ID N. AIDR01/16883

Università di Catania

UNIVERSITÀ CAMPUS BIO-MEDICO DI ROMA

DEPARTMENT OF ENGINEERING

UNIVERSITÀ DEGLI STUDI DI CATANIA

DIPARTIMENTO DI INGEGNERIA ELETTRICA, ELETTRONICA E INFORMATICA

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

Advancing AI in Healthcare Through Federated Continual Learning

Supervisors
Prof. Concetto Spampinato
Prof. Simone Calderara

Candidate
Matteo Pennisi

January, 2025

Abstract

Artificial neural networks have gained significant attention, beginning also to influence healthcare. However, integrating AI into medical applications comes with multiple challenges. Deep neural networks face difficulties with non-i.i.d. (non-independent and identically distributed) medical data, which limits their accuracy and generalizability. Medical data, in particular, is highly decentralized and continually evolving, adding complexity to AI deployment and adaptation in clinical settings. Moreover, privacy regulations hinder the centralization of patient data, complicating efforts to train effective models. These factors make the effective integration of AI in healthcare both critical and complex. This thesis tackles these challenges by focusing on Continual Learning and Federated Learning, employing bio-inspired methods and generative AI techniques to advance AI adoption in healthcare. These approaches aim to improve adaptability and privacy-preserving model training, addressing key barriers to effective AI integration in medical settings.

This thesis begins by exploring Continual Learning, a research area focused on reducing model forgetting. To address this, various bio-inspired solutions are proposed, leveraging auxiliary knowledge, auxiliary tasks, and pretraining techniques to enhance learning retention. The thesis then shifts focus to Federated Learning, which enables distributed model training while protecting data privacy—a critical requirement in healthcare. In this section, realistic federated medical scenarios are simulated, and methods are introduced to facilitate data sharing within privacy-preserving frameworks. Specifically, a GAN-based latent space aggregation method is proposed, transforming private datasets into an aggregated and shareable form. This aggregation strategy is further refined through a privacy-preserving latent space navigation technique, increasing the generation of shareable samples by a GAN trained on medical data (such as chest X-rays and retinal fundus images).

The thesis progresses by integrating Continual Learning strategies into Federated Learning to address challenges within decentralized medical applications. Building on earlier methods, Continual Learning techniques are combined to a novel Privacy-Preserving GAN, effectively tackling specific obstacles in Medical Federated Learning. The effectiveness of this integrated strategy is assessed on two distinct medical federations focused, respectively,

on tuberculosis classification and skin lesion classification.

Finally, the thesis addresses the more complex scenario of Federated Continual Learning, where data is both spatially distributed across nodes and evolves over time. To meet this challenge, a communication strategy inspired by experience replay is introduced, enabling effective inter-client communication across nodes.

This comprehensive exploration lays a strong foundation for advancing AI in healthcare by merging Federated Learning and Continual Learning, demonstrating pathways for effective privacy-preserving models suitable for real-world medical applications.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

Deep learning [110], and more generally, the field of artificial intelligence, has gained significant attention in recent years. It is now impacting not only the industrial sector, but also everyday consumers: AI technologies (e.g., Large Language Models) are increasingly integrated into daily tasks. AI research has become one of the most active fields globally, with applications that influence disparate domains.

Healthcare, particularly in medical imaging analysis, stands out as one of the most promising yet complex areas of AI application [39,122]. In this field, deep learning has shown high diagnostic potential, enabling faster, more accurate diagnoses able to even detect subtle disease patterns that may be imperceptible to the human eye. Two of the most researched AI applications in medical imaging are classification and segmentation, commonly performed on data from X-rays, PET scans, CT scans, and MRIs. Classification involves predicting a specific label, such as disease presence or severity, from medical images. Segmentation, in contrast, classifies each pixel to distinguish regions of interest, such as organs, tumors, or lesions. These two tasks alone demonstrate AI's potential to enhance the efficiency and accuracy of medical diagnostics, supporting healthcare professionals in their work.

Despite these advancements, real-world adoption of AI in clinical settings remains limited. The intrinsic nature of healthcare demands models that are not only highly accurate but also interpretable by physicians. Currently, many AI models lack this interpretability, which hinders their acceptance among medical professionals and patients alike. Research fields such as Explainable AI (XAI) [202] and Active Learning [22] are working to address this gap. XAI aims to make AI decisions more transparent and understandable, while Active Learning incorporates human expertise throughout the prediction process, fostering trust

and facilitating the safe deployment of AI-driven healthcare solutions.

However, a major challenge in training AI models for medical imaging is the lack of sufficient, high-quality data. For a model to perform well and generalize across diverse data distributions, a large amount of training data is essential. Nevertheless, obtaining such large datasets is particularly challenging in healthcare, especially from a single institution. Existing open-source datasets still do not meet the necessary quality and quantity standards to train production-ready models [147].

One potential solution is to aggregate data from multiple healthcare centers into a centralized dataset. However, this is often unfeasible due to strict privacy regulations that prevent hospitals and diagnostic centers from sharing patient data beyond the facility where it was collected. Furthermore, even when a single center manages to gather a sufficient volume of high-quality data, additional legal constraints, such as those imposed by GDPR [81], can limit its usage. These regulations may restrict data retention periods, limit the use of data for extended purposes, or allow patients to revoke consent for using their personal information.

As a result, data availability and distribution in medical imaging are subject to variability across different locations and over time. This variability adds complexity to AI model development, as models must be trained to adapt to shifts in data distributions while maintaining high performance and generalizability.

In summary, two key challenges will be the focus of this thesis:

- The restrictions on sharing medical data outside individual medical centers.
- The time-sensitive nature of data within a single center, which limits or prohibits the reuse of older data.

The first challenge prevents smaller centers from pooling data to create sufficiently large datasets for training deep learning models. The second challenge affects even larger centers, which, despite having the potential to accumulate enough data, still may face issues due to limitations on reusing historical data.

We believe both of these issues stem from a central limitation in current AI: *catastrophic forgetting* [133]. Catastrophic forgetting makes it difficult to train models effectively on a continuous stream of data, as models tend to lose previously acquired knowledge when new data is introduced sequentially. This limitation hinders the feasibility of training a model sequentially across multiple centers, as the knowledge gained from earlier centers would degrade over time.

Both challenges are also rooted in distribution shifts and the inherently non-i.i.d. (non-independent and identically distributed) nature of medical data. The first issue involves

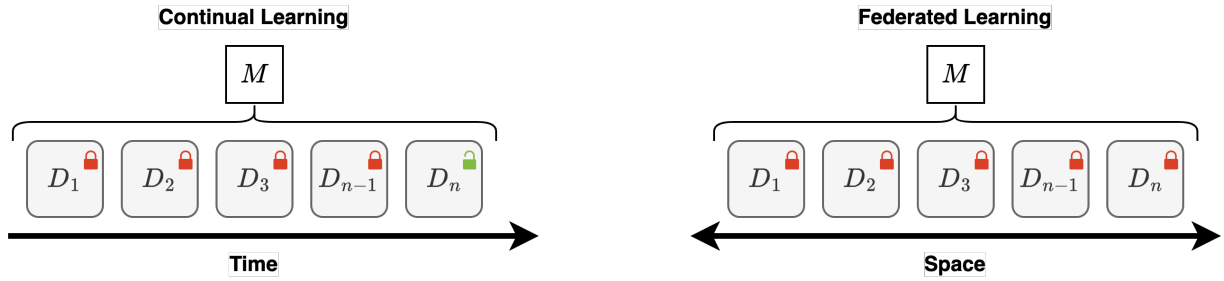

Figure 1.1: **Continual Learning vs. Federated Learning.** In continual learning only the last partition of the data is available for training, while in federated learning each data partition is available to the respective client but cannot be shared to any other client or central server.

non-i.i.d. distributions across space (e.g., data is scattered across different institutions and cannot be easily shared), while the second issue concerns non-i.i.d. distributions over time (e.g., the data distribution within a single center evolves, rendering older data obsolete).

Two areas of deep learning research – Continual Learning (CL) and Federated Learning (FL) –offer promising approaches to address these challenges by facilitating model training in the presence of data limitations and distribution shifts.

Continual Learning (CL) is a field in machine learning focused on training models on a continuous stream of data, where data is presented sequentially in tasks, and data from previous tasks is no longer accessible as training progresses. One of the most widely used and straightforward methods to address catastrophic forgetting is Experience Replay [167]. In Experience Replay, the model retains a small buffer of samples from previous tasks to "rehearse" previously acquired knowledge during training on new tasks.

More formally, let D represent the entire dataset, and $D_1, D_2, \ldots, D_n$ be a sequence of non-overlapping subsets of D, such that $D = D_1 \cup D_2 \cup \ldots \cup D_n$. In CL, training proceeds sequentially: the model is initially trained on D_1, then on D_2, and so forth, without access to past data or knowledge of future tasks. At any given training step t_n, the goal of CL is to train a model M—which was previously trained on $D_1, \ldots, D_{n-1}$ (now inaccessible)—to perform well across the entire data distribution D, despite only having access to the current subset D_n

Federated Learning (FL), instead, aims at training a single, unified model from a federation of nodes that cannot share their data directly. A well-known and straightforward approach in FL is FedAVG [134], where each node in the federation trains a local model on its local data and then sends only the model weights (not the data) to a central server. The server aggregates these weights to update the global model, which is then sent back to each node. This process is repeated iteratively until convergence.

More formally, let D represent the entire dataset, and $D_1, D_2, \ldots, D_n$ be a set of non-overlapping subsets such that $D = D_1 \cup D_2 \cup \ldots \cup D_n$. In FL, training is conducted in parallel, with each node training its own private model, $M_1, M_2, \ldots, M_n$, on its respective dataset $D_1, D_2, \ldots, D_n$. The objective is to produce a global model M that performs well across all the private datasets (i.e., across the overall distribution D), without requiring nodes to exchange raw data.

Figure 1.1 illustrates the key differences between continual learning and federated learning while highlighting their similarities; both approaches address distributional shifts in data and aim to achieve robust performance across diverse data sources.

It is important to highlight that the challenges addressed by Continual Learning and Federated Learning are not mutually exclusive. In real-world scenarios, we often encounter situations with multiple nodes that cannot exchange data due to privacy restrictions (the FL problem), while data within each node evolves over time (the CL problem). This has led to active research in Federated Continual Learning [218]. A relevant application example is the rapid expansion of wearable health devices and IoT systems that continuously collect data, but have limited storage capacities and cannot transmit data to a central server due to privacy concerns.

One potential solution for Federated Continual Learning in medical applications is the use of synthetic data generated by generative models [98, 220]. Synthetic data can support the training of new tasks without leading to the forgetting of past tasks (addressing continual learning) and facilitate knowledge sharing across distributed settings (addressing federated learning). However, generating synthetic data introduces significant challenges, which this thesis will explore further. These challenges range from privacy-preserving issues to the difficulty in creating informative samples. Generative models can sometimes encode real data in their latent space [95], potentially reproducing samples nearly identical to those in the training set, which raises privacy concerns. Additionally, generated data, especially when designed to enhance privacy, may lack sufficient variety and diversity to effectively mimic real data, limiting its usefulness in distributed learning contexts.

1.2 Objectives

Since catastrophic forgetting is a central challenge, the first objective of this thesis is to explore solutions within the field of Continual Learning. In this phase, we develop bio-inspired methods to mitigate forgetting, leveraging auxiliary information, auxiliary tasks, and previously acquired knowledge to support better learning over time.

The second objective focuses on Federated Learning, approached here from a data-centric

perspective. We assume that data sharing is possible through synthetic data generation that includes privacy-preservation guarantees. We test this principle of privacy-safe generative models within realistic federated medical imaging settings such as tuberculosis classification in X-Ray images and diabetic retinopathy classification with reting images.

Finally, this thesis combines Federated and Continual Learning, based on their shared goal of handling distribution shifts, over space and time, in data. In this phase, we integrate continual learning methods within federated settings, using generative models to address data limitations, and then progress to the more complex scenario of Federated Continual Learning.

Overall, this thesis aims to integrate continual and federated Learning principles to address key challenges in medical contexts. This work, thus, represents an initial step towards applying these methods in healthcare, providing foundational insights to support future AI deployment in clinical settings.

Chapter 2

Continual Learning

In this chapter, we begin our exploration toward Federated Continual Learning by first addressing Continual Learning, the AI field focused on mitigating neural network forgetting. As stated earlier in Sec. 2.3.1, a decentralized federation can be framed as a Continual Learning problem, where each node represents a task to learn, and mitigating forgetting involves developing a model capable of handling the data from all nodes within the federation. Our approach to reducing the impact of *catastrophic forgetting* is grounded in three biologically inspired strategies: 1) Humans receive additional, often unrelated visual stimuli when learning new tasks, 2) Humans do not start from zero knowledge when learning something new, 3) Humans are multitask learners, often using other signals (e.g., selective attention) to support complex learning processes like visual categorization.

We translate these three bio-inspired insights into practical methods: a) introducing an auxiliary stream of data during Continual Learning tasks (Section 2.1); b) leveraging prior knowledge from a pretrained, frozen network (Section 2.2); c) incorporating an auxiliary low-level task, such as saliency prediction, to enhance learning in visual classification tasks (Section 2.3).

2.1 Effects of auxiliary knowledge on continual learning

2.1.1 Motivation

Human beings and animals are naturally able to memorize information presented in a sequence [40]; on the contrary, Artificial Neural Networks (ANNs) learning from a non-i.i.d. stream of data incur in *Catastrophic Forgetting* [133,163]. Continual Learning (CL) [109,151]

aims at designing methods that compensate for this issue and facilitate the retention of previous knowledge either by means of regularization [102, 119], architectural designs [177, 182] or (pseudo-)replay of past data [36, 163, 186].

The insurgence of catastrophic forgetting is ascribed to the tendency of models to rewrite their hidden representations as they adjust their parameters to best fit an input distribution that changes in time [63]. However, *McRae & Hetherington* highlight a meaningful difference in the way humans and ML models learn from a sequence of data: whenever human subjects are evaluated on their ability to memorize a sequence of concepts, they start out possessing an already-large body of knowledge [135]. In other words, humans are generalists that can anchor novel data in the context of previous knowledge, while ANNs must specialize on a limited pool of data at each time without any additional reference.

An obvious choice to bridge this gap is *pre-training* the models on a large amount of available off-the-shelf i.i.d. data, leading to a better initialization for the learning procedure [74, 135]. However, we observe that pre-training is not always rewarding in a CL setting, especially in case of small-size replay memories: the ever-changing stream of data entails large changes in model parameters, leading to the forgetting of the pre-training.

We instead propose a learning strategy to limit catastrophic forgetting by providing an additional data stream (uncorrelated from the target data), from which the network can draw auxiliary knowledge. The role of this data stream is to provide models with a more stable representation of the world that can be re-used for incrementally learning new classes or categories leveraging the already-learned low-level features. Indeed, it appears that the human brain can adapt and rewire itself more easily when learning new things related to familiar skills because pre-existing neuronal structure constrains what one can learn [179]. We attempt to enforce this concept into CL through the definition of *an associative rule* that helps learning new classes by measuring the simultaneous firing of neurons between past knowledge and the current data stream. This is implemented through a simple yet effective strategy named *MAH*, that, during a new task, assigns new classes to model's corresponding **m**ost **a**ctivated **h**eads.

Experimental results carried out on standard CL settings, involving CIFAR-10 and (a subset of) Tiny-ImageNet benchmarks, demonstrate that using a separate auxiliary data stream is mostly beneficial with limited size buffer leading to a performance gain of several percent points w.r.t. state-of-the-art methods. Analogously, the MAH strategy reveals to be more effective than the standard class mapping procedure independently from the buffer size. We also investigate the role of model pre-training as compared to sustained auxiliary data employment highlighting that, for small-size buffers, auxiliary data is to be preferred to pre-training.

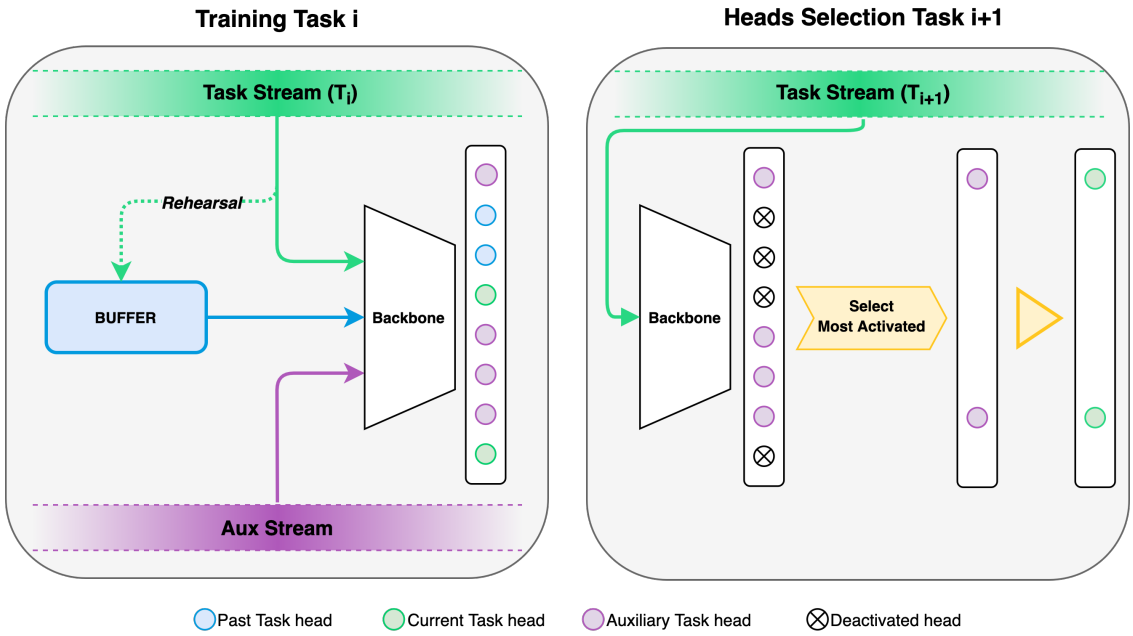

Figure 2.1: **Method overview.** During the training of the current Task T_i the model is trained combining the "current" data coming from the Task Stream and the "past" data stored inside the buffer. In addition, the remaining heads are trained using the auxiliary data stream. At the beginning of a new Task T_{i+1} the MAH procedure is conducted as follows: 1) Only the heads trained with the auxiliary data are kept activated; 2) the T_{i+1} task is forwarded to the frozen model in order to store activation information about the heads; 3) for each class in task T_{i+1}, each new class is assigned to the head that activates the most, replacing the corresponding auxiliary data class.

In conclusion, our strategy is beneficial in Continual Learning from multiple perspectives: the model avoids overfitting current examples, learns more general features and – as auxiliary data-points stand in for future examples – better prepares to learn future classes by suitably associating past knowledge to the new acquired one. All these aspects are mainly observed with reduced buffer size, thus contributing to the efforts that aim at generalizing CL approaches to real-world scenarios.

2.1.2 Related Work

The seminal study by *McCloskey and Cohen* first drew attention to the tendency of ANNs to forget previously learned knowledge catastrophically [133]. In spite of the outstanding results achieved by deep learning models in recent years [189, 203], this problem still persists and prevents ANNs from learning flexibly from non-i.i.d. data-streams. To tackle this issue, researchers and practitioners design CL methods, i.e., strategies that make machine learning models retain high accuracy on previously seen data when trained on an ever-changing input

distribution [109, 151]. While many distinct strategies have been applied for this purpose, CL approaches can be broadly categorized into two families: *structuring* or *generalization*.

Structuring approaches to Continual Learning

Methods in the first class aim at making interference between distinct concepts less likely by endowing the stored knowledge with a disentangled structure. [113] first pioneered the idea to reduce forgetting by orthogonalizing feature representations. A similar approach was recently taken by [33]. Alternatively, *structuring* can be pursued at an architectural level, by explicitly allocating distinct subsets of model parameters to distinct tasks [90, 130, 177], encouraging non-overlapping activation patterns for different data [160, 182], or by simply applying dropout [69, 139]. Finally, several approaches regularize back-propagation by projecting the gradient to minimize the interference between tasks learned at different times [34, 59, 127]. While *structuring* approaches are usually characterized by a simpler training procedure, they typically require the availability of a task-identifier at test time.

Generalization approaches to Continual Learning

At the opposite end of the spectrum, *generalization* methods prevent forgetting by encouraging the model to compare and contrast input data encountered throughout the sequence, thus recovering the i.i.d. property of training [135]. Most notably, rehearsal-based approaches do so by maintaining a working memory of previously seen examples and interleaving them with the input data [6, 8, 23, 36, 163], while pseudo-rehearsal methods approximate this procedure with a generative model [168, 186]. Other works prioritize the learning of high-level representations either by adopting learning objectives designed not to disrupt the performance on previous tasks [78, 119, 164, 214], or by making use of semi-supervised learning techniques to learn general features [28, 100, 157]. The *generalization* approach is taken to an extreme by [159], which shows satisfactory results on CL benchmarks by training a model in an i.i.d. fashion on samples gathered greedily from the input stream.

Generalization strategies naturally blend knowledge gathered at different times to build a unified predictor, making them more reliable than their *structuring* counterparts in the realistic settings where no task-identifier is given at testing-time [60, 200]. The proposed approach aligns with the former group of methods; indeed, we argue that generalization should be extended beyond already-seen data and embrace yet-unseen knowledge as well.

Table 2.1: **Accuracy on Split-CIFAR-10 for several Continual Learning methods.** Best results in bold, second-best in italic.

	Buffer	ER [167]	GEM [127]	A-GEM [34]	iCaRL [164]	FDR [15]	GSS [8]	HAL [32]	DER [23]	DER++ [23]	**Ours**
Class-IL	50	32.69 ± 0.39	22.10 ± 0.41	20.02 ± 0.08	*55.51 ± 1.64*	28.32 ± 4.51	26.62 ± 1.36	25.26 ± 1.73	44.85 ± 2.71	49.28 ± 3.16	**56.33 ± 0.95**
	200	44.79 ± 1.86	25.54 ± 0.76	20.04 ± 0.34	49.02 ± 3.20	30.91 ± 2.74	39.07 ± 5.59	32.36 ± 2.70	61.93 ± 1.79	*64.88 ± 1.17*	**70.86 ± 0.95**
	500	57.74 ± 0.27	26.20 ± 1.26	22.67 ± 0.57	47.55 ± 3.95	28.71 ± 3.23	49.73 ± 4.78	41.79 ± 4.46	70.51 ± 1.67	*72.70 ± 1.36*	**75.07 ± 0.41**
	5120	82.47 ± 0.52	25.56 ± 3.46	21.99 ± 2.29	55.07 ± 1.55	19.70 ± 0.07	67.27 ± 4.27	59.12 ± 4.41	83.81 ± 0.33	**85.24 ± 0.49**	*84.56 ± 0.55*
	Buffer	ER [167]	GEM [127]	A-GEM [34]	iCaRL [164]	FDR [15]	GSS [8]	HAL [32]	DER [23]	DER++ [23]	**Ours**
Task-IL	50	86.98 ± 1.19	81.36 ± 1.43	81.09 ± 1.88	*88.86* ± 2.51	85.23 ± 1.24	85.22 ± 1.03	78.73 ± 3.16	85.04 ± 1.17	86.14 ± 2.56	**89.57 ± 2.47**
	200	91.19 ± 0.94	90.44 ± 0.94	83.88 ± 1.49	88.99 ± 2.13	91.01 ± 0.68	88.80 ± 2.89	82.51 ± 3.20	91.40 ± 0.92	*91.92 ± 0.60*	**93.30 ± 0.64**
	500	93.61 ± 0.27	92.16 ± 0.69	89.48 ± 1.45	88.22 ± 2.62	93.29 ± 0.59	91.02 ± 1.57	84.54 ± 2.36	93.40 ± 0.39	**93.88 ± 0.50**	*93.62 ± 0.58*
	5120	**96.98 ± 0.17**	95.55 ± 0.02	90.10 ± 2.09	92.23 ± 0.84	94.32 ± 0.97	94.19 ± 1.15	88.51 ± 3.32	95.43 ± 0.33	*96.12 ± 0.21*	95.84 ± 0.42

2.1.3 Method

Most CL methods use current and, if the method implies a rehearsal strategy, past task classification heads during the training of the current task. Future heads, that will be mapped to classes from following tasks, are not involved in the process at all. This poses a potentially dangerous situation due to the model minimizing its prediction scores for future heads, which results in a high loss peak when these heads are used at the beginning of future tasks.

We propose to leverage an auxiliary data stream, not correlated with the main task stream, in order to keep these future heads activated since the beginning of training. The proposed strategy is also beneficial to learn more distinguishing and reusable features, as the model cannot focus on simply discriminating between the classes from the task at hand. Furthermore, since auxiliary training leads future task heads to learn to recognize their own specific patterns, we exploit this property to devise a "most activated heads" (MAH) assignment strategy for future classes, that minimizes the loss peak that the model typically incurs at the beginning of a new task. Hence, the use of an auxiliary stream favors the current task and improves forward transfer to future tasks. The proposed approach is illustrated in Fig. 2.1.

