features. By utilizing typological databases, they established a framework to correlate the internal representations of BERT with established linguistic classifications, enabling a more interpretable analysis of the model's behavior.

- **Cross-Linguistic variation and similarity Analysis** The authors provided detailed insights into how Monolingual BERTs handle the inherent variations and similarities that exist across different languages. They explored the model's ability to generalize linguistic properties from one language to another, shedding light on its cross-linguistic transfer learning capabilities. The study investigated the impact of language family relationships on the model's performance, examining whether related languages exhibit similar representation patterns.

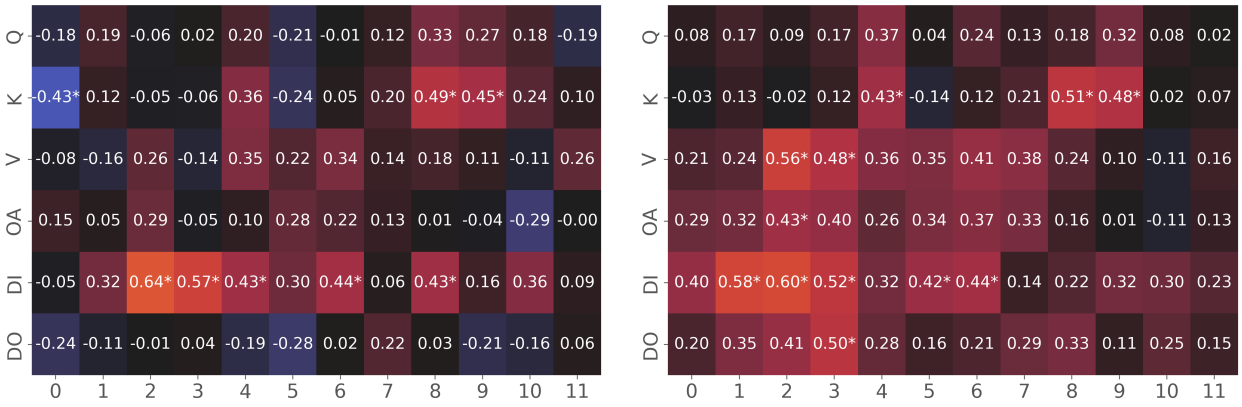
- **Detailed analysis of linguistic property encoding:** The authors conducted a fine-grained analysis of how specific linguistic properties, such as word order, morphological complexity, and syntactic structures, are encoded within the BERT representations. They examined the model's sensitivity to these properties, identifying which features

are well-captured and which pose challenges. The authors provided a method for understanding which layers of the BERT model, are responsible for encoding what type of linguistic information.

- **Methodological Framework for Model Analysis:** The authors established a methodological framework that can be adopted by other researchers to analyze the internal workings of language models. This framework enables the systematic exploration of how linguistic information is represented within these models, facilitating further research in this area.

Findings

- **Emergence of syntactic features:** BERT's middle layers show the highest syntactic similarity across typologically related languages, suggesting that these layers may serve as a repository for shared linguistic features.
- **Impact of domain adaptation:** Domain adaptation increases typological alignment, meaning BERT's linguistic representations become more similar when trained on related language domains. This finding emphasizes the adaptability of monolingual BERT models in encoding generalizable linguistic patterns.
- **Typological Clustering:** The CKA results show that languages with shared typological properties (e.g., SVO vs. SOV languages) exhibit stronger syntactic alignment within the middle BERT layers, reinforcing the hypothesis that typology influences BERT's internal representations.
- **Broader Implications for Multilingual NLP:** These findings highlight the potential of using typologically motivated training to improve multilingual models, where shared syntactic features can enhance cross-lingual transfer and generalization.



(a) Syntactic Similarities before domain adaptation

(b) Syntactic Similarities after domain adaptation

Figure D.5: Spearman's correlation between European language pairs, ranked by typological features and BERT similarity matrices (domain-adapted and pretrained). Redish colors indicate stronger positive correlations, and blueish colors stronger negative correlations. Statistically significant results ($p < 0.01$) are marked with *.

Conclusion

This research contributes to the understanding of how BERT models encode linguistic properties, specifically syntactic typology, by revealing that syntactic features become most apparent in the middle layers of BERT. The findings demonstrate that domain adaptation plays a significant role in enhancing syntactic alignment across typologically similar languages. These insights open up new avenues for improving multilingual models by leveraging typological data, ensuring better generalization and transfer learning. Ultimately, the study encourages a more nuanced approach to training and adapting BERT-based models, emphasizing the importance of linguistic structure in enhancing the performance of multilingual NLP systems.