Formally, a typical CL classification problem requires solving several tasks sequentially, where each task T_t, with $t \in \{1, ..., T\}$ and T being the number of tasks, consists in learning to classify a set of classes $\mathbf{C}_t$. In this work, we follow the common Class-IL and Task-IL settings [200], which assume no overlap between classes from different tasks.

Each task is associated with an i.i.d. distribution D_t of $(\mathbf{x}, y)$ pairs of a data point with the corresponding class label from $\mathbf{C}_t$. In practice, the distribution is approximated by a finite set of samples, i.e., $\mathbf{D}_t = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_{N_t}, y_{N_t})\}$, where N_t is the number of examples for task t.

The objective of CL is to find a function f_θ, depending on a set of learnable parameters

θ, that minimizes a classification objective over the entire task sequence, such as:

$$\arg\min_{\theta} \sum_{t=1}^{T} \sum_{(\mathbf{x}_i, y_i) \in \mathbf{D}_t} \mathcal{L}_{\mathrm{C}} \left(f_\theta(\mathbf{x}_i), y_i \right), \tag{2.1}$$

where $\mathcal{L}_{\mathrm{C}}$ is the classification loss (e.g., cross-entropy).

While training for the current task, most recent CL approaches [23, 36, 102, 119] attempt to reduce forgetting by adding an additional loss term that attempts to retain accuracy on previously-seen tasks. The in-task objective at task t then becomes:

$$\arg\min_{\theta} \sum_{(\mathbf{x}_i, y_i) \in \mathbf{D}_t} \mathcal{L}_{\mathrm{C}} \left(f_\theta(\mathbf{x}_i), y_i \right) + \mathcal{L}_{\mathrm{CL}}, \tag{2.2}$$

where $\mathcal{L}_{CL}$ is a generic additional loss term that implements countermeasures to catastrophic forgetting (e.g., experience replay from a buffer of samples from previous tasks).

In the proposed scenario, an additional distribution of i.i.d. auxiliary data A, where $A \neq D_t \ \forall t$, is available to the model at training time. Again, the distribution is represented by a set of sample/label pairs $\mathbf{A} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_{N_t}, y_{N_t})\}$, where labels belong to the class set $\mathbf{C}_A$.

In the following, we explain the two key aspects of the proposed approach: head pre-activation and "most activated heads" class mapping.

Head pre-activation

To ensure that the model employs all of its classification heads from the start, we use classes from the auxiliary dataset as "place-holders" for classes from future tasks.

The basic requirement of an auxiliary dataset $\mathbf{A}$ is related to the cardinality of its set of classes, $|\mathbf{C}_A|$, which should satisfy the following condition:

$$|\mathbf{C}_A| \geq \sum_{t=2}^{T} |\mathbf{C}_t| . \tag{2.3}$$

In other words, the number of auxiliary classes should be at least equal to the total number of classes in the sequence of continual learning tasks, minus the number of classes from the first task. This guarantees that, when training on the first task, the auxiliary dataset provides enough classes for the classification heads reserved to future tasks.

Before starting to train on the first task $t = 1$, we randomly choose a subset $\mathbf{C}_{A,t} \subseteq \mathbf{C}_A$, with cardinality $|\mathbf{C}_{A,t}| = \sum_{t=2}^{T} |\mathbf{C}_t|$. Samples from the selected classes are included in the

auxiliary sub-dataset $\mathbf{A}_t$ and class indexes from $\mathbf{C}_{A,t}$ are re-mapped to the indexes of classes in $\cup_{t=2}^{T}\mathbf{C}_t$ corresponding to future tasks.

At task t, we merge the corresponding dataset $\mathbf{D}_t$ and the auxiliary sub-dataset $\mathbf{A}_t$ and train on the joint set of classes, in order to minimize the following new in-task objective:

$$\arg\min_{\theta} \sum_{(\mathbf{x}_i,y_i)\in\mathbf{D}_t\cup\mathbf{A}_t} \mathcal{L}_{\mathrm{C}}\left(f_\theta(\mathbf{x}_i), y_i\right) + \mathcal{L}_{\mathrm{CL}}. \tag{2.4}$$

As a result, we ensure that all classification heads are employed, reducing the risk of loss peaks on new tasks, and encourage the model to learn more complex, discriminative and stable features.

"Most activated heads" (MAH) class mapping

At the beginning of each task $t > 1$, it is necessary to update the set of auxiliary classes in $\mathbf{A}_t$, since $|\mathbf{C}_t|$ classes must be removed to make room for classes from the new task.

Moreover, in this scenario, it also makes sense to *assign* the specific heads that will correspond to classes in the new task, rather than simply associating them to the next available heads. An appropriate class mapping can make better (re)use of features learned by the model for classification of auxiliary classes, and reduce high losses that may lead to forgetting previously-learned features.

Our head assignment approach, named MAH from "most activated heads", acts before beginning to train on task $t > 1$, by first computing the average logits $\mathbf{l}_c$, i.e., pre-softmax head activations, for each task class $c \in \mathbf{C}_t$: to this aim, we select the subset $\mathbf{D}_{t,c} \subset \mathbf{D}_t$ which only contains elements of class c, and average the corresponding logit vectors as returned by model f_θ:

$$\mathbf{l}_c = \frac{1}{N_{t,c}} \sum_{(\mathbf{x}_i,y_i)\in\mathbf{D}_{t,c}} f_\theta\left(\mathbf{x}_i\right), \tag{2.5}$$

where $N_{t,c}$ is the number of elements of class c in $\mathbf{D}_t$.

Then, each new class in $\mathbf{C}_t$ is simply associated to the classification head that maximizes its predicted score, i.e., $\arg\max \mathbf{l}_c$. In case of index collisions, largest values are given priority. Finally, the new auxiliary sub-dataset $\mathbf{A}_t$ is updated by removing from $\mathbf{A}_{t-1}$ the set of classes corresponding to selected indexes, i.e., $\{\arg\max \mathbf{l}_c\}_{c\in\mathbf{C}_t}$, and training proceeds as previously described.

2.1.4 Experimental Results

Datasets

We focus our experiments on two common evaluation protocols [200]: class-incremental (Class-IL), where the model is asked to gradually solve the complete problem but classes become available at different times; task-incremental (Task-IL), where the model is guided by the task-identity and can only focus to solve each task independently. Specifically, we leverage **Split-CIFAR-10** [229], a widely-used image classification dataset obtained by splitting the 32×32 images of CIFAR-10 into 5 binary tasks. For a more comprehensive evaluation, we also test on the larger 64×64 **Split-Micro-ImageNet**: a novel benchmark composed of a 20-class subset of Tiny-ImageNet [194], split into 5 tasks of 4 classes each.

As for the choice of the auxiliary data, we pair the original data with similarly-sized datasets. In particular, the auxiliary dataset for Split-CIFAR-10 consists of a subset of 10 super-classes from CIFAR-100, selected among those which are not semantic-related to those contained in CIFAR-10. For Split-Micro-ImageNet, we select a subset of 20 classes from ILSVRC-2012, making sure that the chosen data is as unrelated as possible with the original Tiny-ImageNet classes. In detail, we first remove Tiny-ImageNet classes from the entire label set; then, we group the remaining 800 classes into 293 super-classes, corresponding to synsets found at distance 8 from the *entity* root node. Finally, we apply Spectral Clustering to select the 20 classes which are most representative of the super-classes.

Training procedure

We apply the approach described in Section 2.1.3 by adapting the DER++ [23] method, a recent rehearsal-based approach inspired by knowledge distillation principles. For a fair comparison among different models, in our experiments we follow [23] and adopt the same training settings. As backbone, we use ResNet-18 [73] (not pretrained). We optimize our model with SGD, for 50 epochs on Split-CIFAR-10 and 100 on Split-Micro-ImageNet. During training, samples from the current task and from auxiliary classes are combined so that each mini-batch contains data from both domains. We apply random crops and horizontal flips as data augmentation. All hyperparameters are as defined in [23].

Results

To validate the effectiveness of our approach using auxiliary data during training, we compare our method with other CL methods based on rehearsal strategies: ER [167], GEM [127], A-GEM [34], iCaRL [164], FDR [15], GSS [8], HAL [32], DER [23] and vanilla DER++ [23].

Performance for these methods is reported from [23], except for the setup with buffer size equal to 50[1].

As performance metrics, we report classification accuracy in the Class-IL setting as the average of per-task classification accuracy computed after the final task, and in the Task-IL setting as the average of per-task classification accuracy computed at the end of each task. Table 2.1 and Table 2.2 report results on Split-CIFAR-10 and Split-Micro-ImageNet, respec-

Table 2.2: **Accuracy on Split-Micro-ImageNet for Experience Replay-based methods.**

Buffer	**Method**	*Class-IL*	*Task-IL*
50	ER [167]	22.58 ± 0.71	66.28 ± 1.83
	DER [23]	29.35 ± 2.16	70.88 ± 0.83
	DER++ [23]	31.92 ± 2.15	69.86 ± 1.96
	Ours	**37.24 ± 2.76**	**71.84 ± 1.82**
200	ER [167]	36.22 ± 1.06	77.82 ± 1.24
	DER [23]	46.18 ± 1.44	81.12 ± 1.59
	DER++ [23]	51.84 ± 1.32	**82.66 ± 1.60**
	Ours	**52.83 ± 0.83**	78.48 ± 1.42
500	ER [167]	49.70 ± 0.71	84.35 ± 0.79
	DER [23]	56.58 ± 2.44	84.56 ± 1.19
	DER++ [23]	**60.28 ± 2.31**	**85.10 ± 0.93**
	Ours	59.36 ± 0.81	79.95 ± 0.61
5120	ER [167]	70.40 ± 1.30	89.20 ± 0.01
	DER [23]	68.75 ± 0.25	89.35 ± 0.85
	DER++ [23]	**74.98 ± 0.66**	**90.72 ± 0.65**
	Ours	71.65 ± 0.82	85.93 ± 1.29

tively. Our method yields the best Class-IL performance when tested with small/medium buffer size. It is also noteworthy that as the performance gain of our approach increases as the size of the buffer decreases. When buffer size becomes significantly larger, vanilla DER++ still achieves the best results, showing that retaining and replaying enough data (5,120 samples represent more than 10% of the entire training set of CIFAR-10) is still the best option to alleviate catastrophic forgetting, although this goes in stark contrast to generalizing continual learning methods to real-world problems. A similar behavior can be observed on the simpler Task-IL setting, where our method obtains the highest performance or is on par with existing methods under low–data availability regimes.

[1]In this case, results were computed using the Mammoth framework for PyTorch: https://github.com/aimagelab/mammoth

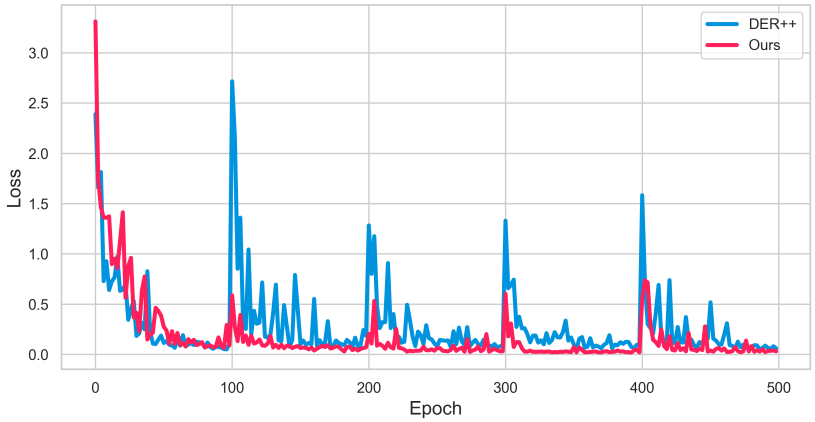

Figure 2.2: **Training loss trend for our approach (red) and DER++ (blue) on Split-Micro-ImageNet.** When the model switches to a new task, MAH reduces loss peaks by assigning new classes to the most suitable available heads trained on auxiliary data.

We also compare our approach to Co^{2}L [28], that recently achieved state-the-art performance in both settings[2]. Nevertheless, on Split-CIFAR-10, our method yields better Class-IL performance than Co^{2}L [28], that reaches 65.57±1.37 with buffer size of 200 and 74.26±0.77 with buffer size of 500, respectively compared 70.86±0.95 and 75.07±0.41 by our method. The lower standard deviation also shows that our approach tends to be more stable across tasks.

Finally, we monitor the loss over consecutive tasks in order to evaluate the impact of auxiliary data during training. Fig. 2.2 shows the average training loss for vanilla DER++ and our method on Split-Micro-ImageNet. It can be observed that, as new tasks come in (every 100 epochs), the proposed approach shows a smoother loss surface, conversely to the vanilla counterpart that, instead, exhibits more noticeable peaks. Thus, the proposed strategy also improves forward transfer and prevents disrupting gradient peaks when the model switches to new tasks, resembling non-continual learning scenarios.

Ablation study

In order to substantiate our design choices, we perform an ablation study to quantify the contribution of a) using the auxiliary data stream and b) the MAH strategy. The obtained results are reported in Table 2.3 and compared to vanilla DER++ and DER++ with auxiliary data but **without MAH**. When the MAH strategy is not used, the classification heads are selected in a sequential order without making use of neural activation mapping between past (auxiliary) and current classes.

[2]Co^{2}L is not reported in Tables 2.1 and 2.2, as its training strategies are significantly different from the methods shown in those tables.

Results show that training with auxiliary data yields significant performance gains for all buffer sizes but 5,120, where replayed knowledge becomes prevalent. In all cases, MAH outperforms sequential head mapping.

Table 2.3: **Ablation Study.** Results obtained by the vanilla DER++ (first row of each block), DER++ with auxiliary data (second row), and the proposed method (third row), combining DER++ with auxiliary data and MAH strategy, for different buffer sizes.

Buffer	**Method**	*Class-IL*	*Task-IL*
50	DER++	49.28 ± 3.16	86.14 ± 2.56
	↪ + AUX	52.74 ± 1.02	88.51 ± 2.01
	↪ + MAH	**56.33 ± 0.95**	**89.57 ± 2.47**
200	DER++	64.88 ± 1.17	91.92 ± 0.60
	↪ + AUX	69.91 ± 1.48	92.57 ± 1.02
	↪ + MAH	**70.86 ± 0.95**	**93.30 ± 0.64**
500	DER++	72.70 ± 1.36	93.88 ± 0.50
	↪ + AUX	74.24 ± 0.61	**93.93 ± 0.47**
	↪ + MAH	**75.07 ± 0.41**	93.62 ± 0.58
5120	DER++	**85.24 ± 0.49**	**96.12 ± 0.21**
	↪ + AUX	84.26 ± 0.22	95.58 ± 0.22
	↪ + MAH	84.56 ± 0.55	95.84 ± 0.42

Effect of Pre-training

We further investigate whether it is better to employ a backbone pre-trained on auxiliary data or to train it from scratch using the proposed strategy. The results of this analysis are reported in Table 2.4: pre-training on auxiliary data appears to be always beneficial compared to training from scratch in DER++ and as the buffer size increases. On the contrary, on small buffers, using auxiliary data with our approach on a model trained from scratch yields better performance than pre-training. Furthermore, with our method, pre-training on auxiliary data leads instead to lower performance than training from scratch, showing that pre-training is not always a reliable alternative to continuously training with auxiliary data.

Generative Auxiliary Model

In the previous sections, we have consistently observed that using auxiliary data helps retaining knowledge of previous tasks, especially with limited buffer size. However, it is not

Table 2.4: **Effect of Pre-training.** Results obtained by the vanilla DER++ (first row of each block), DER++ pre-trained with auxiliary data (second row), the proposed method trained from scratch (third row), and the proposed model pre-trained with auxiliary data (fourth row), for different buffer sizes.

Buffer	**Method**	*Class-IL*	*Task-IL*
50	DER++	49.28 ± 3.16	86.14 ± 2.56
	DER++ with pre-training	50.82 ± 3.34	84.02 ± 2.98
	Ours	**56.33 ± 0.95**	**89.57 ± 2.47**
	Ours with pre-training	53.52 ± 3.60	89.52 ± 0.88
200	DER++	64.88 ± 1.17	91.92 ± 0.60
	DER++ with pre-training	68.71 ± 1.01	92.43 ± 0.53
	Ours	**70.86 ± 0.95**	**93.30 ± 0.64**
	Ours with pre-training	65.17 ± 2.67	91.35 ± 1.71
500	DER++	72.70 ± 1.36	93.88 ± 0.50
	DER++ with pre-training	**75.91 ± 0.26**	**94.39 ± 0.29**
	Ours	75.07 ± 0.41	93.62 ± 0.58
	Ours with pre-training	71.39 ± 2.77	91.77 ± 0.76
5120	DER++	85.24 ± 0.49	96.12 ± 0.21
	DER++ with pre-training	**86.60 ± 0.42**	**96.29 ± 0.09**
	Ours	84.56 ± 0.55	95.84 ± 0.42
	Ours with pre-training	82.20 ± 1.37	94.65 ± 0.36

efficient to maintain the auxiliary data in memory, as in that case it is still preferable to simply use a larger buffer.

A viable alternative would be to replace the auxiliary stream with a generative replay model and use generated samples during task training. In order to investigate the feasibility of this option, we use a generative adversarial network (GAN) [68] to learn the distribution of auxiliary data, and employ synthetic images in place of real ones in Eq. 2.4 in our method.

We then replicate the experiments carried out on Split-CIFAR-10 by pre-training a BigGAN model [21] on super-classes of CIFAR-100 used as auxiliary data, and compare the results with those obtained when using real images. As it can be seen in Table 2.5, performance achieved when using generated images follow the same behavior observed with real data, i.e., performance increase in settings with small buffer size, while the approach is less beneficial when the replay memory increases.

Table 2.5: **Effect of replacing auxiliary data with a GAN.** Results obtained on Split-CIFAR-10 when using no auxiliary data (vanilla DER++), real auxiliary data (the proposed method) and synthetic data.

Buffer	**Aux. data**	*Class-IL*	*Task-IL*
50	*none*	49.28 ± 3.16	86.14 ± 2.56
	real	56.33 ± 0.95	89.57 ± 2.47
	synthetic	54.47 ± 3.05	89.23 ± 1.83
200	*none*	64.88 ± 1.17	91.92 ± 0.60
	real	70.86 ± 0.95	93.30 ± 0.64
	synthetic	68.84 ± 0.77	92.85 ± 0.24
500	*none*	72.70 ± 1.36	93.88 ± 0.50
	real	75.07 ± 0.41	93.62 ± 0.58
	synthetic	74.35 ± 1.04	93.42 ± 0.40
5120	*none*	85.24 ± 0.49	96.12 ± 0.21
	real	84.56 ± 0.55	95.84 ± 0.42
	synthetic	84.14 ± 0.45	95.65 ± 0.23

2.1.5 Discussion

This section proposed a novel approach for improving continual learning accuracy by leveraging external data. Our experiments show that providing the model with an additional auxiliary stream leads to an increase in performance, especially when the employed memory buffer is small, and to more stable training at the beginning of each task. We also observe that our approach works consistently better, on small buffer settings, than alternative knowledge transfer strategies such as direct pre-training on auxiliary data, even though an effective solution to leverage pretraining in continual learning is investigated in the subsequent section. The approximation of the auxiliary data distribution through the use of generative models also outperforms state-of-the-art models, thus indicating the future direction of this work, i.e., a more effective modeling of previous real-world knowledge as it seems to happen in the human hippocampus [174] which will be further explored in last section of this chapter.

2.1.6 Publications

Bellitto, G., Pennisi, M., Palazzo, S., Bonicelli, L., Boschini, M., & Calderara, S. (2022, August). Effects of auxiliary knowledge on continual learning. In 2022 26th International Conference on Pattern Recognition (ICPR) (pp. 1357-1363). IEEE. [11]

2.2 Transfer without forgetting

2.2.1 Motivation

Thanks to the enthusiastic development carried out by the scientific community, there exist myriad widely available deep learning models that can be either readily deployed or easily adapted to perform complex tasks [9, 72, 158, 189, 203]. However, the desiderata of practical applications [183] often overstep the boundaries of the typical *i.i.d.* paradigm, fostering the study of different learning approaches.

In contrast with the natural tendency of biological intelligence to seamlessly acquire new skills and notions, deep models are prone to an issue known as *catastrophic forgetting* [133], *i.e.*, they fit the current input data distribution to the detriment of previously acquired knowledge. In light of this limitation, the sub-field of Continual Learning (CL) [45, 151, 200] aspires to train models capable of adaptation and lifelong learning when facing a sequence of changing tasks, either through appositely designed architectures [130, 177, 180], targeted regularization [102,119,229] or by storing and replaying previous data points [23,36,164,167].

On a similar note, human intelligence is especially versatile in that it excels in contrasting and incorporating knowledge coming from multiple domains. Instead, the application of deep supervised learning algorithms typically demands large annotated datasets, whose collection has significant costs and may be impractical. To address this issue, Transfer Learning (TL) techniques are typically applied with the purpose of transferring and re-using knowledge across different data domains. In this setting, the simplest technique is to pretrain the model on a huge labeled dataset (*i.e.* the source) and then finetune it on the *target* task [51, 71, 165]. Such a simple schema has been recently overcome by more sophisticated domain adaptation algorithms [38, 125, 126] mainly based on the concept of *feature alignment*: here, the goal is to reduce the shift between the feature distributions of target and source domains. Unfortunately, these approaches often require the availability of the source dataset during training, which clashes with the usual constraints imposed in the CL scenarios.

In this work, we explore the interactions between pretraining and CL and highlight a blind spot of continual learners. Previous work underlined that naive pretraining is beneficial as it leads the learner to reduced forgetting [137]. However, we detect that the pretraining task itself is swiftly and catastrophically forgotten as the model veers towards the newly introduced stream of data. This matter is not really detrimental if all target classes are available at once (*i.e.*, joint training): as their exemplars can be accessed simultaneously, the learner can discover a joint feature alignment that works well for all of them while leaving its pretraining initialization. However, if classes are shown in a sequential manner, we argue that

transfer mostly concerns the early encountered tasks: as a consequence, pretraining ends up being fully beneficial only for the former classes. For the later ones, since pretraining features are swiftly overwritten, the benefit of pretraining is instead lowered, thus undermining the advantages of the source knowledge. In support of this argument, this work reports several experimental analyses (Sec. 2.2.3) revealing that state-of-the-art CL methods do not take full advantage of pretraining knowledge.

To account for such a disparity and let all tasks profit equally from pretraining, this work sets up a framework based on Transfer Learning techniques. We show that the Continual Learning setting requires specific and *ad-hoc* strategies to fully exploit the source knowledge without incurring its forgetting. Consequently, we propose an approach termed **Transfer without Forgetting (TwF)** that equips the base model with a pretrained and fixed sibling network, which continuously propagates its internal representations to the former network through a per-layer strategy based on *knowledge distillation* [77]. We show that our proposal is more effective than alternative approaches (*i.e.*, extending anti-forgetting regularization to the pretraining initialization) and beneficial even if the data used for pretraining is strongly dissimilar w.r.t. to the target task.

2.2.2 Related Work

Continual Learning (CL) [45, 151] is an increasingly popular field of machine learning that deals with the mitigation of catastrophic forgetting [133]. CL methods are usually grouped as follows, according to the approach they take.

Regularization-based methods [30, 35, 102, 127] typically identify subsets of weights that are highly functional for the representations of previous tasks, with the purpose to prevent their drastic modification through apposite optimization constraints. Alternatively, they consolidate the previous knowledge by using past models as soft teachers while learning the current task [119].

Architectural approaches dedicate distinct sets of parameters to each task, often resorting to network expansion as new tasks arrive [130, 177, 182]. While capable of high performance, they are mostly limited to the Task-IL scenario (described in Sec. 2.2.4) as they require task-identifiers at inference time.

Rehearsal-based methods employ a fixed-size buffer to store a fraction of the old data. ER [163, 169] interleaves training samples from the current task with previous samples: notably, several works [23, 60] point out that such a simple strategy can effectively mitigate forgetting and achieve superior performance. This method has hence inspired several works: DER [23] and its extension X-DER [18] also store past model responses and pin them as

an additional teaching signal. MER [167] combines replay and meta-learning [62, 146] to maximize transfer from the past while minimizing interference. Other works [8, 24] propose different sample-selection strategies to include in the buffer, while GEM [127] and its relaxation A-GEM [35] employ old training data to minimize interference. On a final note, recent works [20, 191] exploit the memory buffer to address semi-supervised settings where examples can be either labeled or not.

Transfer Learning (TL) [149] is a machine learning methodology aiming at using the knowledge acquired on a prior task to solve a distinct target task. In its classical formulation [224], a model is trained on the source dataset and then finetuned on the (possibly much smaller) target dataset to adapt the previously learned features. Alternatively, transfer can be induced via multi-level Knowledge Distillation, guided by meta-learning [86], attention [207] or higher-level descriptions of the flow of information within the model [221].

2.2.3 Method

Setting

In CL, a classification model $f_{(\theta,\phi)}$ (composed of a multilayered feature extractor $h_\theta = h^{(l)}_{\theta_l} \circ h^{(l-1)}_{\theta_{l-1}} \circ \cdots \circ h^{(1)}_{\theta_1}$ and a classifier g_ϕ, $f_{(\theta,\phi)} = g_\phi \circ h_\theta$) is trained on a sequence of N tasks $\mathcal{T}_i = \{(x^i_j, y^i_j)\}_{j=1}^{|\mathcal{T}_i|}$. The objective of $f_{(\theta,\phi)}$ is minimizing the classification error across all seen tasks:

$$\min_{\theta,\phi} \mathcal{L} = \mathop{\mathbb{E}}_{i} \left[\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}_i} \Big[\ell(y, f_{(\theta,\phi)}(x)) \Big] \right], \tag{2.6}$$

where ℓ is a suitable loss function. Unfortunately, the problem framed by Eq. 2.6 cannot be directly optimized due to the following key assumptions: *i)* while learning the current task $\mathcal{T}_c$, examples and labels of previous tasks are inaccessible; *ii)* the label space of distinct tasks is disjoint ($y^i_m \neq y^j_n \; \forall i \neq j$) *i.e.*, classes learned previously cannot recur in later phases. Therefore, Eq. 2.6 can only be approximated, seeking adequate performance on previously seen tasks (*stability*), while remaining flexible enough to adapt to upcoming data (*plasticity*).

Pretraining incurs Catastrophic Forgetting

Mehta et al. [137] have investigated the entanglement between continual learning and pretraining, highlighting that the latter leads the optimization towards wider minima of the loss landscape. As deeply discussed in [18, 23], such property is strictly linked to a reduced tendency in incurring forgetting.

On this latter point, we therefore provide an alternate experimental proof of the benefits deriving from pretraining initialization. In particular, we focus on ResNet-18 trained with

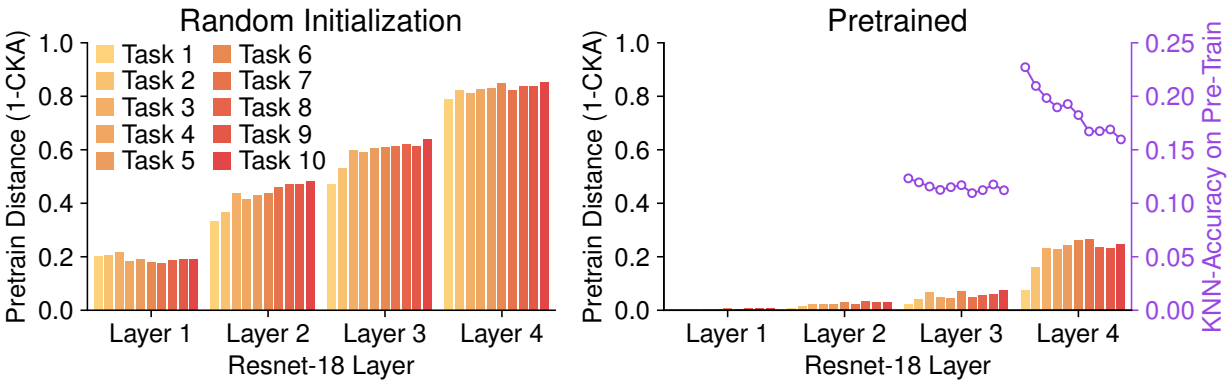

Figure 2.3: **Forgetting of the initialization, measured as the distance from the pretrain (1−CKA [104]) (lower is better) and kNN accuracy (higher is better).** Features extracted by a pretrained model remain closer to the initialization w.r.t. a randomly initialized model. Furthermore, the steady decrease in kNN accuracy as training progresses reveals that features become less specific for past tasks.

ER [169] on Split CIFAR-100[3] and measure how each individual layer differs from its initialization. It can be observed that a randomly initialized backbone (Fig. 2.3, *left*) significantly alters its parameters at all layers while tasks progress, resulting in a very low Centered Kernel Alignment [104] similarity score already at the first CL task. On the contrary, a backbone pretrained on Tiny ImageNet (Fig. 2.3, *right*) undergoes limited parameter variations in its layers, with the exception of the last residual layer (although to a lesser extent w.r.t. random init.). This latter finding indicates that its pretraining parametrization requires relevant modifications to fit the current training data. This leads to the *catastrophic forgetting* of the source pretraining task: namely, the latter is swiftly forgotten as the network focuses on the initial CL tasks. This is corroborated by the decreasing accuracy for pretraining data of a kNN classifier trained on top of *Layer 3* and *Layer 4* representations in Fig. 2.3 (*right*).

To sum up, while pretraining is certainly beneficial, the model drifts away from it one task after the other. Hence, only the first task takes full advantage of it; the optimization of later tasks, instead, starts from an initialization that increasingly differs from the one attained by pretraining. This is detrimental, as classes introduced later might be likewise advantaged by the reuse of different pieces of the initial knowledge.

Transfer without Forgetting

To mitigate the issue above, we propose a strategy that enables a continuous transfer between the source task and the incrementally learned target problem.

[3]This preliminary experiment follows the same setting presented in Sec. 2.2.4

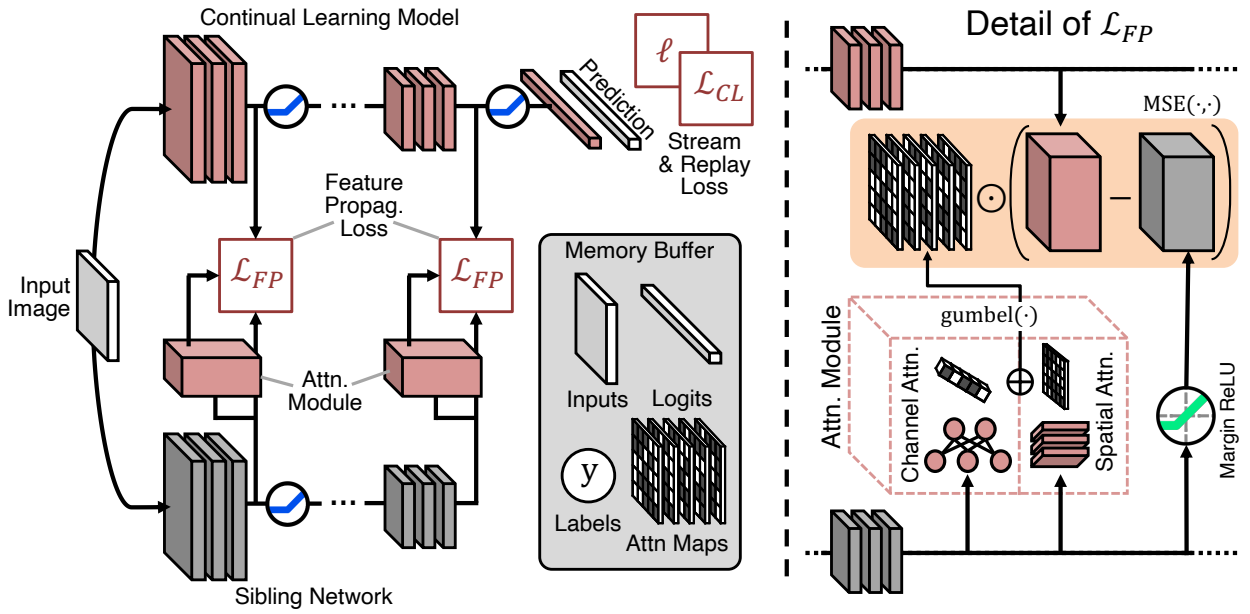

Figure 2.4: **Overview of TwF and detail of $\mathcal{L}_{FP}$:** Given a batch of samples from the current task or from $\mathcal{B}$, we *i)* extract intermediate features from both the student and fixed sibling backbones at multiple layers; *ii)* compute the corresponding binarized attention maps $\mathbb{M}(\cdot)$; *iii)* pull the attention-masked representations of the two models closer.

Feature Propagation

As the training progresses, the input stream introduces new classes that might benefit from the adaptation of specific features of the pretrained model. To enable feature transfer without incurring pretraining forgetting, we maintain a copy of it (the *sibling* model) and adopt an intermediate feature knowledge distillation [4, 75, 140, 173, 207] objective. Considering a subset of L layers, we seek to minimize the distance between the activations of the base network $h_\theta^{(l)} \triangleq h_\theta^{(l)}(x)$ and those from its pretrained sibling $\widehat{h}^{(l)} \triangleq h_{\theta^t}^{(l)}(x)$:

$$\mathop{\mathbb{E}}_{x\sim\mathcal{T}_c}\left[\sum_{l=1}^{L}||h_\theta^{(l)} - \mathrm{ReLU_m}(\widehat{h}^{(l)})||_2^2\right], \tag{2.7}$$

where c is the current task and $\mathrm{ReLU_m}(\cdot)$ indicates the application of a margin ReLU activation [75]. It is noted that the objective outlined by Eq. 2.7 leads the CL model to focus on mirroring the internal representations of the pretrained teacher and maximizing transfer. However, focusing on the latter solely can lead to excessive rigidity, thus preventing the model from fitting the data from the current task altogether. On these grounds, we take inspiration from [207] and use a weighted version of Eq. 2.7. In particular, an apposite learnable module computes a gating attention map $\mathbb{M}(\cdot)$ over the feature maps of the sib-

ling, which serves as a binary mask selecting which spatial regions have to be aligned. The resulting objective is consequently updated as follows:

$$\mathop{\mathbb{E}}_{x\sim\mathcal{T}_c}\left[\sum_{l=1}^{L}||\mathbb{M}(\widehat{h}^{(l)})\odot\left(h_\theta^{(l)}-\mathrm{ReLU_m}(\widehat{h}^{(l)})\right)||_2^2\right], \tag{2.8}$$

where $\odot$ indicates the Hadamard product between two tensors of the same dimensions. The attention maps $\mathbb{M}(\cdot)$ are computed through specific layers, whose architectural design follows the insights provided in [152]. Specifically, they forward the input activation maps into two parallel branches, producing respectively a Channel Attention $\mathbb{M}_{\mathrm{Ch}}(\cdot)$ map and a Spatial Attention $\mathbb{M}_{\mathrm{Sp}}(\cdot)$ map. These two intermediate results are summed and then activated through a binary Gumbel-Softmax sampling [85], which allows us to model discrete *on-off* decisions regarding which information we want to propagate. In formal terms:

$$\mathbb{M}(\widehat{h}^{(l)})\triangleq\mathrm{gumbel}(\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)})+\mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)})). \tag{2.9}$$

The Spatial Attention $\mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)})$ regulates the propagation of spatially localized information and is obtained by stacking four convolutional layers [152] with different configurations (*i.e.*, the kernel sizes and dilation rates.

$$\mathbb{M}_{\mathrm{Sp}}(\widehat{h}^{(l)})\triangleq\mathrm{C}_{1\times1}\circ\mathrm{C}_{3\times3}\circ\mathrm{C}_{3\times3}\circ\mathrm{C}_{1\times1}(\widehat{h}^{(l)}), \tag{2.10}$$

where C denotes a sequence of convolutional, batch normalization, and ReLU activation layers. On the other hand, the Channel Attention $\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)})$ estimates the information across the channels of $\widehat{h}^{(l)}$; in its design, we draw inspiration from the formulation proposed in [80]. Formally, considering the result $\widehat{h}^{(l)}_{\mathrm{GAP}}$ of the Global Average Pooling (GAP) applied on top of $\widehat{h}^{(l)}$, we have:

$$\mathbb{M}_{\mathrm{Ch}}(\widehat{h}^{(l)})\triangleq\tanh(\mathrm{BN}(W_1^{\mathsf{T}}\widehat{h}^{(l)}_{\mathrm{GAP}}))\cdot\sigma(\mathrm{BN}(W_2^{\mathsf{T}}\widehat{h}^{(l)}_{\mathrm{GAP}}))+W_3^{\mathsf{T}}\widehat{h}^{(l)}_{\mathrm{GAP}}, \tag{2.11}$$

where W_1, W_2, and W_3 are the weights of three fully connected layers organized in parallel and BN indicates the application of batch normalization.

Diversity loss

Without a specific loss term supervising the attention maps, we could incur in useless behaviors, *e.g.*, all binary gates being either on or off, or some channels being always propagated and some others not. While recent works provide a target expected activation ratio [2,182] as

a countermeasure, we encourage the auxiliary modules to assign different propagation gating masks to different examples. The intuition is that each example has its own preferred subset of channels to be forwarded from the sibling. To do so, we include an additional auxiliary loss term [141] as follows:

$$\begin{aligned} \mathcal{L}_{\text{AUX}} &\triangleq -\lambda \sum_{l=1}^{L} \mathop{\mathbb{E}}_{x_1,\ldots,x_n \sim \mathcal{T}_c} \left[\sum_{j=1}^{n} \log \frac{e^{g_{ij}^{\text{T}} g_{ij}/T}}{\frac{1}{n}\sum_{k=1}^{n} e^{g_{ij}^{\text{T}} g_{ik}/T}} \right], \\ g_{ij} &\triangleq \text{NORM}(\text{GAP}(\mathbb{M}(\widehat{h}^{(l)}(x_j)))), \end{aligned} \tag{2.12}$$

where n indicates the batch size, NORM a normalization layer, T a temperature and finally λ is a scalar weighting the contribution of this loss term to the overall objective. In practice, we ask each vector containing channel-wise average activity to have a low dot product with vectors of other examples.

Knowledge Replay

The training objective of Eq. 2.8 is devised to facilitate selective feature transfer between the in-training model and the immutable sibling. However, to prevent forgetting tied to previous CL tasks to the greatest extent, the model should also be provided with a targeted strategy. We thus equip the continual learner with a small memory buffer $\mathcal{B}$ (populated with examples from the input stream via *reservoir sampling* [205]) and adopt the simple labels and logits replay strategy proposed in [23]:

$$\mathcal{L}_{\text{CL}} \triangleq \mathop{\mathbb{E}}_{(x,y,l)\sim\mathcal{B}} \Big[\alpha \cdot ||f_{(\theta,\phi)}(x) - l||_2^2 + \beta \cdot \ell(y, f_{(\theta,\phi)}(x)) \Big], \tag{2.13}$$

where (x, y, l) is a triplet of example, label and original network responses $l = f(x)$ recorded at the time of sampling and α, β are scalar hyperparameters. Although extremely beneficial, we remark that the model need not optimize $\mathcal{L}_{\text{CL}}$ to achieve basic robustness against catastrophic forgetting (as shown in Sec. 2.2.5): preserving pretraining features already serves this purpose.

Replaying past propagation masks

With the purpose of protecting the feature propagation formulated in Eq. 2.8 from forgetting, we also extend it to replay examples stored in memory. It must be noted that doing so requires taking additional steps to prevent cross-task interference; indeed, simply applying Eq. 2.8 to replay items would apply the feature propagation procedure unchanged to all tasks, regardless of the classes thereby included. For this reason, we take an extra step and make

all batch normalization and fully connected layers in Eq. 2.9, 2.10 and 2.11 conditioned [47] w.r.t. the CL task. Consequently, we add to $\mathcal{B}$ for each example x both its task label t and its corresponding set of binary attention maps $m = (m^1, ..., m^l)$ generated at the time of sampling. Eq. 2.8 is finally updated as:

$$\begin{aligned}\mathcal{L}_{\text{FP}} \triangleq & \mathop{\mathbb{E}}_{\substack{(x,t=c)\sim\mathcal{T}_c \\ (x;t)\sim\mathcal{B}}} \left[\sum_{l=1}^{L} ||\mathbb{M}(\widehat{h}^{(l)};t) \odot \left(h^{(l)} - \text{ReLU}_{\text{m}}(\widehat{h}^{(l)}) \right)||_2^2 \right] \\ & + \mathop{\mathbb{E}}_{\substack{(x,t,m)\sim\mathcal{B} \\ l=1,...,L}} \left[\text{BCE}\left(\mathbb{M}(\widehat{h}^{(l)};t), m^{(l)} \right) \right],\end{aligned} \tag{2.14}$$

where the second term is an additional replay contribution distilling past attention maps, with BCE indicating the binary cross entropy criterion.

Overall objective

Our proposal – dubbed **Transfer without Forgetting (TwF)** – optimizes the following training objective, also summarized in Fig. 2.4:

$$\min_{\theta,\phi} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{T}_c} \left[\ell(y_j^i, f_{(\theta,\phi)}(x_j^i)) \right] + \mathcal{L}_{\text{CL}} + \mathcal{L}_{\text{FP}} + \mathcal{L}_{\text{AUX}}. \tag{2.15}$$

We remark that: *i)* while TwF requires keeping a copy of the pretrained model during training, this does not hold at inference time; *ii)* similarly, task labels t are not needed during inference but only while training, which makes TwF capable of operating under both the Task-IL and Class-IL CL settings [200]; *iii)* the addition of t and m in $\mathcal{B}$ induces a limited memory overhead: t can be obtained from the stored labels y for typical classification tasks with a fixed number of classes per task, while m is a set of Boolean maps that is robust to moderate re-scaling (as we demonstrate by storing m at half resolution for our experiments in Sec. 2.2.4). We finally point out that, as maps m take discrete binary values, one could profit from lossless compression algorithms (such as Run-Length Encoding [170] or LZ77 [237]) and thus store a compressed representation into the memory buffer. We leave the comprehensive investigation of this application to future works.

2.2.4 Experimental Results

Experimental Setting

Metrics

We assess the overall performance of the models in terms of *Final Average Accuracy* (FAA), defined as the average accuracy on all seen classes after learning the last task, and *Final Forgetting* [30] (FF), defined as:

$$\text{FF} \triangleq \frac{1}{T-1} \sum_{i=0}^{T-2} \max_{t \in \{0,\dots,T-2\}} \{a_i^t - a_i^{T-1}\}, \tag{2.16}$$

where a_i^t denotes the accuracy on task τ_i after training on the t^{th} task.

Settings

We report results on two common protocols [200]: *Task-Incremental Learning* (Task-IL), where the model must learn to classify samples only from within each task, and *Class-Incremental Learning* (Class-IL), where the model must gradually learn the overall classification problem. The former scenario is a relaxation of the latter, as it provides the model with the task identifier of each sample at test time; for this reason, we focus our evaluation mainly on the Class-IL protocol, highlighted as a more realistic and challenging benchmark [8, 60].

Datasets

We initially describe a scenario where the transfer of knowledge from the pretrain is facilitated by the similarity between the two distributions. Precisely, we use **CIFAR-100** [106] as the pretrain dataset and then evaluate the models on **Split CIFAR-10** [229] (5 binary tasks) (see Tab. 2.6). In Tab. 2.7 we envision a second and more challenging benchmark, which relies on **Split CIFAR-100** [229] with the opportunity to benefit from the knowledge previously learned on **Tiny ImageNet** [194]. Due to the size mismatch between CIFAR-100 and the samples from Tiny ImageNet, we resize the latter to 32×32 during pretraining. The last scenario (Tab. 2.8) involves pretraining on ImageNet [48] and learning incrementally **Split CUB-200** [35, 225], split into 10 tasks of 20 classes each. With an average of only 29.97 images per class and the use of higher-resolution input samples (resized to 224×224), this benchmark is the most challenging. We use ResNet18 [73] for all experiments involving Split CIFAR-10 and Split CIFAR-100, as in [23,164], while opting for ResNet50 on Split CUB-200.

Competitors

We focus our comparison on state-of-the-art rehearsal algorithms, as they prevail on most benchmarks in literature [23, 36, 200].

- **Experience Replay (ER)** [163, 169] is the first embodiment of a rehearsal strategy that features a small memory buffer containing an *i.i.d.* view of all the tasks seen so far. During training, data from the stream is complemented with data sampled from the buffer. While this represents the most straightforward use of a memory in a CL scenario, ER remains a strong baseline, albeit with a non-negligible memory footprint.
- **Dark Experience Replay (DER)** [23] envisions a self-distillation [64] constraint on data stored in the memory buffer and represents a simple extension to the basic rehearsal strategy of ER. In this work, we compare against DER++, which includes both ER and DER objectives.
- **Incremental Classifier and Representation Learning (iCaRL)** [164] tackle catastrophic forgetting by distilling the responses of the model at the previous task boundary and storing samples that better represent the current task. In addition to simple replay, those *exemplars* are used to compute class-mean prototypes for nearest-neighbor classification.
- **ER with Asymmetric Cross-Entropy (ER-ACE)** [26] recently introduced a method to alleviate class imbalances to ER. The authors obtain a major gain in accuracy by simply separating the cross-entropy contribution of the classes in the current batch and that of the ones in the memory buffer.
- **Contrastive Continual Learning (CO^2L)** [28] proposes to facilitate knowledge transfer from samples stored in the buffer by optimizing a contrastive learning objective, avoiding any potential bias introduced by a cross-entropy objective. To perform classification, a linear classifier needs to be first trained on the exemplars stored in the buffer.

In addition, we also include results from two popular regularization methods. **Online Elastic Weight Consolidation (oEWC)** [102] penalizes changes on the most important parameters by means of an online estimate of the Fisher Information Matrix evaluated at task boundaries. **Learning without Forgetting (LwF)** [119] includes a distillation target similar to iCaRL but does not store any exemplars. We remark that **all competitors undergo an initial pretraining phase** prior to CL, thus ensuring a fair comparison.

Table 2.6: **Final Average Accuracy (FAA) [↑] and Final Forgetting (FF) [↓] on Split CIFAR-10 w. pretrain on CIFAR-100.**

FAA (FF)	**Split CIFAR-10** (*pretr. CIFAR-100*)			
Method	**Class-IL**		**Task-IL**	
Joint (UB)	92.89 (–)		98.38 (–)	
Finetune	19.76 (98.11)		84.05 (17.75)	
oEwC [180]	26.10 (88.85)		81.84 (19.50)	
LwF [119]	19.80 (97.96)		86.41 (14.35)	
Buffer Size	500	5120	500	5120
ER [169]	67.24 (38.24)	86.27 (13.68)	96.27 (2.23)	97.89 (0.55)
CO^2L [28]	75.47 (21.80)	87.59 (9.61)	96.77 (1.23)	97.82 (0.53)
iCaRL [164]	76.73 (14.70)	77.95 (12.90)	97.25 (0.74)	97.52 (0.15)
DER++ [23]	78.42 (20.18)	87.88 (8.02)	94.25 (4.46)	96.42 (1.99)
ER-ACE [26]	77.83 (10.63)	86.20 (5.58)	96.41 (2.11)	97.60 (0.66)
TwF (ours)	**83.65** (**11.59**)	**89.55** (**6.85**)	**97.49** (**0.86**)	**98.35** (**0.17**)

To gain a clearer understanding of the results, all the experiments include the performance of the upper bound (**Joint**), obtained by jointly training on all classes in a non-continual fashion. We also report the results of the model obtained by training sequentially on each task (**Finetune**), *i.e.*, without any countermeasure to forgetting.

Comparison with State-Of-The-Art

Regularization methods

Across the board, non-rehearsal methods (oEWC and LwF) manifest a profound inability to effectively use the features learned during the pretrain. As those methods are not designed to extract and reuse any useful features from the initialization, the latter is rapidly forgotten, thus negating any knowledge transfer in later tasks. This is particularly true for oEWC, whose objective proves to be both too strict to effectively learn the current task and insufficient to retain the initialization. Most notably, on Split CUB-200 oEWC shows performance lower than Finetune on both Task- and Class-IL.

Rehearsal methods

In contrast, rehearsal models that feature some form of distillation (DER++ and iCaRL) manage to be competitive on all benchmarks. In particular, iCaRL proves especially effective on Split CIFAR-100, where it reaches the second highest FAA even when equipped with a

Table 2.7: **Accuracy (forgetting) on Split CIFAR-100 w. pretrain on Tiny ImageNet.**

FAA (FF)	**Split CIFAR-100** (*pretr. Tiny ImageNet*)			
Method	**Class-IL**		**Task-IL**	
Joint (UB)	75.20 (–)		93.40 (–)	
Finetune	09.52 (92.31)		73.50 (20.53)	
oEwC [180]	10.95 (81.71)		65.56 (21.33)	
LwF [119]	10.83 (90.87)		86.19 (4.77)	
Buffer Size	500	2000	500	2000
ER [169]	31.30 (65.40)	46.80 (46.95)	85.98 (6.14)	87.59 (4.85)
CO^{2}L [28]	33.40 (45.21)	50.95 (31.20)	68.51 (21.51)	82.96 (8.53)
iCaRL [164]	56.00 (19.27)	58.10 (16.89)	**89.99** (**2.32**)	90.75 (1.68)
DER++ [23]	43.65 (48.72)	58.05 (29.65)	73.86 (20.08)	86.63 (6.86)
ER-ACE [26]	53.38 (21.63)	57.73 (17.12)	87.21 (3.33)	88.46 (2.46)
TwF (ours)	**56.83** (**23.89**)	**64.46** (**15.23**)	89.82 (3.06)	**91.11** (**2.24**)

small memory thanks to its *herding* buffer construction strategy. However, this effect is less pronounced on Split CIFAR-10 and Split CUB-200, where the role of pretraining is far more essential due to the similarity of the two distributions for the former and the higher difficulty of the latter. In these settings, we see iCaRL fall short of DER++, which better manages to maintain and reuse the features available from its initialization. Moreover, we remark that iCaRL and DER++ show ranging Class-IL performance in different tasks, whereas our method is much less sensitive to the specific task at hand.

While it proves effective on the easier Split CIFAR-10 benchmark, CO^{2}L does not reach satisfactory results on either Split CIFAR-100 or Split CUB-200. We ascribe this result to the high sensitivity of this model to the specifics of its training process (*e.g.*, to the applied transforms and the number of epochs required to effectively train the feature extractor with a contrastive loss). Remarkably, while we extended the size of the batch in all experiments with CO^{2}L to 256 to provide a large enough pool of negative samples, it still shows off only a minor improvement on non-rehearsal methods for Split CUB-200.

Interestingly, while both ER and ER-ACE do not feature distillation, we find their performance to be competitive for large enough buffers. In particular, the asymmetric objective of ER-ACE appears less sensitive to a small memory buffer but always falls short of DER++ when this constraint is less severe.

Table 2.8: **Accuracy (forgetting) on Split CUB-200 w. pretrain on ImageNet.**

FAA (FF)	**Split CUB-200** (*pretr. ImageNet*)			
Method	**Class-IL**		**Task-IL**	
Joint (UB)	78.54 (–)		86.48 (–)	
Finetune	8.56 (82.38)		36.84 (50.95)	
oEwC [180]	8.20 (71.46)		33.94 (40.36)	
LwF [119]	8.59 (82.14)		22.17 (67.08)	
Buffer Size	400	1000	400	1000
ER [169]	45.82 (40.76)	59.88 (25.65)	75.26 (9.82)	80.19 (4.52)
CO^2L [28]	8.96 (32.04)	16.53 (20.99)	22.91 (26.42)	35.79 (16.61)
iCaRL [164]	46.55 (12.48)	49.07 (11.24)	68.90 (3.14)	70.57 (3.03)
DER++ [23]	56.38 (26.59)	67.35 (13.47)	77.16 (7.74)	82.00 (3.25)
ER-ACE [26]	48.18 (25.79)	58.19 (16.56)	74.34 (9.78)	78.27 (6.09)
TwF (ours)	**57.78** (**18.32**)	**68.32** (**6.74**)	**79.35** (**5.77**)	**82.81** (**2.14**)

Transfer without Forgetting

Finally, results across all proposed benchmarks depict our method (TwF) as consistently outperforming all the competitors, with an average gain of 4.81% for the Class-IL setting and 2.77% for the Task-IL setting, w.r.t. the second-best performer across all datasets (DER++ and ER-ACE, respectively). This effect is especially pronounced for smaller buffers on Split CIFAR-10 and Split CUB-200, for which the pretrain provides a valuable source of knowledge to be transferred. We argue that this proves the efficacy of our proposal to retain and adapt features available from initialization through distillation. Moreover, we remark that its performance gain is consistent in all settings, further attesting to the resilience of the proposed approach.

2.2.5 Ablation Studies

Breakdown of the individual terms of TwF

To better understand the importance of the distinct loss terms in Eq. 2.15 and their connection, we explore their individual contribution to the final accuracy of TwF in Tab. 2.9. Based on these results, we make the following observations: *i)* $\mathcal{L}_{\text{CL}}$ is the most influential loss term and it is indispensable to achieve results in line with the SOTA; *ii)* $\mathcal{L}_{\text{FP}}$ applied on top of $\mathcal{L}_{\text{CL}}$ induces better handling of pretraining transfer, as testified by the increased accuracy; *iii)* $\mathcal{L}_{\text{AUX}}$ on top of $\mathcal{L}_{\text{FP}}$ reduces activation overlapping and brings a small but

Table 2.9: **Impact of each loss term and of using no memory buffer on TwF.** Results given in the Class-IL scenario following the same experimental settings as Tab.2.6-2.8.

$\mathcal{L}_{CL}$	$\mathcal{L}_{FP}$	$\mathcal{L}_{AUX}$	Split CIFAR-10			Split CIFAR-100			Split CUB-200		
Buffer Size			w/o/buf.	500	5120	w/o/buf.	500	2000	w/o/buf.	400	1000
✓	✓	✓	–	**83.65**	**89.55**	–	**56.83**	**64.46**	–	**59.67**	**68.32**
✓	✗	✗	–	75.79	87.54	–	44.01	57.84	–	56.53	67.29
✓	✓	✗	–	83.29	89.53	–	55.50	63.53	–	59.06	67.83
✗	✓	✗	60.07	62.63	62.75	49.14	50.20	50.22	37.57	38.43	38.93
✗	✓	✓	60.90	63.19	63.79	49.74	50.88	50.52	37.99	39.20	39.31

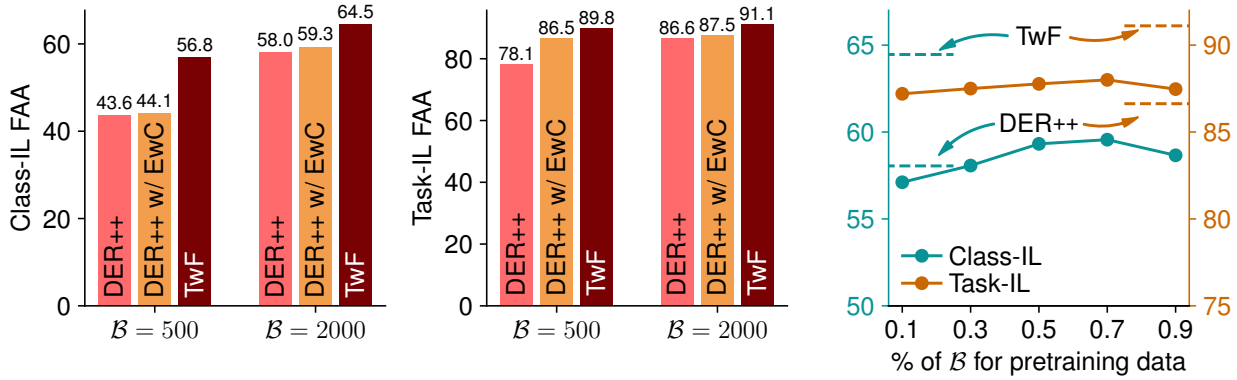

Figure 2.5: **Class-IL (left) and Task-IL (center) FAA performance comparison** of our proposal with different possible methods to retain knowledge from pretrain. **(Right) Influence of different allocation rates** of pretrain examples in $\mathcal{B}$ for DER++, $|\mathcal{B}| = 2000$.

consistent improvement.

Further, in the columns labeled as w/o/buf., we consider what happens if TwF is allowed **no replay example at all** and only optimizes $\mathcal{L}_{FP}$ and $\mathcal{L}_{AUX}$ on current task examples. Compared to oEwC in Tab. 2.6-2.8 – the best non-replay method in our experiments – we clearly see preserving pretraining features is in itself a much more effective approach, even with rehearsal is out of the picture.

Alternatives for the preservation of pretraining knowledge

TwF is designed to both preserve pretraining knowledge and facilitate its transfer. However, other approaches could be envisioned for the same purpose. Hence, we compare here TwF with two alternative baselines for pretraining preservation.

Pretraining preservation with EwC

We complement a strong approach such as DER++ with an additional regularization term based on EwC:

$$\mathcal{L}_{\text{EwC}} = \lambda(\theta - \theta^t)^T \operatorname{diag}(F)(\theta - \theta^t), \tag{2.17}$$

where $\operatorname{diag}(F)$ indicates the diagonal of the empirical Fisher Information Matrix, estimated on the pretraining data at the optimum θ^t. When equipped with this additional loss term, DER++ is anchored to its initialization and prevented from changing its pretraining weights significantly, while its replay-based loss term prevents forgetting of knowledge acquired in previous tasks. As shown by Fig. 2.5 (left, center), the EwC loss allows DER++ to improve its accuracy on Split CIFAR-100 with Tiny ImageNet pretraining (especially in the Task-IL setting). However, this improvement is not actively incentivizing feature reuse and thus falls short of TwF. We finally remark that TwF and DER++ w/ EwC have a comparable memory footprint (both retain the initialization checkpoint).

Pretraining preservation through rehearsal

An alternative for preserving the source knowledge is to assume that pretraining data is available and can be treated as an auxiliary data stream [11]. To evaluate this strategy with a bounded memory footprint, we test our baseline method (DER++) on Split CIFAR-100 with different percentages of the buffer dedicated to pretraining images (from Tiny ImageNet). The results shown in Fig. 2.5 (right) confirm our main claim: DER++ coupled with pretraining rehearsal improves over DER++ with only pretraining. This finding proves that, if pretraining is available, it is beneficial to guard it against catastrophic forgetting.

Furthermore, we highlight that TwF outperforms the baseline introduced here. When replaying pretraining data, indeed, the model has to maintain its predictive capabilities on the classes of the source task, *i.e.*, we enforce both backward and forward transfer. TwF, instead, allows the model to disregard the classes of the source dataset, as long as the transfer of its internal representations favors the learning of new tasks ($\Rightarrow$ **it only enforces forward transfer**). This substantial distinction helps to understand the merits of TwF: namely, a full but still functional exploitation of the pretraining knowledge.

Role of pretraining datasets

Here, we seek to gain further proof of our claim about the ability of TwF to adapt features from the pretrain. Specifically, we study a scenario where the source data distribution and the target one are highly dissimilar: namely, we first pretrain a ResNet18 backbone on

Table 2.10: **Dissimilar pretrain tasks: accuracy on CIFAR-100 pretrained on SVHN.**

FAA (FF)	**Class-IL**		**Task-IL**	
Buffer size	500	2000	500	2000
iCaRL [164]	39.59 (21.81)	42.02 (18.78)	78.89 (4.04)	80.65 (2.24)
DER++ [23]	36.46 (53.47)	52.29 (24.04)	75.05 (16.22)	83.36 (8.04)
TwF (ours)	**43.56** (**40.02**)	**56.15** (**21.51**)	**80.89** (**10.12**)	**87.30** (**3.12**)

SVHN [143] and then follow with Split CIFAR-100. We compare our model with the second-best performer from Tab. 2.7, *i.e.*, iCaRL, and DER++. The results, reported in Tab. 2.10, suggest that our method outranks the competitors not only when pretrained on a similar dataset – as in Tab. 2.7 – but also when the tasks are very dissimilar. We argue that this result further shows the ability of TwF to identify which pretraining features are really advantageous to transfer.

2.2.6 Discussion

This chapter introduced Transfer without Forgetting, a hybrid method combining Rehearsal and Feature transfer, designed to exploit pretrained weights in an incremental scenario. It encourages feature sharing throughout all tasks, yielding a stable performance gain across multiple settings. We also show that TwF outperforms other hybrid methods based on rehearsal and regularization and that it is able to profit even from pretraining on a largely dissimilar dataset. The proposed method was tested using two backbones with the same architecture, which in certain scenarios could be a limitation. As a next step, the Transfer without Forgetting components will be refactored to address this issue. This section, along with the preceding one, demonstrated that auxiliary knowledge can be utilized both in the form of raw data and as a pretrained network. This approach paves the way for incorporating more biologically inspired auxiliary information, which will be the focus of the next section

2.2.7 Publications

Boschini, M., Bonicelli, L., Porrello, A., Bellitto, G., Pennisi, M., Palazzo, S., ... & Calderara, S. (2022, October). Transfer without forgetting. In European Conference on Computer Vision (pp. 692-709). Cham: Springer Nature Switzerland. [19]

2.3 Saliency-driven Experience Replay for Continual Learning

2.3.1 Motivation

Humans possess the remarkable capability to keep learning, with limited forgetting of past experience, and to quickly re-adapt to new tasks and problems without disrupting consolidated knowledge. Machine learning, on the contrary, has shown significant limitations when dealing with non-stationary data streams with a limited possibility to replay past examples. The main reason for this shortcoming can be found in the inherent structure, organization and optimization approaches of artificial neural networks, which differ significantly from how humans learn and how their neural connectivity is built when accumulating knowledge over a lifetime. According to the *Complementary Learning Systems (CLS) theory* [107, 132], the human ability to learn effectively may be due to the interplay between two learning processes that originate, respectively, on the hippocampus and on the neocortex. This theory has inspired several continual learning methods [99, 101, 127]. In particular, the recent DualNet method [157] translates CLS concepts into a computational framework for continual learning. Specifically, it employs two learning networks: a *slow learner*, emulating the memory consolidation process happening in the hippocampus through contrastive learning techniques, and a *fast learner*, that aims at adapting current representations to new observations. However, this strategy still appears insufficient for addressing the problem of continual learning, because it starts from the (possibly wrong) assumption that human neural networks directly process visual input with the objective of performing categorization from early vision layers. On the contrary, neurophysiological studies [53, 103] are in near universal agreement that the object manifolds conveyed to primary visual cortex V1 (one of the earliest areas involved in vision) are as tangled as the pixel space. In other words, the neurons of the earliest vision areas do not contribute to object manifold untangling for categorization, but rather enforce luminance and contrast robustness [103]. This suggests that training early neurons with a visual categorization objective — as done not only in DualNet, but in all existing continual learning methods — is in stark contrast to the biological counterparts observed in primates.

Moreover, recent studies on the causes of forgetting in artificial neural networks showed that deeper layers (i.e., closer to the output) are less stable in presence of task shifts [162], which is consistent with the hypothesis that earlier layers do not bear specific categorization responsibilities.

Given these premises, it is peculiar that existing bio-inspired continual learning methods tend to ignore all upstream neural processes underlying visual categorization, such as visual

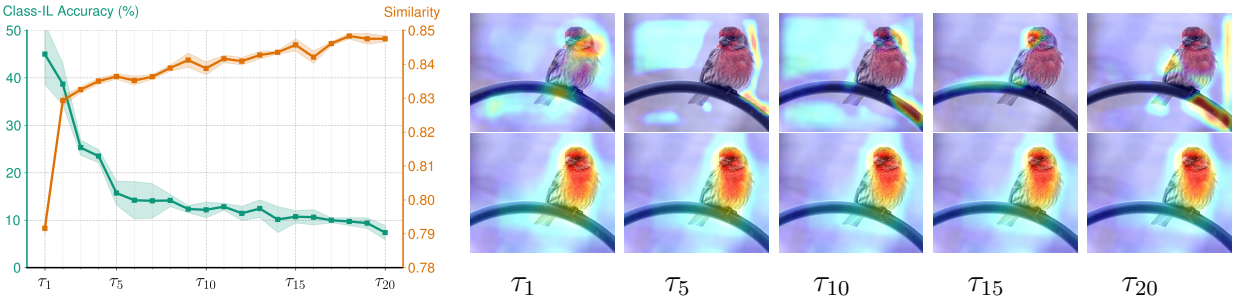

Figure 2.6: **Comparison of Forgetting-Free Saliency Prediction vs. Catastrophic Forgetting in Classifiers and Activation Maps in Continual Learning Scenarios**. *(Left figure)*: Saliency accuracy (measured as *similarity* [25]) improves with more tasks, while classification accuracy declines, indicating that saliency detection remains i.i.d. even with non-i.i.d. data. Images on the x axis show predicted saliency maps remain constant across tasks. *(Right figure)*: The top row shows activation maximization maps via GradCAM, which are prone to catastrophic forgetting due to their dependence on the classifier. In contrast, the bottom row shows saliency maps that remain consistent over time.

saliency processes. Indeed, the ability to select relevant visual information appears to be the hallmark of human/primate cognition. Moreover, recent findings in cognitive neuroscience have shown that the visual attention priorities of human hunter-gatherer ancestors are still embedded in the modern brain [144]: humans pay attention faster to animals than to vehicles, although we now see more vehicles than animals. This primordial saliency bias embedded in human brains suggests that the neuronal circuits of the ventral visual pathway are somehow inherited, as a form of genetic legacy from ancestral experience, and tend to remain stable over time — thus not subject to forgetting, though we have long stopped hunting to survive. Interestingly, we observed the same **forgetting-free** behavior for saliency prediction on artificial neural networks. Fig. 2.6 shows the trend of the *similarity* [25] metric for a saliency prediction model trained in a continual learning scenario, and compares it to the accuracy of a classification model under the same settings. While classification accuracy drops as the classifier learns new classes, the saliency metric remains stable, and even slightly improves.

From this observation, we propose *SER*, a *Saliency-driven Experience Replay* strategy that employs visual saliency prediction [17] to drive the learning of a sequence of classification tasks in a continual learning setting. To emulate what has been observed in primates, where visual saliency modulates the firing rate of neurons that represent the attended stimulus at different stages of visual processing [131, 198], SER adopts a two-branch model: one branch performs visual saliency prediction [55, 89, 121], and its responses modulate the features learned by a paired classification model in the second branch.

While the SER strategy stands out in its approach, it's important to note a similar category of methodologies that utilize attribution maps (e.g., computed via GradCAM), also known as

attention maps, as a distilled form of classifier knowledge for future replay [10,52,57,178,181]. However, **saliency prediction maps are significantly different from attribution maps**. Indeed, attribution maps elucidate the inner workings of DNNs by highlighting relevant input features for predictions and as such they suffer catastrophic forgetting (as shown in Fig. 2.6), while saliency maps, rooted in neuroscience and human visual processing, aim to emulate how humans perceive and prioritize visual information, and, most importantly, they are forgetting-free.

SER is model-agnostic and can be used in combination to any continual learning method. We demonstrate that saliency modulation positively impacts classification performance in online continual learning settings, leading to a significant gain in accuracy (up to 20 percent points) w.r.t. baseline methods. We further demonstrate the usefulness of saliency modulation on different benchmarks (including a challenging one that tackles fine-grained classification) and substantiate our claims through a set of ablation studies. We finally show that saliency modulation, besides being biologically plausible, leads to learn saliency-modulated features that are more robust to the presence of spurious features and to adversarial attacks.

2.3.2 Related Work

Continual Learning (CL) [45,133,151] addresses the problem of *catastrophic forgetting* in neural networks, wherein they tend to lose previously acquired knowledge when faced with shifts in input data distribution. Various solutions have been proposed to address this, including the incorporation of regularization terms [102, 229], specific architectural designs [130, 180], and rehearsal of previously encountered data points [23, 164, 169]. However, the application of these solutions to real-world scenarios is challenging due to evaluations often being based on unrealistic benchmarks [7, 201]. *Online Continual Learning* (OCL) [129] addresses this challenge by limiting multiple epochs on the input stream, reflecting the realistic assumption that data points encountered in real-world settings occur only once.

To address this challenge, many strategies adopt a replay approach [163,169]. Some focus on memory management: GSS [8] optimizes the basic rehearsal formula to store maximally informative samples, while HAL [31] identifies synthetic replay data points maximally affected by forgetting. CoPE [46] employs class prototypes for gradual evolution of the shared latent space, while ER-ACE [26] adjusts the cross-entropy loss asymmetrically to minimize task imbalance. Our proposal adopts a remarkably different approach w.r.t. these classes of methods, in that we take inspiration from cognitive neuroscience theory of learning and exploiting the features of a conjugate forgetting-free task (i.e., saliency prediction) to modulate the responses of our OCL model. Doing so produces a stabilizing effect on our model

and makes it more resilient to forgetting.

An approach similar in the spirit to ours is [123] that leverages saliency prediction for exemplar-free class incremental learning. To compensate for the absence of past task data, this methods relies on a pre-trained saliency detector, which remains frozen throughout the learning process, providing guidance for attribution maps of the classification backbone. Consequently, it tackles the challenge of forgetting by employing a pre-trained backbone to constrain feature drift. In contrast, SER operates on a dynamic framework where the visual saliency network is continuously trained, showcasing remarkable resistance to forgetting, while concurrently modulating the drift of classification features. This approach offers a more flexible visual saliency-classification paradigm that adapts to any dataset without external dependencies, as opposed to [123], which requires the use of a pre-trained saliency detector trained on the same data distribution as the target data.

Another approach, similarly inspired by cognitive theories, is DualNet [157], which employs two networks that loosely emulate how slow and fast learning work in humans. However, DualNet employs contrastive learning on the slow network (the earliest layers of the model), while it seems that object-identifying transformations happens later in the human visual system [53, 103]. Our results, reported later, substantiate the suitability of our choice to use low-level processes, such as saliency prediction, to drive continual learning tasks, rather than contrastive learning or classification pre-training techniques as, respectively, in DualNet and TwF [19].

Though the concept of utilizing saliency prediction maps in online continual learning is relatively new, recent trends have shown promising advancements in mitigating forgetting by encouraging models to recall evidence for past decisions, stored as activation maps [57]. Specifically, [10, 57, 178] employ attribution methods, such as Gradient-weighted Class Activation Mapping (Grad-CAM) [181], to compute and store visual model explanations for each sample (or parts thereof) in the buffer and ensures model consistency with previous decisions during the training phase.

Similarly, Dhar [52] adopts Grad-CAM, but it does not store any information, but it employs knowledge distillation on the activation maps across consecutive tasks.

However, as presented in the introduction, there is a fundamental distinction between saliency maps and activation maps with the latter being subject to forgetting, while the former not (Fig. 2.6).

Finally, our approach diverges from the recent trend in the continual learning (CL) field, which primarily employs foundation models (mostly Vision Transformers, ViTs) and focuses on learning prompts to mitigate forgetting [65, 192, 210, 211]. The main limitation of these methods is that they are restricted to transformer-based architectures. In contrast, our

strategy does not rely on any specific model type, thereby enhancing its potential impact on real-world applications.

2.3.3 Method

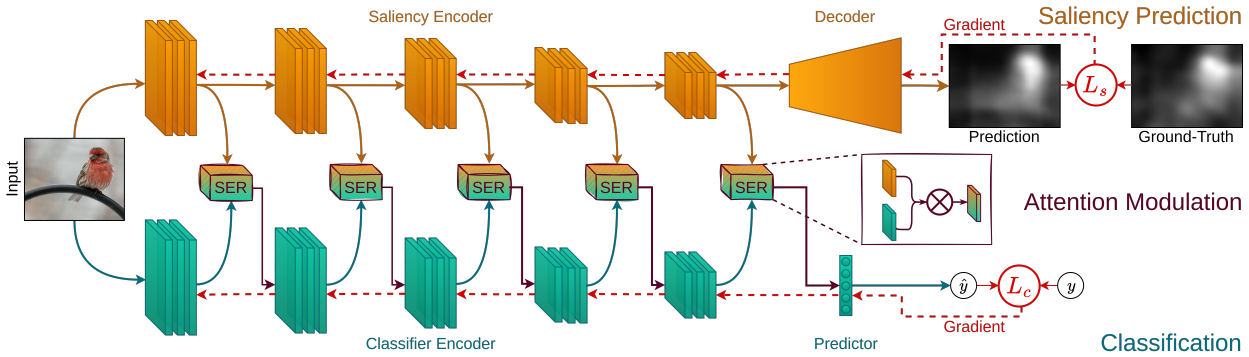

Figure 2.7: **Architecture of the proposed saliency-driven experience replay (SER) strategy.** The classification backbone is paired with a saliency prediction network that, given its capability of being forgetting-free, aims at adjusting the learned classification features in order to mitigate overall forgetting.

Online Continual Learning

Following the recent literature, we pose OCL as a supervised image classification problem with an online non-i.i.d. stream of data, where each training sample is only seen once. Although our saliency-driven modulation does not require the presence or knowledge of *task boundaries*, in this formulation and in our experiments we assume that these are given, to the benefit of any baseline method enhanced by the proposed extension.

More formally, let $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_T\}$ be a sequence of data streams, where each pair $(\mathbf{x}, y) \sim \mathcal{D}_i$ denotes a data point $\mathbf{x} \in \mathcal{X}$ with the corresponding class label $y \in \mathcal{Y}$; the sample distributions (in terms of both the data point distribution and the class label distribution) of different $\mathcal{D}_i$ and $\mathcal{D}_j$ may vary — for instance, class labels from $\mathcal{D}_i$ might be different from those from $\mathcal{D}_j$, though both must belong to the same domain $\mathcal{Y}$.

Given a classifier $f : \mathcal{X} \to \mathcal{Y}$, parameterized by $\boldsymbol{\theta}$, the objective of OCL is to train f on $\mathcal{D}$, organized as a sequence of T tasks $\{\tau_1, \dots, \tau_T\}$, under the constraint that, at a generic task τ_i, the model receives inputs sampled from the corresponding data distribution, i.e., $(\mathbf{x}, y) \sim D_i$, and sees each sample only once during the whole training procedure. The classification model may optionally keep a limited *memory buffer* $\mathbf{M}$ of past samples, to reduce forgetting of features from previous tasks. The model update step between tasks can

be summarized as:

$$\langle f, \boldsymbol{\theta}_{i-1}, \mathcal{D}_{i-1}, \mathbf{M}_{i-1} \rangle \rightarrow \langle f, \boldsymbol{\theta}_i, \mathbf{M}_i \rangle \tag{2.18}$$

where $\boldsymbol{\theta}_i$ and $\mathbf{M}_i$ represent the set of model parameters and the buffer at the end of task τ_i, respectively.

For methods that do not exploit buffer, $\mathbf{M}_i = \varnothing, \forall i$.

The training objective is to optimize a classification loss over the sequence of tasks (without losing accuracy on past tasks) by the model instance at the end of training:

$$\arg\min_{\boldsymbol{\theta}_T} \sum_{i=1}^{T} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_i} \Big[\mathcal{L}\Big(f\left(\mathbf{x}; \boldsymbol{\theta}_T\right), y \Big) \Big] \tag{2.19}$$

where $\mathcal{L}$ is a generic classification loss (e.g., cross-entropy), which a continual learning model attempts to optimize while accounting for model *plasticity* (the capability to learn current task data) and *stability* (the capability to retain knowledge of previous tasks) [133].

SER: Saliency-driven Experience Replay

Our method is grounded on the neurophysiological evidence that attention-driven neuronal firing rate modulation is multiplicative and the scaling of neuronal responses depends on the similarity between a neuron's preferred stimulus and the attended feature [131, 198]. This hypothesis is translated into a general artificial neural architecture, where we emulate the process of human selective attention through a visual saliency prediction network [17] whose activations modulate, through multiplication, neuron activations of a paired classification network at different stages of visual processing. Formally, let $S : \mathcal{X} \rightarrow \mathcal{S}$ be a saliency prediction network, where $\mathcal{X}$ is the space of input images and $\mathcal{S}$ the space of output saliency maps. Generally, if $\mathcal{X} = \mathbb{R}^{3\times H\times W}$ for RGB images, then $\mathcal{S} = \mathbb{R}^{H\times W}$, where each location of a map $\mathbf{s} \in \mathcal{S}$ measures the *saliency* of the corresponding pixel in the RGB space. We assume that S can be decomposed into two functions, an encoder $E : \mathcal{X} \rightarrow \mathcal{H}$ and a decoder $D : \mathcal{H} \rightarrow \mathcal{S}$, such that $S\left(\mathbf{x}\right) = D\left(E\left(\mathbf{x}\right)\right)$, for $\mathbf{x} \in \mathcal{X}$. Then, given an online continual learning problem with data stream $\mathcal{D}$ and set of classes $\mathcal{Y}$, let $C : \mathcal{X} \rightarrow \mathcal{Y}$ be a classification network, such that C and the saliency encoder E share the same architecture (with independent parameters). An illustration of the proposed architecture is shown in Fig. 2.7.

At training time, both S and C observe the same data stream, from which pairs $(\mathbf{x}, y)$ of input data and class label are iteratively sampled. Through the use of an external *saliency oracle*, we extend each data sample to a triple $(\mathbf{x}, y, \mathbf{s})$, where $\mathbf{s}$ is the target saliency map associated to $\mathbf{x}$. The oracle can be either a set of ground-truth maps, when available, or *pseudo-labels* provided as the output of a pre-trained saliency predictor (unrelated to S). We

therefore proceed to optimize a multi-objective loss function $\mathcal{L} = \mathcal{L}_s + \lambda\mathcal{L}_c$, with λ being a weighing hyperparameter. Loss term $\mathcal{L}_s$ is computed on the output of saliency predictor S, and compares the estimated saliency map $S(\mathbf{x})$ with the target $\mathbf{s}$ by means of the Kullback-Leibler divergence (commonly employed as a saliency prediction objective [12,25,55,79,212]):

$$\mathcal{L}_s = \sum_i s_i \log\left(\frac{s_i}{S_i(\mathbf{x}) + \epsilon} + \epsilon\right) \tag{2.20}$$

with s_i and $S_i(\mathbf{x})$ iterating over map pixels in $\mathbf{s}$ and $S(\mathbf{x})$, respectively. Loss term $\mathcal{L}_c$ encodes a generic online continual learning objective, as introduced in Eq. 2.19. As the proposed approach is method-agnostic, details on the formulation of $\mathcal{L}_c$ may vary.

In order to enforce selective attention-driven modulation of classification neuronal activations, we leverage the architectural identity of saliency prediction encoder E and classifier C to alter the feedforward pass of the latter, by multiplying pre-activation features in C by the corresponding features in E, before applying a non-linearity and feeding them to the next layer of the network. Formally, let us assume that the C and E networks consist of a sequence of layers $\{l_1, l_2, \ldots, l_L\}$. Without loss of generality, let each layer l_i compute its output as $\mathbf{z}_i = \sigma(\mathbf{W}_i\mathbf{z}_{i-1})$, with σ being an activation function, $\mathbf{W}_i$ the network-specific layer parameters (i.e., not shared between E and C) and $\mathbf{z}_{i-1}$ the output of the previous layer (or the network's input $\mathbf{x}$, if appropriate). Then, let us distinguish between features $\mathbf{z}_i^{(s)}$ and $\mathbf{z}_i^{(c)}$, respectively representing the output of layer l_i by the saliency prediction encoder S and the classifier C. We apply saliency-driven modulation by modifying the computation of $\mathbf{z}_i^{(c)}$ as follows:

$$\mathbf{z}_i^{(c)} = \sigma\left(\mathbf{W}_i^{(c)}\left(\mathbf{z}_{i-1}^{(c)} \odot \mathbf{z}_{i-1}^{(s)}\right)\right) \tag{2.21}$$

where $\odot$ denotes the Hadamard product. Intuitively, the proposed approach encourages the classification model to attend to "salient" features of the input, where the concept of *saliency* is generalized from the pixel space to hidden representations. It is important to note that, at training time, gradient descent optimization of $\mathcal{L}_c$ would also affect on the saliency encoder E. This is undesirable, as we previously showed (see Fig. 2.6) that saliency features are robust to task shifts, unlike classification features: hence, in order to guarantee this property, we stop the gradient flow from $\mathcal{L}_c$ to parameters in E, and use it to update the parameters of classifier C only.

In the above formulation, we assumed the presence of a classification network with fully-connected layers; however, our method can be applied in an agnostic manner to any method employing, at least in part, a feature extractor implemented as a neural network. As such, the proposed method can be equally applied, for instance, both to end-to-end classification

models (e.g., DER++ [23]) and to approaches with a neural backbone that computes class-representative prototypes (e.g., CoPE [46]).

2.3.4 Experimental Results

Experimental setup

Benchmarks. We build two OCL benchmarks by taking image classification datasets and splitting their classes equally into a series of disjoint tasks:

- **Split Mini-ImageNet** [36, 50, 56, 204] that includes 100 classes from ImageNet, allowing for a longer task sequence. For each class, 500 images are used for training and 100 for evaluation.
- **Split FG-ImageNet**[4] [176] is a benchmark for fine-grained image classification that we use to test CL methods on a more challenging task than traditional ones. It includes 100 classes of animals extracted from ImageNet, belonging to 7 different species, reducing inter-class variability and leading to harder tasks. Each class contains 500 samples for training and 50 for evaluation.

For both datasets, images are resized to 288×384 pixels and split into twenty 5-way tasks.

Baseline methods. We evaluate the contribution of the SER strategy when paired to a classification network trained using several state-of-the-art continual learning approaches, including rehearsal and non-rehearsal methods:

- **DER++** [18]: a seminal work that combines rehearsal and knowledge distillation strategies for supporting model plasticity while limiting forgetting.
- **ER-ACE** [26]: a variant of experience replay [163, 169] which aims to prevent imbalances due to the simultaneous optimization of the current and past tasks by selectively masking softmax outputs.
- **CoPE** [46]: a prototype-based classifier with experience replay, whose careful update scheme prevents sudden disruptions in the latent space during incremental learning.
- **LwF** [119]: a non-rehearsal method that enforces a model to preserve outputs of past model instances on new samples to limit forgetting.
- **oEWC** [102]: a non-rehearsal method that mitigates forgetting by selectively limiting the changes on weights that are most informative of past tasks.

[4] https://www.kaggle.com/datasets/ambityga/imagenet100

Implementation details. We apply the SER strategy at five feature modulation points of ResNet-18's architecture, namely, the outputs of the first convolutional block and of the four main residual blocks. In compliance with online learning, all models are trained for a single epoch, using SGD as optimizer, with a fixed batch size of 8 both for the input stream and the replay buffer. Rehearsal methods are evaluated with three different sizes of the memory buffer (1000, 2000 and 5000). When applying SER, besides each method's specific training objective, we also optimize the saliency prediction loss $\mathcal{L}_s$ from Eq. 2.20, with $\lambda = 1$. Saliency is estimated using DeepGaze IIE network [121] as oracle.

When using SER, classifier C and saliency predictor S are identical ResNet-18 architectures, followed — respectively — by a linear classification layer and a saliency map decoder.

While C is trained from scratch, we employ a pre-trained saliency predictor S, consistently with neuroscience evidence showing that humans have selective attention already embedded in the brain [144]. For a fair comparison, in all our experiments feature extraction backbones of baseline methods are initialized to the same pre-trained weights as S (except where explicitly stated).

Care was taken to ensure that the set of OCL classes $\mathcal{C}$ did not semantically overlap with pre-training data, to prevent any contamination from the saliency predictor to the classification task. Specifically, S was pre-trained for 20 epochs on a subset of 100 ImageNet classes (disjoint from our two main benchmark datasets), using DeepGaze IIE as oracle. No class label information was used at this stage. All experiments were conducted on a workstation with an 24-core CPU, 500GB RAM, and an NVIDIA A100 GPU (40GB VRAM). Results are computed using the Mammoth framework [23].

Metrics and evaluation. As a primary metric of OCL model performance, we report the *final average accuracy* as $\frac{1}{T}\sum_{i=1}^{T} a_i^T$, where a_i^T is the accuracy of the final model on the test set of task τ_i. Accuracy a_i^T can be computed in a *Class-Incremental Learning* (*Class-IL*) or in a *Task-Incremental Learning* (*Task-IL*) setting. In the latter, we assume that a task identifier is provided to the model at inference time, simplifying the problem by restricting the set of class predictions for a given sample.

While Task-IL is often depicted as a trivial scenario in recent literature [8, 60, 200], we emphasize its usefulness, as it isolates the effect of within-task forgetting from the model's bias towards the currently learned classes [18, 78, 214].

Results

We first evaluate the contribution that saliency-driven modulation provides to state-of-the-art OCL baselines. For each method, we compute Class-Incremental accuracy and compare to those obtained when integrating SER, as described in Sec. 2.3.3.

Since our strategy foresees two paired networks for classification and saliency prediction, we also compare with similar multi-branch CL baselines:

- **DualNet** [157], mentioned in Sec. 2.3.1, employs a dual-backbone architecture to decouple incremental classification (by a *fast learner*) from self-supervised representation learning [227] (by a *slow learner*). We adapt SER to DualNet by replacing the slow learner and its training objective with our saliency prediction backbone, forcing the fast learner to use saliency features for classification.

- **TwF** [19] employs a frozen pre-trained classification backbone to stabilize the learning of Class-Incremental features, by means of an attention mechanism. To enable SER, the pre-trained classification backbone and the feature distillation strategy are replaced with the saliency encoder, and the features of the two backbones are combined through multiplication, as described in Sec. 2.3.3.

Results are reported in Table 2.11, showing a pattern of enhanced performance when integrating SER up to 20 percent points. In terms of comparison against two-paired networks, integrating SER outperforms both of them, suggesting that controlling learning through saliency leads to better representation for classification than, for instance, contrastive learning (as done in DualNet) or feature attention with a pre-trained backbone (as in TwF)[5]. This is inline with cognitive neuroscience [53,114], for which object identity-preservation, that also involves contrastive learning, happens mostly at later layers (e.g., IT neurons), while selective attention (through visual saliency) acts during the whole categorization process.

2.3.5 Ablation Studies

The proposed strategy is grounded on cognitive neuroscience literature, according to which selective attention modulates neuronal responses of all layers involved in the categorization process, in a multiplicative fashion. Our next experiments are meant to assess whether this hypothesis (i.e., feature modulation through multiplication for all classification layers) is optimal also for artificial neural networks, or if other integration modalities of saliency information may be equally effective. We thus compare our SER strategy with the following baselines, all exploiting saliency information in different ways:

- **Saliency-based input modulation (SIM)**: the input image is multiplied by the corresponding estimated saliency map (thus highlighting salient regions only).

[5]We could not run TwF with buffer size of 5000, due to excessive computing requirements.

Table 2.11: **Class-Incremental accuracy of SOTA rehearsal-based methods** with and without SER.

Model	**Split Mini-ImageNet**			**Split FG-ImageNet**		
Joint	14.79±1.17			9.06±1.07		
Fine-tune	3.43±0.35			2.43±0.81		
Buffer size	**1000**	**2000**	**5000**	**1000**	**2000**	**5000**
DER++	14.95±3.11	12.82±4.97	14.58±2.55	8.08±1.54	8.27±1.72	9.20±0.86
↪**SER**	**19.13**±1.62	**22.92**±2.25	**25.35**±2.56	**11.71**±2.36	**12.97**±1.62	**13.73**±1.95
ER-ACE	20.86±3.69	24.93±3.20	26.31±5.22	14.28±0.96	16.45±1.24	18.21±3.45
↪**SER**	**27.48**±2.83	**33.09**±1.28	**35.58**±1.79	**20.03**±3.13	**23.80**±2.11	**28.68**±0.50
CoPE	21.58±1.60	23.58±4.39	24.77±3.56	16.45±1.38	16.81±0.83	17.77±2.02
↪**SER**	**26.66**±2.22	**33.35**±4.67	**45.04**±2.44	**18.17**±2.79	**27.14**±1.62	**34.34**±3.51
	Dual-branch methods					
TwF	23.78±1.67	29.05±2.02	–	15.32±2.59	18.72±1.75	–
↪**SER**	**28.36**±3.72	**35.55**±0.61	–	**20.04**±1.63	**22.54**±2.20	–
DualNet	20.57±0.91	27.41±1.79	32.08±1.55	15.62±1.54	21.04±1.08	22.07±2.08
↪**SER**	**28.58**±1.40	**33.76**±1.21	**36.44**±0.77	**19.48**±0.59	**22.53**±1.56	**24.83**±2.01

- **Saliency as additional input (SAI)**: we modify the classification network to receive as input a 4D data tensor, with the saliency map concatenated to RGB channels.
- **Learning saliency-based modulation (LSM)**: rather than multiplying classification features $\mathbf{z}_{i-1}^{(c)}$ and saliency features $\mathbf{z}_{i-1}^{(s)}$ (see Eq. 2.21), we feed them to convolutional layer with 1×1 kernel to produce $\mathbf{z}_{i}^{(c)}$, and let the model learn the corresponding parameters.

Fig. 2.8 reports the results of this analysis, using DER++ and ER-ACE as baseline methods, and clearly indicates the superiority the SER strategy to other saliency integration variants. However, it is interesting to note that saliency helps classification performance in all cases, demonstrating its usefulness for continual learning tasks. We argue that this is due to the intrinsic nature of saliency prediction, which we found to be i.i.d. with respect to the data stream.

We then investigate whether the impact of selective-driven modulation is uniform across the backbone layers. To this aim, we define a positional binary coding scheme, controlling the application of the SER strategy at the predefined points of the network (see Sect. 2.3.4): if position i of the coding scheme is 1, then the i-th feature modulation point is enabled, i.e., features from the i-th block of the classification network are multiplied by the features of the

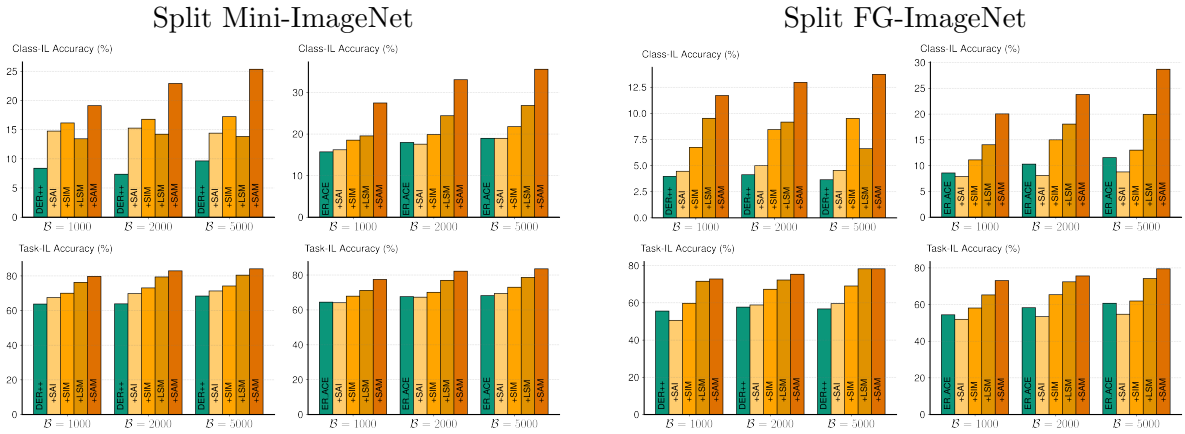

Figure 2.8: **Comparison of SER to alternative saliency integration strategies**. **SIM** modulates input images by saliency maps. **SAI** provides saliency maps as an additional input channel to the classification network. **LSM** merges classification and saliency features through a learnable convolutional layer.

i-th block of the saliency network. Results are reported in Table 2.12 for both DER++ and ER-ACE, and indicate that the best strategy is to modulate the features of all classification layers through the corresponding saliency ones, similarly to what neurophysiological evidence reports [131, 198].

Table 2.12: **Performance comparison when applying SER to DER++ and ER-ACE** at different layers of the ResNet-18 backbone, with buffer size 2000 (Class-IL).

	Split Mini-ImageNet		**Split FG-ImageNet**	
SER Scheme	**DER++**	**ER-ACE**	**DER++**	**ER-ACE**
1 1 1 0 0	12.97±2.62	23.72±0.77	6.54±0.67	18.08±0.96
1 1 1 1 0	17.46±1.02	26.44±2.33	8.77±1.45	16.55±2.55
1 1 1 1 1	**22.92**±2.25	**33.09**±1.28	**12.97**±1.62	**23.80**±2.11

Model Robustness

We finally assess the robustness of the SER strategy to *spurious features* and *adversarial attacks*. Spurious features are information that correlates well with labels in training data. but not in test data (e.g., in a classification task between birds and dogs, training with yellow birds and black dogs only), leading to low generalization [112]. This effect is exacerbated in continual learning settings, where the covariate shift between train data and test data increases as new tasks come in. Thus, we measure to what extent our SER strategy can mitigate the tendency of learning methods to exploit spurious features to solve classification

tasks. We crafted an ad-hoc benchmark consisting of ten classes from ImageNet. For each class, we added a class signature for training images, leaving the test images unaltered. In detail, we modified each training image by increasing the brightness of all pixels by a class-dependent offset, computed as $5(c+1)$ (in a 0-255 brightness range), where c is a numeric class label. We then define five continual learning tasks with two classes each. We then compare ER-ACE to the corresponding SER-enabled variant and ground its performance with the one obtained when it is trained with original images (i.e., without enforcing spurious features in the data). Results in Table 2.13 show that SER effectively limits the possibility for the classifier to use spurious features, resulting in a more robust and generalizing model. The drop of performance (about 22 percent points) observed between training with the original data and training with data biased by spurious features is almost completely recovered when SER is used.

Finally, we evaluate the robustness of SER against adversarial perturbations of the input space. To this aim, we apply the Projected Gradient Descent (PGD) attack [128] with different ε values (determining the strength of the attack) and compare the average performance drop experienced by ER-ACE, in its original version and when combined with SER. We conduct the evaluation on both Split Mini-ImageNet and Split FG-ImageNet, repeating each experiment three times. As shown in Figure 2.9, SER considerably improves model stability, counteracting perturbations by regularizing classification features with saliency ones.

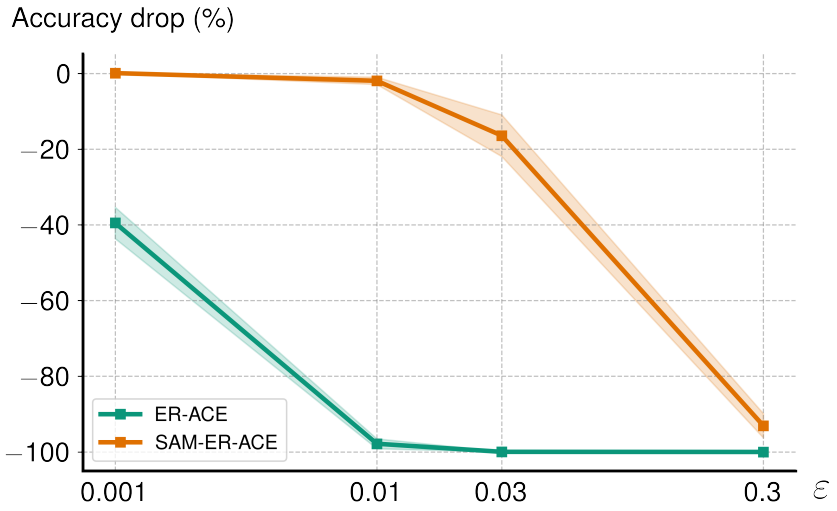

Figure 2.9: **Robustness to adversarial attacks**. ER-ACE baseline drops even with small attacks, while SER significantly enhances robustness.

Method	Class-IL	Task-IL
ER-ACE	50.07±3.88	86.77±1.63
ER-ACE$^{\mathcal{SF}}$	28.46±3.46	74.40±4.37
↪**SER**	**44.08**±3.67	**83.04**±3.06

Table 2.13: **Effect of the SER strategy in the presence of spurious features**. The $\mathcal{SF}$ apex shows training on a biased dataset with spurious features.

2.3.6 Discussion

In this section we presented SER, a biologically-inspired saliency-driven modulation strategy for online continual learning, which regularizes classification features using visual saliency, effectively reducing forgetting. The proposed approach, grounded on neurophysiological evidence, significantly improves performance of state-of-the-art OCL methods, and has been shown to be superior to other multi-branch solutions, either biologically-inspired (e.g., DualNet) or based on attention mechanisms (e.g., TwF). Our results confirm that adapting neurophysiological processes into current machine learning techniques is a promising direction to bridge the gap between humans and machines.

We introduced the use of saliency maps as auxiliary knowledge to mitigate forgetting in continual learning. This involves pre-training our saliency predictor with an oracle, which could be in the form of either ground-truth maps or an external model generating pseudo-labels. High-quality input images are necessary for producing meaningful saliency maps, thus, datasets like CIFAR10/100 cannot be employed due to their lower resolution.
Although SER is model-agnostic, its formulation necessitates that the saliency encoder and the classifier share identical architectures. To apply this to heterogeneous networks, we will explore defining or learning mappings between activations at different network stages.

Finally, our finding that saliency prediction is *i.i.d.* with respect to classification distribution shifts opens the door to investigating whether other low-level visual tasks share this property.

2.3.7 Publications

Bellitto, G., Salanitri, F. P., Pennisi Matteo, Boschini, M., Porrello, A., Calderara, S., ... Spampinato, C. (2024). Selective Attention-based Modulation for Continual Learning. arXiv Preprint arXiv:2403. 20086. Accepted at NIPS 2024. [13]

Chapter 3

Federated Learning with Privacy Preserving Generative AI

This chapter tackles the challenge of Federated Learning, but from a novel perspective. While most existing methods focus on model training and aggregation, we assume that Federated Learning can be improved by focusing on the data itself, specifically, by generating synthetic data that is safe to share. To explore this, in this chapter, we present two works leveraging generative models to create synthetic versions of the data that can be securely transferred across federation nodes. Enabling privacy-safe data sharing paves the way to apply Continual Learning methods, such as rehearsal strategies discussed in Chap. 2, to Federated Learning, which will be explored in the next chapter.

In the first study (Section 3.1), we use latent-space clustering to generate aggregated synthetic samples, which are then shared with a central server within a simulated healthcare federation. In the second study (Section 3.2), we further enhance the privacy-preservation strategy by also enhancing sample diversity. This is achieved by identifying a privacy-safe sampling path within the latent space of an image generator, ensuring the synthetic data offers both privacy and representational power to solve specific tasks.

3.1 GAN latent space manipulation and aggregation for federated learning in medical imaging

3.1.1 Motivation

The recent success of deep learning in the medical domain has shown it to be a promising tool to support medical diagnosis and treatment, but large amounts of training data are still

needed to build models able to achieve good accuracy and generalization. However, medical institutions generally curate their own datasets and keep them private for privacy concerns. Due to their small size, models trained on private datasets tend to overfit, introduce biases and generalize badly on other data sources that address the same task [228].

A viable solution for increasing the size and diversity of data is to employ a collaborative learning strategy, where multiple distributed nodes support the training of a model for a shared task [217]. Federated Learning [134, 184], in particular, has emerged as a training paradigm where each node trains a copy of a shared model on its private data and sends the local updates to a central server, where model parameters are tuned based on aggregated local updates. However, aggregating gradients or weights from multiple nodes does not deal with the non-i.i.d. nature of distributed data. Furthermore, gradient integration raises privacy issues as training data might be reconstructed, to a certain degree, starting from the shared gradients as demonstrated in [67, 232, 235].

In this work, we propose a generative approach where each distributed node generates, and shares, a synthetic version of its own data through manipulation and aggregation of latent spaces learned by a Generative Adversarial Network (GAN). In particular, our synthetic samples are drawn from the same distribution as the original ones, but are designed to prevent the inclusion of patient-specific visual patterns. Sharing the manipulated images, rather than the generation model, prevents the reconstructions of real data through attacks to the model and circumvents the gradient/weight aggregation problem.

We tested our approach on the task of tuberculosis classification from X-ray images of two different datasets, namely, the Montgomery County X-ray Set and Shenzhen Hospital X-ray Set [27,83,84]. Our experiments simulate a multi-node multimodal data scenario, where each dataset is located on a different node. It achieves 75% and 60% in classification accuracy on the Shenzhen and the Montgomery datasets, respectively, whereas standard centralized training on the dataset union (i.e., not in a federated learning setting) yields 78% and 43%. The capabilities of our approach to synthesize images visually distant from the real ones are measured quantitatively by evaluating LPIPS distance [231] between real images and samples generated through latent space optimization on a standard (non-privacy-preserving) GAN and by the proposed approach. Qualitatively, we also show several examples of generated images with corresponding closest match in the real dataset, demonstrating significant differences that prevent tracing back to the original real distribution.

3.1.2 Related Work

Federated learning (FL) embraces a family of privacy-preserving distributed learning strategies that allow nodes to keep training data private, while supporting the creation of a shared model. Typically, a central server sends a model to a set of client nodes; local model updates are aggregated by the server, which sends the new model to the clients in an iterative process. In FedAvg [134], the server computes model averaging combining local stochastic gradient descent updates of each client. FedProx [116] is a generalization and re-parametrization of FedAvg proving theoretically convergence guarantee when training over non-identical distributed data (statistical heterogeneity). FedMA [206] builds a shared global in a layer-wise manner by matching and averaging hidden elements with similar feature extraction signatures. All these methods attempt to train a central model using the gradients gathered from multiple models trained on local private data.

FL particularly suits medical field applications, where data privacy is a critical concern. Li et al. [117] present the first FL system for medical image analysis, employing FedAvg and differential privacy [1] for brain tumor segmentation. Roy et al. [175] also apply FL for whole-brain segmentation in MRI. Recently, several other collaborative learning methods [44, 61, 150] have been proposed, especially because of the emergency need raised by the COVID-19 pandemic, in order to harness multiple data sources to promptly react to emergency scenarios.

However, gradient aggregation does not seem to guarantee the required level of data privacy, as it has been demonstrated that network inputs can be recovered from gradient updates [67, 209, 235]. Differential privacy [1, 70, 111] attempts to reduce this issue by obfuscating gradients through noise. Zhu et al. [235], for instance, add Gaussian/Laplacian noise to gradients and compress the model with gradient pruning. However, adding noise to the gradients significantly compromises model's performance.

In this work, we tackle the problem of federated learning from a data-perspective: rather than sharing weights/updates, which can be attacked, we share a synthetic version of private data — generated through a GAN — that retains visual content to support distributed training, but improves privacy by hiding specific visual patterns of patients. GANs have been also employed in federated learning regime, but always in the view of aggregating parameters to create a general model. In GS-WGAN [37], a gradient-sanitized Wasserstein GAN improves differential privacy, by carefully distorting gradient information in a way that reduces loss of information and generates more informative samples. Federated CycleGAN [193] is designed to perform unsupervised image translation; however, they still share local gradients, which may introduce the above privacy concerns. FedDPGAN [230] designs a distributed

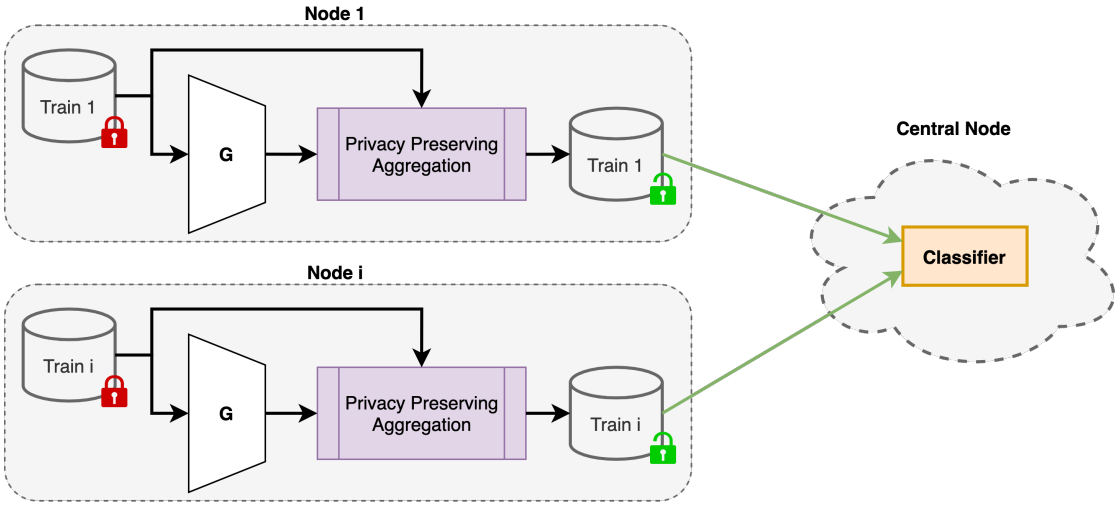

Figure 3.1: **The proposed federated learning framework.**

DPGAN [215] trained in a FL framework, to train models for COVID-19 diagnosis from chest X-ray images, without data sharing. In [161], the authors propose a framework to extend a large family of GANs to a FL setting utilizing a centralized adversary.

3.1.3 Method

Overview

In our approach, shown in Fig. 3.1, a set of distributed nodes create synthetic images and share them with a central node, where a model is trained using the received data. Specifically, each node trains a GAN to transform its own private dataset into a privacy-preserved one where patient information leak is minimized. The visual features of the privacy-preserved dataset still come from the same distribution of the real private one (as per GAN training) in order to support the training of the centralized model.

Although we do not perform a formal security analysis of our approach, for the sake of readability we will refer to it as "privacy-preserving", to distinguish it from the cases where no precaution is taken to prevent patient information leak in the sharing and learning process (referred to as "non privacy-preserving").

Generative Adversarial Network

Generative Adversarial Networks (GANs) [68] consist of two networks, a generator model and a discriminator model: the former is trained to generate realistic images, while the latter is trained to distinguish between real and synthetic samples. In the conditional settings, where

the generation process is controlled by a label to synthesize samples for a specific class, the two models are alternately trained to minimize the following losses, respectively:

$$L_D = \mathbb{E}_{x,y}[\log(D(x,y))] + \mathbb{E}_{z,y}[\log(1 - D(G(z,y),y))] \tag{3.1}$$

$$L_G = \mathbb{E}_{z,y}[\log(D(G(z,y),y))] \tag{3.2}$$

where (x, y) is sampled from the real data distribution $\mathcal{D}$, z is sampled from a latent distribution $\mathcal{Z}$ (mapped by generator G to the real distribution for class y) and D is the discriminator model that predicts the likelihood of the input being real, given the target label. During training, the better D becomes at recognizing fake samples, the more G has to improve its generation capabilities, thus increasing the realism of synthetic data.

In this work, our GAN architecture is based on StyleGAN2 [95], where an auxiliary network maps a class-conditioned latent vector z to an intermediate latent vector $w \in \mathcal{W}$, which helps to improve generation quality and simplifies the projection of real images in $\mathcal{D}$ to the latent space $\mathcal{W}$. Indeed, given a real image x of class y, it is then possible to find an intermediate latent point $\hat{w}$ such that $G(\hat{w}) \approx x$, by optimizing the LPIPS distance loss [231] between x and $G(\hat{w})$ with respect to $\hat{w}$, which measures the similarity of activations by a pre-trained model. Of course, this projection property negatively affects the sought privacy in FL, as the generated synthetic distribution may contain visual patterns highly similar to those of the original samples.

Privacy-Preserving Aggregation

To address the privacy limitation of existing GAN methods, we propose a *Privacy-Preserving Aggregation* strategy (shown in Fig. 3.2) injected in the GAN training during data generation to encourage privacy.

Let $\hat{W} = \{\hat{w_1}, \hat{w_2}, ..., \hat{w_N}\}$ be a set of points obtained by projecting N images onto the GAN latent space, for a given dataset class. We carry out spectral clustering [145] based on LPIPS distances between the images corresponding to $\hat{W}$ projections. Cluster centroids $\hat{W^c} = \{\hat{w_1^c}, \hat{w_2^c}, ..., \hat{w_M^c}\}$, representing latent aggregations with similar visual features in terms of LPIPS distance, are then employed as a starting point for data synthesis. Working with centroids allows us to capture shared patterns between dataset samples while improving privacy, since the resulting latent vectors cannot be traced back to specific patients. To create enough synthetic samples to allow model's training, we then carry out an augmentation procedure based on linearly interpolating the $\hat{W^c}$ centroids in the latent space and generating training samples using points along the trajectories between them. This is also beneficial for increasing dataset variability, as it allows to produce samples that combine patterns of

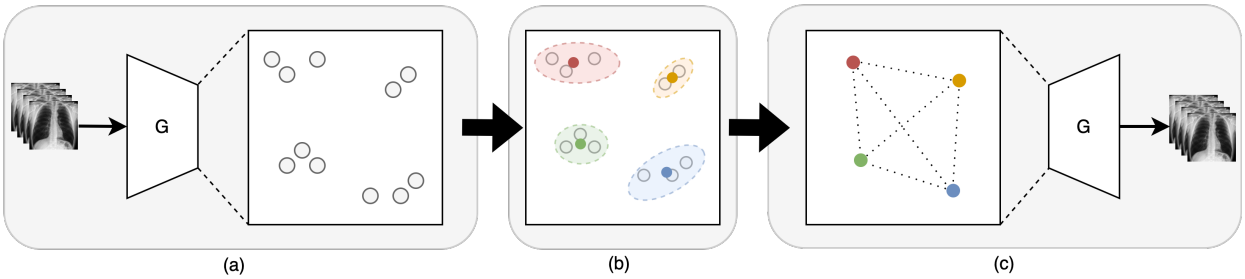

Figure 3.2: **Privacy Preserving Aggregation:** a) a generator G is trained for each node using its own private dataset. Training images are then projected in the generator latent space; b) projected latent vectors are clustered through spectral clustering, based on pairwise LPIPS distance between corresponding images; c) linear interpolation among cluster centroids produces new latent vectors, which are used to generate synthetic samples that are sent to the central node.

groups of patients (e.g., interpolating clusters with lesions on left/right lung may produce synthetic images with lesions on both lungs), leading to better generalization capabilities. Note that clustering and interpolation are carried out independently for each dataset class, by exploiting the conditional generation capabilities of the generator. This ensures that sampled latent vectors are assigned a well-defined label, making the corresponding synthetic images suitable for training the central node classifier. Clusters with only one sample are discarded in the process.

3.1.4 Experimental Results

We test the proposed approach on the task of tuberculosis classification from X-ray images in a non-i.i.d. federated learning setting, where different datasets are used for each node, to simulate a more realistic training scenario. Each node generates synthetic X-ray images by applying our aggregation approach on its private dataset; images generated by each node are shared with a central node and used to train a classification model.

Datasets and training procedure

We employ the Montgomery County X-ray Set and the Shenzhen Hospital X-ray set[1] [27, 83, 84]. The Montgomery Set contains 138 frontal chest X-ray images (80 negatives and 58 positives), captured with a Eureka stationary machine (CR) at 4020×4892 or 4892×4020 pixel resolution. The Shenzhen dataset was collected using a Philips DR Digital Diagnostic system. It includes 662 frontal chest X-ray images (326 negatives and 336 positives), with a variable resolution of approximately 3000×3000 pixels. In our federated learning setting,

[1]This dataset was released by National Library of Medicine, National Institute Of health, Bethesda, USA

each dataset is associated to a node. We employ 80% of each dataset to train a GAN and generate synthetic images using the proposed approach. The remaining 20% of each dataset is used for testing the model trained on the central node. Test labels are balanced: 65 positives and 65 negatives on the Shenzhen dataset, and 15 positives and 15 negatives on the Montgomery dataset.

We use StyleGAN2-ADA [94] for image generation on each node, because of its suitability in low-data regimes and its intrinsic latent projection mechanism. GANs are trained in a label-conditioned setting and yield a Fréchet inception distance (FID) of 21.36 and 55.38 on the Shenzhen and Montgomery datasets, respectively. Latent space projection is carried out as in [95] for 500 iterations. Spectral clustering is carried out using 20 clusters on the Shenzhen Dataset and 10 on Montgomery one, due to the difference in sizes. Centroid interpolation computes 9 intermediate points for each pair of centroids. The resulting synthetic datasets include 1,730 samples per class on Shenzen and 415 samples per class on Montgomery. On the central node, we use a ResNet-50 classifier, trained by minimizing a cross-entropy loss with mini-batch gradient descent using the Adam optimizer for a total of 1,000 epochs; mini-batch size is set to 64 and the learning rate is 10^{-6}. All images are resized to 256×256, and data augmentation is carried out with random horizontal flip and random 90-degree rotations. Experiments are performed on an NVIDIA GeForce RTX 3090, using PyTorch.

Experimental Results

We evaluate the performance of our approach by considering three different data usage scenarios:

1. **Real data:** the central server trains a classifier on the original joint dataset using images of all nodes (this is the standard supervised centralized setting).

2. **Synthetic (non privacy-preserving) data:** each node generates a synthetic training set by sampling from a GAN trained on the real data; synthetic samples are then used to train on the central server. No privacy-preserving mechanism is enforced: sampled images are drawn from the original distribution as learned by the GAN.

3. **Synthetic privacy-preserving data:** the training set for the central server is created by employing our privacy-preserving generation procedure (see Sect. 3.1.3).

Tab. 3.1 reports the test accuracy on each dataset under the above three scenarios. On the Shenzhen dataset, our approach is close to centralized training using all data, respectively

Table 3.1: **Classification accuracy on the test set of each dataset, in different training scenarios.**

Dataset	Training data	Accuracy
Shenzhen	Real	0.78
	Synthetic (non privacy-preserving)	0.82
	Synthetic (privacy-preserving)	0.75
Montgomery	Real	0.43
	Synthetic (non privacy-preserving)	0.60
	Synthetic (privacy-preserving)	0.60

0.75 and 0.78 classification accuracy. Interestingly, the non-privacy-preserving synthetic setting achieves even higher performance, which is explained by the larger number of training samples (662 real samples in Shenzhen, compared to 3,460 synthetic samples), confirming that sample synthesis helps making up for data scarcity — although in this case no precautions are taken to improve privacy. This phenomenon is even more evident on the smaller Montgomery dataset (138 samples), where the usage of synthetic data yields significantly improved accuracy (0.43 on the original dataset vs 0.60 on the synthetic one).

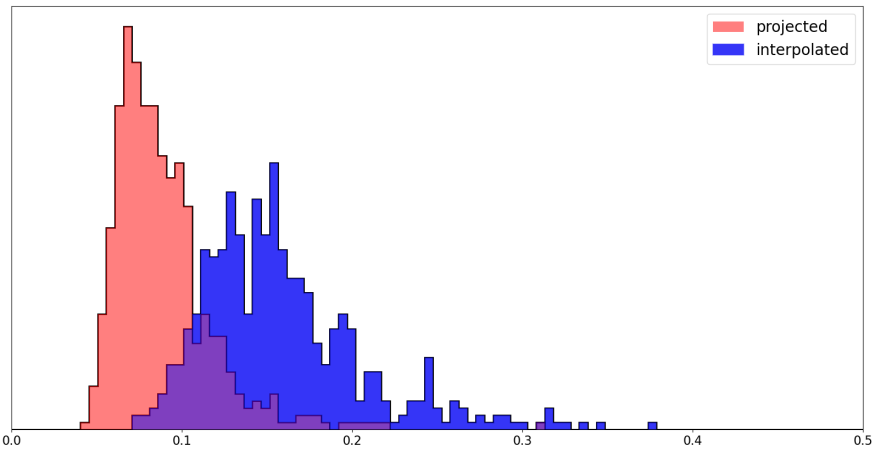

Figure 3.3: **Quantitative evaluation of generated images.** In red, LPIPS distance histogram between real images and the corresponding images obtained through latent space projection. In blue, LPIPS distance histogram between real images and the closest images generated with the proposed approach.

Privacy-preserving capabilities of the proposed approach are measured quantitatively by computing the LPIPS distance between real training images and a) their projected counterparts using StyleGAN2, and b) the most similar samples from the pool of images generated by our strategy. Ideally, we would expect that, when using a standard StyleGAN2 network,

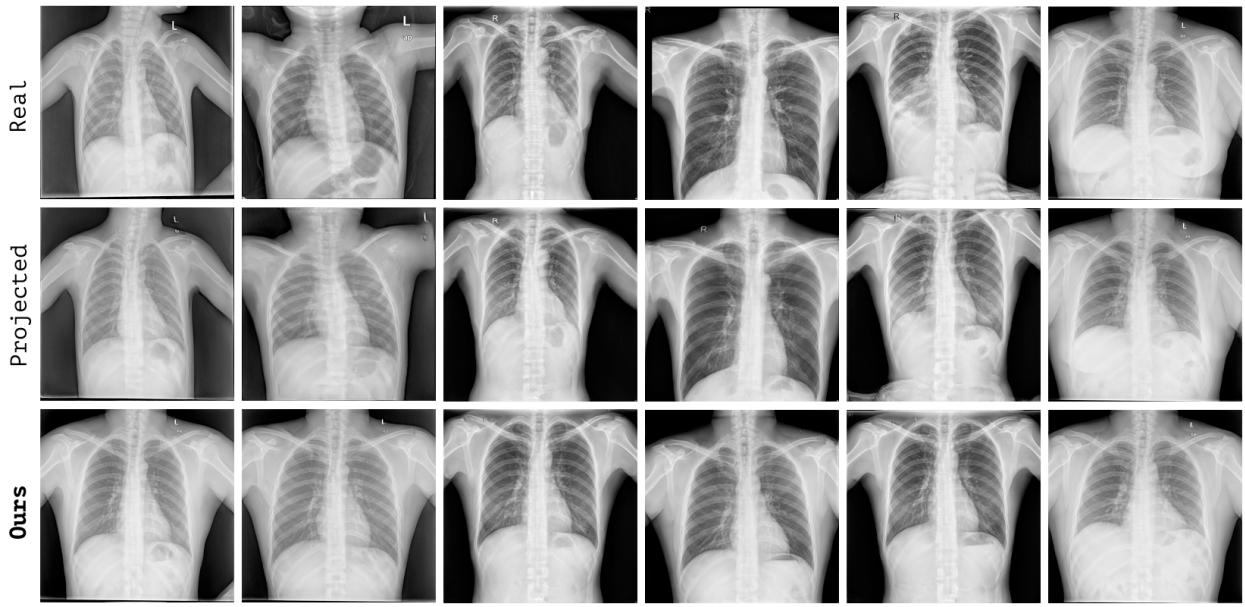

Figure 3.4: **Qualitative evaluation of generated images.** Top: real images from Shenzhen Dataset; middle: images generated by latent projection; bottom: most similar synthetic images obtained with the proposed method.

the latent projection procedure should be able to recover an image that the model has used at training time — which is undesirable, since knowledge of the model would allow an attacker to reconstruct original samples; we also expect that images synthesized through generative aggregation should be significantly dissimilar to any real sample. Indeed, LPIPS distance histograms in Fig. 3.3 show that a distribution shift can be observed between the two sets of measured distances: latent space projection of real images tends to produce samples with significantly smaller distances than those obtained with most similar synthetic images generated by our approach. This effect can be also appreciated qualitatively in the samples reported in Fig. 3.4, showing six images randomly sampled from the Shenzhen Dataset (top row) compared to their projection in the generator latent space (middle row) and the closest image in the aggregated dataset (bottom row).

3.1.5 Discussion

In this section we propose a synthetic data aggregation approach as an alternative to classic federated learning with gradient aggregation, which is subject to privacy concerns due to the risk of reconstructing the original inputs. Rather than training a central model by aggregating gradients from individual nodes, we propose to generate a synthetic dataset for each node and use the union of these datasets to train the central model. We tested our approach in a realistic scenario, using two X-Rays datasets for Tuberculosis classification,

simulating a system with two nodes and non-i.i.d. data. The results demonstrated the validity of our approach, which obtains comparable performance to those obtained when training on the union of all datasets. Moreover, we showed, both quantitatively and qualitatively, that the generated images exhibit visual features typical of the original data, while being significantly different from any actual real image, thus preventing to trace them back to individual patients. Still, this is a preliminary work: future developments will investigate its validity in the presence of more nodes or in the presence of i.i.d. distributions. The proposed aggregation method has the drawback of reducing the resulting dataset size by a factor of k. To address this limitation while preserving privacy guarantees, the next chapter will explore potential solutions.

3.1.6 Publications

Pennisi, M., Proietto Salanitri, F., Palazzo, S., Pino, C., Rundo, F., Giordano, D., & Spampinato, C. (2022, September). Gan latent space manipulation and aggregation for federated learning in medical imaging. In International Workshop on Distributed, Collaborative, and Federated Learning (pp. 68-78). Cham: Springer Nature Switzerland. [155]

3.2 A Privacy-Preserving Walk in the Latent Space of Generative Models for Medical Applications

3.2.1 Motivation

The success of deep learning for medical data analysis has demonstrated its potential to become a core component of future diagnosis and treatment methodologies. However, in spite of the efforts devoted to improve data efficiency [105], the most effective models still rely on large datasets to achieve high accuracy and generalizability. An effective strategy for obtaining large and diverse datasets is to leverage collaborative efforts based on data sharing principles; however, current privacy regulations often hinder this possibility. As a consequence, small private datasets are still used for training models that tend to overfit, introduce biases and generalize badly on other data sources addressing the same task [228]. As a mitigation measure, generative adversarial networks (GANs) have been proposed to synthesize highly-realistic images, extending existing datasets to include more (and more diverse) examples [153], but they pose privacy concerns as real samples may be encoded in the latent space. *K-same* techniques [87, 136] attempt to reduce this risk by following the *k-anonymity* principle [196] and replacing real samples with synthetic aggregations of groups of k samples. As a downside, these methods reduce the dataset size by a factor of k, which greatly limits their applicability.

To address this issue, we propose an approach, complementing *k-same* techniques, for generating an extended variant of a dataset by sampling a privacy-preserving walk in the GAN latent space. Our method directly optimizes latent points, through the use of an *auxiliary identity classifier*, which informs on the similarity between training samples and synthetic images corresponding to candidate latent points. This optimized navigation meets three key properties of data synthesis for medical applications: 1) *equidistance*, encouraging the generation of diverse realistic samples suitable for model training; 2) *privacy preservation*, limiting the possibility of recovering original samples, and, 3) *class-consistency*, ensuring that synthesized samples contain meaningful clinical information. To demonstrate the generalization capabilities of our approach, we experimentally evaluate its performance on two medical image tasks, namely, tuberculosis classification using the Shenzhen Hospital X-ray dataseet [27, 83, 84] and diabetic retinopathy classification on the APTOS dataset [97]. On both tasks, our approach yields classification performance comparable to training with real samples and significantly better than existing *k-same* techniques such as k-SALSA [87], while keeping the same robustness to membership inference attacks.

Contributions: 1) We present a latent space navigation approach that provides a large

amount of diverse and meaningful images for model training; 2) We devise an optimization strategy of latent walks that enforces privacy; 3) We carry out several experiments on two medical tasks, demonstrating the effectiveness of our generative approach on model's training and its guarantees to privacy preservation.

3.2.2 Related Work

Conventional methods to protect identity in private images have involved modifying pixels through techniques like masking, blurring, and pixelation [16, 166]. However, these methods have been found to be insufficient for providing adequate privacy protection [3]. As an alternative, GANs have been increasingly explored to synthesize high-quality images that preserve information from the original distribution, while disentangling and removing privacy-sensitive components [216, 223]. However, these methods have been mainly devised for face images and cannot be directly applicable to medical images, since there is no clear distinction between identity and non-identity features [87].

Recent approaches, based on the *k-same* framework [136], employ GANs to synthesize clinically-valid medical images principle by aggregating groups of real samples into synthetic privacy-preserving examples [87,155]. In particular, k-SALSA [87] uses GANs for generating retinal fundus images by proposing a local style alignment strategy to retain visual patterns of the original data. The main downside of these methods is that, in the strive to ensure privacy preservation following the *k-anonymity* [196] principle, they significantly reduce the size of the original dataset.

Our latent navigation strategy complements these approaches by synthesizing large and diverse samples, suitable for downstream tasks. In general, latent space navigation in GANs manipulates the latent vectors to create new images with specific characteristics. While many works have explored this concept to control semantic attributes of generated samples [21,95], to the best of our knowledge, no method has tackled the problem from a privacy-preservation standpoint, especially on a critical domain such as medical image analysis.

3.2.3 Method

The proposed **P**rivacy-preserving **LA**tent **N**avigation (**PLAN**) strategy envisages three separate stages: 1) GAN training using real samples; 2) latent privacy-preserving trajectory optimization in the GAN latent space; 3) privacy-preserving dataset synthesis for downstream applications. Fig. 3.5 illustrates the overall framework and provides a conceptual interpretation of the optimization objectives.

Formally, given a GAN generator $G : \mathcal{W} \rightarrow \mathcal{X}$, we aim to navigate its latent space $\mathcal{W}$ to

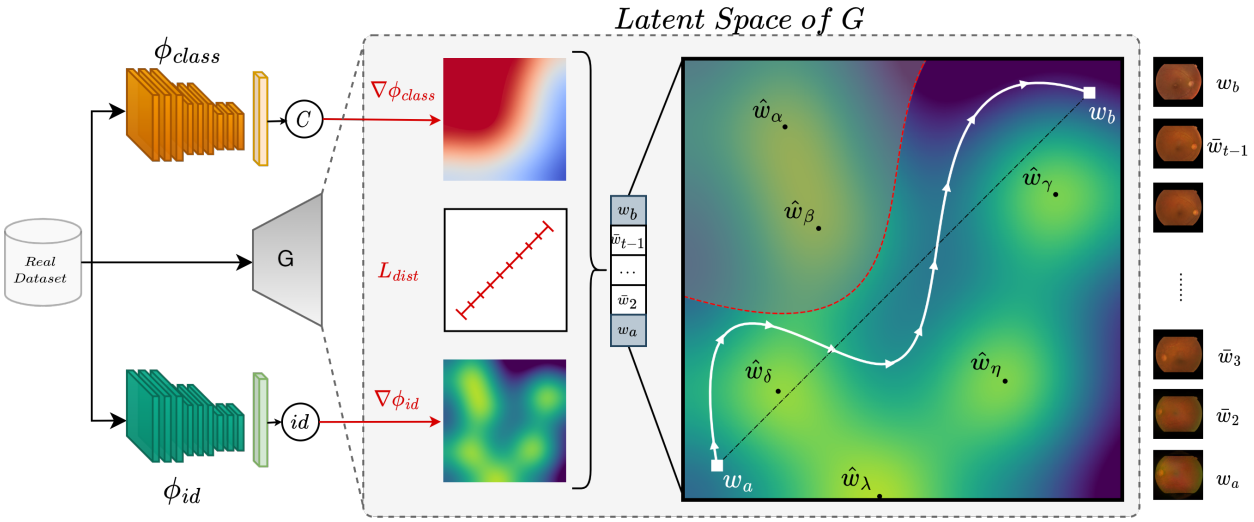

Figure 3.5: **Overview of the PLAN approach**. Using real samples, we train a GAN, an *identity classifier* ϕ_{id} and an *auxiliary classifier* ϕ_{class}. Given two arbitrary latent points, $\mathbf{w}_a$ and $\mathbf{w}_b$, PLAN employs ϕ_{id} and ϕ_{class} to gain information on latent space structure and generate a privacy-preserving navigation path (right image), from which synthetic samples can be sampled (far right images, zoom-in for details).

generate samples in image space $\mathcal{X}$ in a privacy-preserving way, i.e., avoiding latent regions where real images might be embedded. The expected result is a synthetic dataset that is safe to share, while still including consistent clinical features to be used by downstream tasks (e.g., classification).

Our objective is to find a set of latent points $\bar{\mathcal{W}} \subset \mathcal{W}$ from which it is safe to synthesize samples that are significantly different from training points: given the training set $\hat{\mathcal{X}} \subset \mathcal{X}$ and a metric d on $\mathcal{X}$, we want to find $\bar{\mathcal{W}}$ such that $\min_{\mathbf{x}\in\hat{\mathcal{X}}} d\left(G\left(\bar{\mathbf{w}}\right), \mathbf{x}\right) > \delta$, $\forall \bar{\mathbf{w}} \in \bar{\mathcal{W}}$, for a sufficiently large δ. Manually searching for $\bar{\mathcal{W}}$, however, may be unfeasible: generating a large $\bar{\mathcal{W}}$ is computationally expensive, as it requires at least $|\bar{\mathcal{W}}|$ forward passes through G, and each synthesized image should be compared to all training images; moreover, randomly sampled latent points might not satisfy the above condition.

To account for latent structure, one could explicitly sample away from latent vectors corresponding to real data. Let $\hat{\mathcal{W}}_i \subset \mathcal{W}$ be the set of latent vectors that produce near-duplicates of a training sample $\mathbf{x}_i \in \mathcal{X}$, such that $G(\hat{\mathbf{w}}_i) \approx \mathbf{x}_i$, $\forall \hat{\mathbf{w}}_i \in \hat{\mathcal{W}}_i$. We can thus define $\hat{\mathcal{W}} = \bigcup_{i=1}^{N} \hat{\mathcal{W}}_i$ as the set of latent points corresponding to all N samples of the training set: knowledge of $\hat{\mathcal{W}}$ can be used to move the above constraint from $\mathcal{X}$ to $\mathcal{W}$, by finding $\bar{\mathcal{W}}$ such that $\min_{\hat{\mathbf{w}}\in\hat{\mathcal{W}}} d\left(\bar{\mathbf{w}}, \hat{\mathbf{w}}\right) > \delta$, $\forall \bar{\mathbf{w}} \in \bar{\mathcal{W}}$. In practice, although $\hat{\mathcal{W}}_i$ can be approximated through latent space projection [5, 95] from multiple initialization points, its cardinality $|\hat{\mathcal{W}}_i|$ cannot be determined *a priori* as it is potentially unbounded.

From these limitations, we pose the search of seeking privacy-preserving latent points as a trajectory optimization problem, constrained by a set of objectives that mitigate privacy risks and enforce sample variability and class consistency. Given two arbitrary latent points (e.g., provided by a *k-same* aggregation method), $\mathbf{w}_a, \mathbf{w}_b \in \mathcal{W}$, we aim at finding a latent trajectory $\bar{\mathbf{W}}_T = [\mathbf{w}_a = \bar{\mathbf{w}}_1, \bar{\mathbf{w}}_2, \ldots, \bar{\mathbf{w}}_{T-1}, \mathbf{w}_b = \bar{\mathbf{w}}_T]$ that traverses the latent space from $\mathbf{w}_a$ to $\mathbf{w}_b$ in T steps, such that none of its points can be mapped to any training sample. We design our navigation strategy to satisfy three requirements, which are then translated into optimization objectives:

1. **Equidistance.** The distance between consecutive points in the latent trajectory should be approximately constant, to ensure sample diversity and mitigate mode collapse. We define the equidistance loss, $\mathcal{L}_{\text{dist}}$, as follows:

$$\mathcal{L}_{\text{dist}} = \sum_{i=1}^{T-1} \|\bar{\mathbf{w}}_i, \bar{\mathbf{w}}_{i+1}\|_2^2 \tag{3.3}$$

   where $\|\cdot\|_2$ is the L_2 norm. Note that without any additional constraint, $\mathcal{L}_{\text{dist}}$ converges to the trivial solution of linear interpolation, which gives no guarantee that the path will not contain points belonging to $\hat{\mathcal{W}}$.

2. **Privacy preservation.** To navigate away from latent regions corresponding to real samples, we employ an auxiliary network ϕ_{id}, trained on $\hat{\mathcal{X}}$ to perform *identity classification*. We then set the privacy preservation constraint by imposing that a sampled trajectory must maximize the uncertainty of ϕ_{id}, thus avoiding samples that could be recognizable from the training set. Assuming ϕ_{id} to be a neural network with as many outputs as the number of identities in the original dataset, this constraint can be mapped to a privacy-preserving loss, $\mathcal{L}_{\text{id}}$, defined as the Kullback-Leibler divergence between the softmax probabilities of ϕ_{id} and the uniform distribution $\mathcal{U}$:

$$\mathcal{L}_{\text{id}} = \sum_{i=1}^{T} \text{KL}[\phi_{id}(G(\bar{\mathbf{w}}_i)) \parallel \mathcal{U}(1/n_{\text{id}})] \tag{3.4}$$

   where n_{id} is the number of identities.

   This loss converges towards points with enhanced privacy, on which a trained classifier is maximally uncertain.

3. **Class consistency.** The latent navigation strategy, besides being privacy-preserving, needs to retain discriminative features to support training of downstream tasks on the

synthetic dataset. In the case of a downstream classification task, given $\mathbf{w}_a$ and $\mathbf{w}_b$ belonging to the same class, all points along a trajectory between $\mathbf{w}_a$ and $\mathbf{w}_b$ should exhibit the visual features of that specific class. Moreover, optimizing the constraints in Eq. 3.3 and Eq. 3.4 does not guarantee good visual quality, leading to privacy-preserving but useless synthetic samples. Thus, we add a third objective that enforces class-consistency on trajectory points. We employ an additional *auxiliary classification network* ϕ_{class}, trained to perform classification on the original dataset, to ensure that sampled latent points share the same visual properties (i.e., the same class) of $\mathbf{w}_a$ and $\mathbf{w}_b$. The corresponding loss $\mathcal{L}_{\text{class}}$ is as follows:

$$\mathcal{L}_{\text{class}} = \sum_{i=1}^{T} \text{CE}\left[\phi_{\text{class}}(G(\bar{\mathbf{w}}_i)), y\right] \tag{3.5}$$

where CE is the cross-entropy between the predicted label for each sample and the target class label y.

Overall, the total loss for privacy-preserving latent navigation is obtained as:

$$\mathcal{L}_{\text{PLAN}} = \mathcal{L}_{dist} + \lambda_1 \mathcal{L}_{id} + \lambda_2 \mathcal{L}_{label} \tag{3.6}$$

where λ_1 and λ_2 weight the three contributions.

In a practical application, we employ PLAN in conjunction with a privacy-preserving method that produces synthetic samples (e.g., a *k-same* approach). We then navigate the latent space between random pairs of such samples, and increase the size of the dataset while retaining privacy preservation. The resulting extended set is then used to train a *downstream classifier* ϕ_{down} on synthetic samples only. Overall, from an input set of N samples, we apply PLAN to $N/2$ random pairs, thus sampling $TN/2$ new points.

3.2.4 Experimental Results

We demonstrate the effectiveness and privacy-preserving properties of our PLAN approach on two classification tasks, namely, tuberculosis classification and diabetic retinopathy (DR) classification.

Training and evaluation procedure

Data preparation. For tuberculosis classification, we employ the Shenzhen Hospital X-ray set[2] [27,83,84] that includes 662 frontal chest X-ray images (326 negatives and 336 positives).

[2]This dataset was released by the National Library of Medicine, NIH, Bethesda, USA.

For diabetic retinopathy classification, we use the APTOS fundus image dataset [97] of retina images labeled by ophthalmologists with five grades of severity. We downsample it by randomly selecting 950 images, equally distributed among classes, to simulate a typical scenario with low data availability (as in medical applications), where GAN-based synthetic sampling, as a form of augmentation, is more needed. All images are resized to 256×256 and split into train, validation and test set with 70%, 10%, 20% proportions.

Baseline methods. We evaluate our approach from a privacy-preserving perspective and by its capability to support downstream classification tasks. For the former, given the lack of existing methods for privacy-preserving GAN latent navigation, we compare PLAN to standard linear interpolation. After assessing privacy-preserving performance, we measure the impact of our PLAN sampling strategy when combined to k-SALSA [87] and the latent cluster interpolation approach from [155] (LCI in the following) on the two considered tasks.

Implementation details. We employ StyleGAN2-ADA [94] as GAN model for all baselines, trained in a label-conditioned setting on the original training sets. For all classifiers (ϕ_{id}, ϕ_{class} and ϕ_{down}) we employ a ResNet-18 network [73]. Classifiers ϕ_{id} and ϕ_{class} are trained on the original training set, while ϕ_{down} (i.e., the task classifier, one for each task) is trained on synthetic samples only. For ϕ_{id}, we apply standard data augmentation (e.g., horizontal flip, rotation) and add five GAN projections for each identity, to mitigate the domain shift between real and synthetic images. ϕ_{down} is trained with a learning rate of 0.001, a batch size of 32, for 200 (Shenzhen) and 500 (APTOS) epochs. Model selection is carried out at the best validation accuracy, and results are averaged over 5 runs. When applying PLAN on a pair of latent points, we initialize a trajectory of $T = 50$ points through linear interpolation, and optimize Eq. 3.6 for 100 steps using Adam with a learning rate of 0.1; λ_1 and λ_2 are set to 0.1 and 1, respectively. Experiments are performed on an NVIDIA RTX 3090.

Results

To measure the privacy-preserving properties of our approach, we employ the *membership inference attack* (MIA) [188], which attempts to predict if a sample was used in a classifier's training set. We use attacker model and settings defined in [88, 142], training the attacker on 30% of the training set (seen by PLAN through ϕ_{id} and ϕ_{class}) and 30% of the test set (unseen by PLAN); as a test set for MIA, we reserve 60% of the original test set, leaving 10% as a validation set to select the best attacker. Ideally, if the model preserves privacy, the attacker achieves chance performance (50%), showing inability to identify samples used for training. We also report the FID of the generated dataset, to measure its level of realism, and the mean of the minimum LPIPS [231] ("mmL" for short) distances between each generated sample and its closest real image, to measure how generated samples differ from

real ones. We compare PLAN to a linear interpolation between arbitrary pairs of start and end latent points, and compute the above measures on the images corresponding to the latent trajectories obtained by two approaches. We also report the results of the classifier trained on real data to provide additional bounds for both classification accuracy and privacy-preserving performance.

Table 3.2: **Comparison between the downstream classifier ($\phi_{\mathbf{down}}$) model trained with real samples and those trained with synthetic samples generated from the linear path and privacy path, respectively.**

	Shenzhen				**Aptos**			
	Acc. (%)(↑)	MIA (↓)	FID (↓)	mmL (↑)	Acc. (%)(↑)	MIA (↓)	FID (↓)	mmL (↑)
Real	81.23±1.03	71.41±3.59	–	–	50.74±2.85	73.30±4.04	–	–
Linear	82.14±1.40	56.28±1.60	63.85	0.125	41.58±2.11	50.53±3.06	**85.17**	0.118
PLAN	**83.85**±1.33	**50.13**±3.99	**63.22**	**0.159**	**46.95**±3.06	**48.51**±2.85	90.81	**0.131**

Results in Table 3.2 demonstrate that our approach performs similarly to training with real data, but with higher accuracy with respect to the linear baseline. Privacy-preserving results, measured through MIA and mmL, demonstrate the reliability of our PLAN strategy in removing sensitive information, reaching the ideal lower bound of MIA accuracy.

Fig. 3.6 shows how, for given start and end points, PLAN-generated samples keep high quality but differ significantly from real samples, while latent linear interpolation may lead to near-duplicates. This is confirmed by the higher LPIPS distance between generated samples and the most similar real samples for PLAN.

After verifying the generative and privacy-preserving capabilities of our approach, we evaluate its contribution to classification accuracy when combined with existing *k-same* methods, namely k-SALSA [87] and LCI [155]. Both methods apply latent clustering to synthesize a privacy-preserving dataset, but exhibit low performance transferability to classification tasks, due to the reduced size of the resulting synthetic dataset. We carry out these experiments on APTOS, using $k = 5$ and $k = 10$, for comparison with [87][3]. Results are given in Table 3.3 and show how our PLAN strategy enhances performance of the two baseline methods, reaching performance similar to training the retinopathy classifier with real samples (i.e., 50.74 on real data vs 44.95 when LCI [155] is combined with PLAN) and much higher than the variants without PLAN. We also measured MIA accuracy between the variants with and without PLAN, and we did not observe significant change among the different con-

[3]Values of k smaller than 5 led to vulnerabilities to MIA on APTOS, as shown in [87].

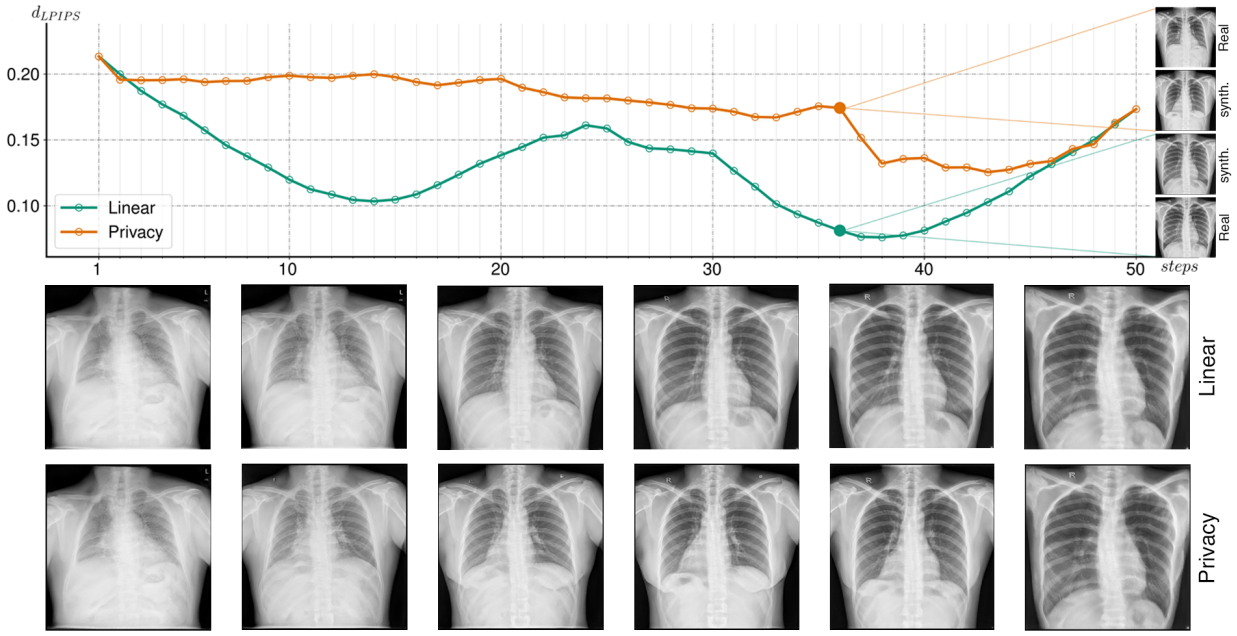

Figure 3.6: **Linear vs PLAN navigation between two arbitrary points.** For each step of the latent trajectory, we compute the LPIPS distance between each synthetic sample and its closest real image. On the right, a qualitative comparison of images at step 35 and their closest real samples: the synthetic image obtained with PLAN differs significantly from its closest real sample; in linear interpolation, synthetic and real samples look similar. Bottom images show synthetic samples generated by linear interpolation and PLAN at the same steps (zoom-in for details).

figurations: accuracy was at the chance level in all cases, suggesting their privacy-preserving capability.

Table 3.3: **Impact of our navigation strategy on k-same methods on the APTOS dataset.** Performance are reported in terms of accuracy.

	k-SALSA [87]	k-**SALSA +PLAN**	LCI [155]	**LCI +PLAN**
k = 5	25.58±6.32	**36.59**±3.48	38.74±4.51	**43.16**±2.71
k = 10	27.47±3.42	**34.21**±1.62	36.42±3.77	**44.95**±1.61

3.2.5 Discussion

In this section we presented PLAN, a latent space navigation strategy designed to reduce privacy risks when using GANs for training models on synthetic data. Experimental results,

on two medical image analysis tasks, demonstrate how PLAN is robust to membership inference attacks while effectively supporting model training with performance comparable to training on real data. Furthermore, when PLAN is combined with state-of-the-art *k-anonymity* methods (e.g. the aggregation technique presented in the previous section), we observe a mitigation of performance drop while maintaining privacy-preservation properties. Future research directions will address the scalability of the method to large datasets with a high number of identities, as well as learning latent trajectories with arbitrary length to maximize privacy-preserving and augmentation properties of the synthetic datasets. This concludes the objective of this chapter demonstrating that is possible to generate privacy safe samples also in the context of medical federations.

3.2.6 Publications

Pennisi, M., Proietto Salanitri, F., Bellitto, G., Palazzo, S., Bagci, U., & Spampinato, C. (2023, October). A privacy-preserving walk in the latent space of generative models for medical applications. In International Conference on Medical Image Computing and Computer-Assisted Intervention (pp. 422-431). Cham: Springer Nature Switzerland.. [154]

Chapter 4

Bridging continual and federated learning

This chapter represents the final stage in the integration of Continual Learning and Federated Learning proposed by this thesis. In previous chapters, we examined each approach individually and highlighted their shared challenges, such as handling non-i.i.d. data distributions. Additionally, we observed that a fully decentralized federated setting, without a central server, can be reframed as a continual learning process, where the model sequentially visits each node, similar to how a continual learning model learns from a sequence of tasks.

In this chapter, we bring these approaches together by adopting a common continual learning strategy, i.e. Experience Replay, within a realistic, fully decentralized federated medical setting, enhanced with generative models for data synthesis. More specifically in FedER, presented in Sec. 4.1, the model and a buffer of privacy safe synthetic images are exchanged between nodes (e.g. the receiver node uses the buffer of the previous node to retain knowledge and mitigate forgetting). In a similar fashion, based on the promising results achieved with this framework, FedRewind (Sec. 4.2), exchanges the model back to the sender node to retain the knowledge. This second strategy is tested on the more complex scenario of Federated Continual Learning, where each node has its own private dataset that evolves over time, creating an additional layer of distributional shift.

4.1 FedER: Federated Learning through Experience Replay and privacy-preserving data synthesis

4.1.1 Motivation

Recent advances of deep learning in the medical imaging domain have shown that, while data-driven approaches represent a powerful and promising tool for supporting physicians' decisions, the availability of large-scale datasets plays a key role in the effectiveness and reliability of the resulting models [42, 82, 208]. However, the curation of large medical imaging datasets is a complex task: data collection at single institutions is relatively slow and the integration of historical data may require significant efforts to deal with different data formats, storage modalities and acquisition devices; moreover, medical institutions are often reluctant to share their own data, due to privacy concerns. As a consequence, this affects the quality, reliability and generalizability of models trained on local datasets, which unavoidably suffer from bias and overfitting issues, reducing the ability to address future data distribution shifts [228]. In order to overcome the lack of large-scale datasets, methodological solutions can be adopted: in particular, federated learning [217] encompasses a family of strategies for distributed training over multiple nodes, each with its own private dataset, which typically communicate with a central node by sending local model updates, used to train the main model. In this scenario, no data is explicitly shared between nodes, thus addressing the required privacy issues. However, this family of techniques generally performs well when dataset distributions are approximately *i.i.d.* and local gradients/models contribute to learning shared features: unfortunately, in practice this hypothesis rarely holds, due to differences in the acquisition and in the clinical nature of data collected by multiple institutions. Moreover, the presence of a central node, besides representing a single point of failure, requires that all nodes trust it to correctly and fairly treat updates from all sources: indeed, privacy issues arise when transferring local updates to the "semi-honest" central node [58], which might attempt to reconstruct original inputs from gradients or parameter variations [67, 232, 235]. To address the above limitations, we present *FedER*, a federated learning approach that, leveraging experience replay from continual learning [23,163,169,172] and generative models [68, 95, 138], proposes a principled way for training local models that approximately converge to the same decisions, without the need of a shared model architecture and of central coordination. *FedER* also enforces privacy preservation through the transmission of synthetic data generated in a way to obfuscate real data patterns.
Specifically, FedER's learning strategy envisages multiple nodes that initially train their local models and a GAN on their own datasets. The GAN will be used in order to generate

a privacy-preserving synthesized version of the dataset (buffer). Once local training is completed in a node, its model and the "buffer" of generated synthetic data are sent to a random node of the network. The receiving node then adapts the incoming model using its own data and the received buffer data, in order to limit model's forgetting. Data privacy is ensured through a privacy-preserving generative adversarial network (GAN) that employs a specific loss designed to maximize the distance from real data, while keeping a high level of realism and — as importantly — clinically-consistent features, in order to allow models to be trained effectively.

FedER is tested on two tasks, simulating a *non-i.i.d.* medical scenario: 1) classification of tuberculosis from X-ray data, using Montgomery County and Shenzhen Hospital datasets [27, 83, 84], and 2) melanoma classification using skin images of the ISIC 2019 dataset [41, 43, 199]. The experimental setting is specifically designed to emulate a realistic medical *non-i.i.d.* scenario, where each node in the federation uses its own dataset. This is in stark contrast with common procedures where non-*i.i.d.* distributions are simulated by splitting a single source dataset. Results show how our approach is able to reach performance similar to using centralized training on all real data together in a single node, while outperforming current state-of-the-art methods, such as FedAvg [134], FedProx [116] and FedBN [118]. Privacy-preserving capabilities are measured quantitatively by evaluating LPIPS distance [231] between real images and samples generated, respectively, through latent space optimization on a standard GAN and by the proposed approach. Qualitatively, we also show several examples of generated images with corresponding closest match in the real dataset, demonstrating significant differences that prevent tracing back to the original real distribution.

In summary, the overall contributions of the proposed work are the following:

- We propose a decentralized federated learning strategy, based on continual learning principles, designed for medical imaging data, which outperforms server-based federated learning approaches and yields performance similar to standard (non-federated) training settings. Furthermore, experience replay allows local node models to converge to the same decisions, thus making the whole approach behave similarly to server-based aggregation models.

- We propose a GAN-based privacy-preserving mechanism that supports synthetic data sharing through a GAN-based technique designed to minimize patient information leak. This is different from most privacy-preserving techniques based on differential privacy, which degrades performance due to added noise.

- Most approaches for model aggregation in federated learning employ gradient/parameter

averaging. These solutions completely neglect any similarity or dissimilarity between merged features, possibly resulting in interference that harm convergence. FedER, instead, takes feature semantics into account when merging models: if a node receives a model that extracts useful features for the local dataset, these can be readily employed and re-used, without the risk of randomly averaging them with other less important features. *FedER*, thus, surpasses the common and straightforward weight/gradient averaging paradigm, replacing it with a principled way for knowledge transfer, which relaxes two of the constraints of the leading federated learning approaches: the presence of a central node and model homogeneity.

4.1.2 Related Work

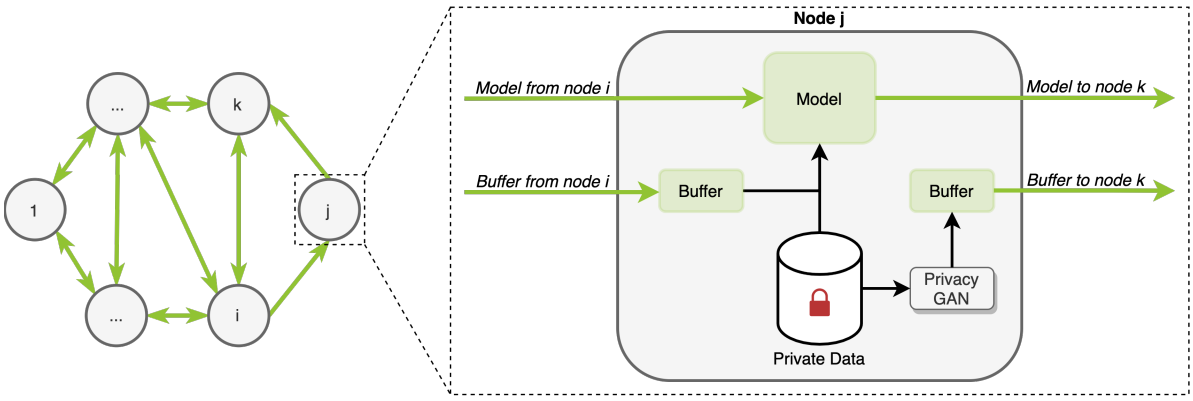

Figure 4.1: **Overview of FedER learning strategy.** Each node initially trains a *privacy-preserving GAN*, that is used to sample synthetic data from the local distribution, without retaining features that may be used to identify patients. Then, each node iteratively receives the local model and a buffer of synthetic samples from a random node, and fine-tunes the received model on its own private data, using the buffer to prevent forgetting of previously-learned features.

Federated Learning (FL) [134] has recently emerged as a family of distributed learning strategies that allow nodes to keep training data private, while supporting the creation of a shared model. In a typical FL setting, a central server sends a model to a set of client nodes; each node fine-tunes the model on its own data, then sends local model updates back to the server; the server aggregates the updates by all nodes into the global model, which is sent back to nodes iteratively until convergence. Given the constraints existing in the medical domain, especially in terms of data sharing, it represents an appropriate test-bench for federated learning methods [44, 61, 117, 175]. The most straightforward way to aggregate information from multiple nodes is through averaging local models of each client, as proposed in FedAvg [134] and FedProx [116]. However, statistical data heterogeneity is an issue as it

may lead to catastrophic forgetting [69,91]. FedCurv [187] addresses this limitation by adding a penalty term to the loss driving the local models to a shared optimum. FedMA [206] builds a shared global model in a layer-wise manner by matching and averaging hidden elements with similar feature extraction signatures. Our method differs from existing feature integration approaches in that, instead of averaging model updates or gradients, which can be subject to input reconstruction attacks [67,215,235], each node attempts to learn features that perform well on its own dataset while retaining knowledge from other nodes, in a more principled way than parameter averaging. The strategy of fitting the global model to local data is also sought by the recent federated *personalized methods*. FedBN [118], for instance, keeps batch normalization layers private, while other model parameters are aggregated by the central node.

However, the presence of a central node that aggregates local updates simplifies the communication protocol when the number of clients is very large (thousands or millions), but introduces several downsides: it represents a single point of failure; it can become a bottleneck when the number of clients increases [120]; in general, it may not always be available or desirable in collaborative learning scenarios [91]. Our proposed method deals with *decentralized federated learning*, in which the central node is replaced by peer-to-peer communication between clients: there is no longer a global shared model as in standard FL, but the communication protocol is designed so that all local models approximately converge to the same solution. Decentralized learning is particularly suitable to application in the medical domain, where the number of nodes (i.e., institutions) is relatively low; however, research is still ongoing, and no effective solutions have been established. In [108], a Bayesian approach is proposed to learn a shared model over a graph of nodes, by aggregating information from local data with the model of each node's one-hop neighbors. A secure weight averaging algorithm is proposed in [213], where model parameters are not shared between nodes, but all converge to the same numerical values (with the disadvantages associated to parameter averaging with *non-i.i.d.* data distributions). Other approaches implement different communication strategies based on parameter sharing (e.g., decentralized variants on FedAvg [134,195]). In general, many of the existing solutions do not target, nor are they tested on, the medical domains — most employ toy datasets, such as MNIST and CIFAR10. Two works, similar in the decentralized learning spirit to ours, are proposed in [66, 175], where use cases of decentralized and swarm learning for medical image segmentation are presented. However, like other approaches, they adopt simple parameter averaging to integrate features or predictions from multiple nodes.

4.1.3 Method

Overview

An overview of FedER is shown in Fig. 4.1. In this scenario, a *federation* consists of a set of N peer nodes, each owning a private dataset.

Before the decentralized training algorithm is started, each node internally trains a *privacy-preserving generative adversarial network*, which is used to generate synthetic samples from its private data distribution. The training objective of the GAN is designed to enforce the constraint that sampled data do not include privacy-sensitive information, while maintaining the clinical features required for successful training.

At each round of decentralized training, each node receives a model and a set of synthetic samples — "buffer" — from a random node in the federation. The input model to the node is fine-tuned on both the private dataset and the buffer, in a way that is reminiscent of experience replay techniques in continual learning (e.g., [23]), in order to learn features that transfer between nodes and that can handle non-*i.i.d.* distributions. At the end of each round (i.e., after performing several training iterations), the locally-trained model is sent to a randomly-chosen successor node together with a buffer of local synthetic samples, and the whole procedure is repeated.

In this work we specifically address the problem of federated learning for medical image classification; thus, the method is presented by considering this task, but the whole strategy can be applied to any other task without losing generalization.

Privacy-preserving GAN

In the proposed method, nodes exchange both models and data, implementing a knowledge transfer procedure based on experience replay (see Sect. 4.1.3 below). Of course, sharing real samples would go against federated learning policies; hence, exchanged samples are generated so that they are representative of the local data, while taking precautions against privacy violations — which may happen, for instance, if the generative model overfits the source dataset.

Formally, we assume that each node n_i, from a set of N nodes, owns a private dataset $\mathcal{D}_i = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_M, \mathbf{y}_M)\}$, where each $\mathbf{x}_j \in \mathcal{X}$ represents a sample in the dataset, and each $\mathbf{y}_j \in \mathcal{Y}$ represents the corresponding target[1]. The local dataset can then be used to train a conditional GAN [138], consisting of a generator G, that synthesizes

[1]The proposed approach is task-agnostic, as long as it is possible to sample from the $\mathcal{Y}$ distribution. For simplicity, within the scope of this work, we will focus on classification tasks, and we will assume that targets are class labels.

samples for a given label by modeling $P(\mathbf{x}|\mathbf{y},\mathbf{z})$), where $\mathbf{z} \in \mathcal{Z}$ is a random vector sampled from the generation latent space, and a discriminator D, which outputs the probability of an input sample being real, modeling $P(\text{real}|\mathbf{x},\mathbf{y})$. The standard GAN formulation introduces a discrimination loss, which trains D to distinguish between real and synthetic samples:

$$\mathcal{L}_D = -\mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\log\left(D\left(\mathbf{x},\mathbf{y}\right)\right)\right] - \mathbb{E}_{\mathbf{z},\mathbf{y}}\left[\log\left(1 - D\left(G\left(\mathbf{z},\mathbf{y}\right),\mathbf{y}\right)\right)\right], \tag{4.1}$$

and a generation loss, which trains G to synthesize samples that appear realistic to the discriminator:

$$\mathcal{L}_G = -\mathbb{E}_{\mathbf{z},\mathbf{y}}\left[D\left(G\left(\mathbf{z},\mathbf{y}\right),\mathbf{y}\right)\right]. \tag{4.2}$$

While it has been theoretically proven that, at convergence, the distribution learned by the generator matches and generalizes from the original data distribution [69], unfortunately GAN architectures may be subject to training anomalies, including mode collapse and overfitting: as a consequence, the basic GAN formulation may lead to the generation of samples that are near duplicates of the original samples, which would be unacceptable in a federated learning scenario.

In order to mitigate this risk, we introduce a *privacy-preserving loss*, enforcing the generation of samples that do not retain potentially sensitive information, but still include features that are clinically relevant to the target $\mathbf{y}$ of the synthetic sample. In other words, if $\mathbf{y}$ encodes generic features for the diagnosis of a certain disease, we want the generator to learn how to synthesize samples conditioned by $\mathbf{y}$, that exhibit evidence of that disease but cannot be traced back to any of the dataset's samples of the same disease.

To do so, our privacy-preserving loss aims at penalizing the model proportionally to the similarity between pairs of real and synthetic samples. We measure "similarity" by means of the LPIPS metric [231], which has been shown to capture perceptual similarity by calibrating the distance between feature vectors extracted from a pre-trained VGG model [190].

In practice, given a batch of real samples $\left\{\mathbf{x}_1^{(r)}, \mathbf{x}_2^{(r)}, \ldots, \mathbf{x}_b^{(r)}\right\}$ and a batch of synthetic samples $\left\{\mathbf{x}_1^{(s)}, \mathbf{x}_2^{(s)}, \ldots, \mathbf{x}_b^{(s)}\right\}$, the privacy-preserving loss term is computed as:

$$\mathcal{L}_{\text{PP}} = \frac{1}{b}\sum_{\mathbf{x}^{(r)}}\sum_{\mathbf{x}^{(s)}} d_L\left(\mathbf{x}^{(r)}, \mathbf{x}^{(s)}\right), \tag{4.3}$$

where d_L is the LPIPS distance defined as:

$$d_{\text{L}}(\mathbf{x}^{(r)}, \mathbf{x}^{(s)}) = \sum_i w_i \cdot \|\phi_i(\mathbf{x}^{(r)}) - \phi_i(\mathbf{x}^{(s)})\|_2 \tag{4.4}$$

where ϕ_i represents the feature maps extracted from the i^{th} layer of a deep neural network and w_i is a weight learned to reflect the perceptual importance of that layer. Note that, in this formulation, we ignore the $\mathbf{y}$ targets associated to each $\mathbf{x}$: we want to prevent the model from generating near-duplicates of real samples in general, regardless of class correspondence. Also, we intentionally employ a pairwise metric on samples, rather than an aggregated metric such as Frêchet Inception Distance [76], since we want to prevent similarity between samples, not between distributions, which would conflict with the GAN objective.

The resulting new loss for the Generator is a combination of Eq. 4.2 and Eq. 4.3:

$$\mathcal{L}_{G\text{-PP}} = \mathcal{L}_G - \alpha \mathcal{L}_{\text{PP}} \tag{4.5}$$

where $\mathcal{L}_{\text{PP}}$ is sign reversed as we want to maximize Eq. 4.3, while α is a hyperparameter used to balance the two terms.

The combined effect of the three loss terms — $\mathcal{L}_D$, $\mathcal{L}_G$, $\mathcal{L}_{\text{PP}}$ — pushes the generator to explore the sample space to match the dataset distribution, while "avoiding" latent space mappings that would project to actual real samples.

Federated learning with experience replay

Current approaches for federated learning are mostly based on parameter averaging (e.g., FedAvg), which is, however, a straightforward way to combine knowledge from multiple sources: feature locations are not aligned over different models and may be disrupted by updates, before slowly converging to consensus: hypothetically, two models could learn the same set of features at different locations of the same layer, to only have them cancel each other when averaging. In a decentralized scenario, this issue is even exacerbated, due to the lack of an entity that enforces global agreement on node features.

In our approach, we address this problem by taking inspiration from continual learning strategies [109] that learn how to perform a task with a non-*i.i.d* data stream without forgetting previously-learned knowledge: as a consequence, models are encouraged to reuse and adapt features so that they can equally serve the current and previous tasks. Analogously, in the federated learning setting, the objective is to train a global model trained on disjoint non-*i.i.d.* data distributions coming from different nodes.

Given these premises, we define a federated learning strategy where a node receives another node's model and surrogate data (generated through our privacy-preserving GAN) — the *"previous task"* — and fine-tunes that model on its own private date — the *"current task"* — while using received synthetic data as a reference to what is necessary to retain/adapt from the knowledge learned by the previous node. The idea is to build for each node a

model able to tackle its internal data while not forgetting about the data seen in previous nodes/iterations.

We first introduce the terminology used in the method's description. In our approach, we define a *set of N tasks* $\mathcal{T} = (T_1, T_2, \ldots, T_N)$, where T_i is the task to be solved within node n_i.

Definition 1. Task T_i aims at optimizing a model M_i, parameterized by $\boldsymbol{\theta}_i$, on dataset $\mathcal{D}_i$ residing on node n_i and that cannot be shared to other nodes.

Definition 2. A buffer $\mathcal{B}_i$ is a set of synthetic images, drawn from a latent space learned through a generative model $\mathcal{G}_i$ using data $\mathcal{D}_i$ available on node n_i.

Definition 3. Training is organized in parallel *rounds*. At the end of round r, each node n_i produces a model M_i^r trained on dataset $\mathcal{D}_i$ and on a buffer $\mathcal{B}_j$, received from another node n_j, to optimize an objective $\mathcal{L}$, i.e., to find $\arg\min_{\boldsymbol{\theta}_i^r} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_i\cup\mathcal{B}_j}[\mathcal{L}(M_i^r(\mathbf{x}, \boldsymbol{\theta}_i^r), \mathbf{y})]$. For each training round, all nodes in parallel share to/receive from other nodes, buffer of synthetic images and trained models.
In the following, we describe our method (whose graphical representation is given in Fig. 4.1) from the point of view of a single node n_j. At a given round r, training for node n_j can be seen as learning a new task T_j, from dataset $\mathcal{D}_j$, in a continual learning setting by finetuning the incoming model $\mathcal{M}_i^{r-1}$ (with parameters $\boldsymbol{\theta}_i^{r-1}$) on $\mathcal{D}_j$ and on the incoming buffer $\mathcal{B}_i$ in order to learn T_j while mitigating the forgetting of T_i. Thus, unlike other federated learning approaches, each node does not have its own local model: as the decentralized learning strategy proceeds, a node iteratively receives a model from another node and updates it with local information, while preserving previously-learned knowledge, before sending it to the next node. Formally, the loss function for model $\mathcal{M}_j^r$ in node n_j at round r is given as:

$$\mathcal{L}(\boldsymbol{\theta}_j^r) = \lambda\, \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_j}[\mathcal{L}(M_j^r(\mathbf{x}, \boldsymbol{\theta}_j^r), \mathbf{y})] + (1-\lambda)\, \mathbb{E}_{(\mathbf{x}',\mathbf{y}')\sim\mathcal{B}_i}[\mathcal{L}(M_j^r(\mathbf{x}', \boldsymbol{\theta}_j^r), \mathbf{y}')] \tag{4.6}$$

where λ controls the importance between real samples from the local dataset D_i and replayed synthetic samples from node n_i. Note that, for a given n_j, the predecessor node n_i is not fixed: in a practical asynchronous implementation, a node may receive a model and buffer from any random node in the federation at any time, using queues to handle incoming data.

After optimizing the $\mathcal{L}(\boldsymbol{\theta}_j^r)$ objective through mini-batch gradient descent for a certain number of training iterations, the resulting model $M_j^r(\boldsymbol{\theta}_j^r)$, with updated parameters $\boldsymbol{\theta}_j^r$,

is sent to a random node n_k of the federation, along with a buffer $\mathcal{B}_j$ of locally-generated synthetic samples. The number of training rounds/iterations and the size of the buffer is discussed in the next section.

Then, the general federated model $\mathcal{M}$, after all training rounds, is given by the union of all the N node models, i.e., $\mathcal{M} = M_1 \cup M_2 \cup \cdots \cup M_N$. However, experimental results, reported below in Sect. 4.1.4, demonstrate that all models converge to similar decisions, thus each node model can be considered as a general model for the entire network.

To ease the understanding of the whole training strategy we also report the algorithm pseudo-code in Alg. 1.

Algorithm 1: FedER Learning Procedure

Notations *The N nodes are indexed by n_i; E is the number of local epochs for each round. R the total round of communications between nodes.*
Each node n_i contains:
$\mathcal{D}_i$ Private Dataset
$\mathcal{G}_i$ Generator (privacy-preserving) trained on D_i
$\mathcal{M}_i^r$ Model for node n_i at round r
$\mathcal{B}_i$ Synthetic data buffer sampled using G_i

```
// Before Federated Training
```
for *each node* $n_i \in N$ **do**
  Train $\mathcal{G}_i$ on $\mathcal{D}_i$
  Generate Buffer $\mathcal{B}_i$ using $\mathcal{G}_i$
  Train $\mathcal{M}_i^0$ on $\mathcal{D}_i$
end

```
// Federated Training
```
for *each round* $r = 1, 2, ..., R$ **do**
  for *each node* $n_j \in N$ *in parallel* **do**
    Send $\mathcal{M}_j^{r-1}$, $\mathcal{B}_j$ to a node $n_k \in \{N \setminus n_j\}$
    Receive $\mathcal{M}_i^{r-1}$, $\mathcal{B}_i$ from a node $n_i \in \{N \setminus n_j\}$
    $\mathcal{M}_j^r \leftarrow \mathcal{M}_i^{r-1}$
    Train $\mathcal{M}_j^r$ on $\{\mathcal{D}_j \cup \mathcal{B}_i\}$ for E epochs
  end
end

4.1.4 Experimental Results

We test FedER on two applications simulating real case scenarios with multiple centers holding, and not sharing, their own data: 1) tuberculosis classification from X-ray images

Table 4.1: **Rounds and epochs in FedER.** Results (mean ± standard deviation) obtained with 5-fold cross-validation. Buffer size = 512.

		Tuberculosis		**Melanoma**		
		Shenzhen	**Montgomery**	**BCN**	**HAM**	**MSK4**
Rounds	Epochs	Accuracy	Accuracy	Accuracy	Accuracy	Accuracy
10	1	82.39 ± 6.91	56.13 ± 3.03	76.73 ± 2.07	82.24 ± 4.01	67.93 ± 4.84
	10	82.86 ± 2.44	86.73 ± 4.22	83.83 ± 1.96	84.72 ± 2.29	73.67 ± 2.59
	100	83.56 ± 1.72	90.79 ± 3.92	85.51 ± 1.85	88.65 ± 1.12	71.81 ± 2.04
100	1	83.31 ± 2.59	88.71 ± 3.82	78.94 ± 2.55	87.34 ± 1.62	72.07 ± 3.45
	10	85.22 ± 2.42	89.72 ± 3.46	84.62 ± 1.40	85.05 ± 1.62	73.72 ± 2.41
	100	87.10 ± 2.31	91.50 ± 2.60	86.06 ± 0.96	89.26 ± 1.11	72.41 ± 1.53

using two different datasets, and 2) skin lesion classification with three different datasets. In this section we present the employed benchmarks, the training procedure and report the obtained results to demonstrate the advantages of the proposed approach w.r.t. the state-of-the-art.

Datasets

X-ray image datasets for tuberculosis classification. We assume two separate nodes in the federation: one with the Montgomery County X-ray set and another one with the Shenzhen Hospital X-ray set [27, 83, 84]. The Montgomery Set consists of 138 frontal chest X-ray images (80 negatives and 58 positives), captured with a Eureka stationary machine (CR) at 4020×4892 or 4892×4020 pixel resolution. The Shenzhen dataset was collected using a Philips DR Digital Diagnostic system. It includes 662 frontal chest X-ray images (326 negatives and 336 positives), with a variable resolution of approximately 3000×3000 pixels.

Skin lesion classification. We employ the ISIC 2019 challenge dataset, which contains 25,331 skin images belonging to nine different diagnostic categories. In this case, we assume a federation with three nodes as data provided belongs to three different sources: 1) the BCN20000 [43] dataset, consisting of 19,424 images of skin lesions captured from 2010 to 2016 in the Hospital Clínic in Barcelona; 2) the HAM10000 dataset [199], which contains 10,015 skin images collected over a period of 20 years from two different sites, the Department of Dermatology at the Medical University of Vienna, Austria, and the skin cancer practice of Cliff Rosendahl in Queensland, Australia; 3) the MSK4 [41] dataset, which is anonymous and includes 819 samples. Among all skin lesion classes, we only consider the melanoma

class, posing the problem as a binary classification task.
In all tasks and datasets we adopt 80% of the available data to train both the privacy-preserving GAN and the classification model, while the remaining 20% of each dataset is used as test set. Test sets are also balanced w.r.t. the label to avoid performance biases due to class imbalance. For all tested federated methods (including state-of-the-art ones), model selection is carried out through with 5-fold cross-validation on the training set, as a grid search on number of training rounds, number of rounds per epoch and learning rate. For FedProx [116], we also include the μ hyperparameter.

Training procedure and metrics

Federated training

In all settings, we employ ResNet-18 as classification model, trained by minimizing the cross-entropy loss with mini-batch gradient descent using the Adam optimizer. Mini-batch size is set to 32 and 8 for the Shenzhen and Montgomery datasets, respectively, and to 64 for skin lesion datasets. The learning rate was found, through cross-validation, to be 10^{-4}. Data augmentation is carried out with random horizontal flip; for skin images we additionally apply random 90-degree rotations. All images are resized to 256×256. The ratio between real and synthetic samples controlled by λ in Eq. 4.6 is set to 0.5 for all experiments, i.e., each mini-batch is composed by the same quantity of real and synthetic images. This also ensures that our method performs the same number of optimization steps as other approaches that do not use any synthetic data.

The node federation is trained for R rounds. In our implementation, at each round nodes are randomly ordered to establish each node's predecessor and successor: given our focus on medical applications, we can assume that the number of nodes is low enough that synchronization is not an issue. However, asynchronicity can be achieved by assuming that nodes can store incoming data in a queue: if the distribution of successor nodes is uniform and computation times are similar for all nodes, this is on average equivalent to the synchronous case. The number of rounds R and epochs E for FedER on the tuberculosis and melanoma classification tasks are set both to 100, according the 5-fold cross-validation results shown in Table 4.1. Buffer size is set for all experiments to 512.

GAN training

We recall that GAN training is carried out **before** federated learning using training data only, while leaving out test samples, as mentioned in Sect. 4.1.4. Our privacy-preserving GAN employs StyleGAN2-ADA [96] as a backbone, because of its suitability in low-data

Table 4.2: **Comparison between FedER and centralized baselines**. Results for FedER are obtained with a buffer size of 512, 100 rounds and 100 epochs per round.

Methods	**Tuberculosis**			**Melanoma**			
	Shenzhen	**Montgomery**	**Mean**	**BCN**	**HAM**	**MSK4**	**Mean**
Standalone	82.31	90.00	86.16	82.90	82.55	69.75	78.40
Centralized training	82.77	77.67	80.22	78.80	82.90	71.23	77.64
Centralized training with synthetic data only	76.92	79.33	78.13	60.71	61.09	61.23	61.01
Centralized training with synthetic data and real data	85.38	86.67	86.03	81.53	80.44	73.46	78.48
FedER (ours)	80.15	86.67	83.41	82.11	84.58	68.40	78.36

regimes and its generation capabilities. Training is carried out in two steps: 1) the GAN is initially trained without any privacy-preserving loss to support learning of high-quality visual features; 2) afterwards, we enable privacy-preserving loss and fine-tune the model in order to limit the embedding of patient-specific patterns in the GAN latent space. For classification purposes, GANs are trained in a label-conditioned fashion with a mini-batch size of 32 and learning rate of 0.0025 for both the generator and the discriminator. Early-stopping criteria are based on the Frêchet Inception Distance (FID) [76] between real and synthetic distributions: in the first training step, we stop training if FID does not improve for 10,000 iterations; in the second training step, we employ a criterion which stops training if FID increases by a factor of 2.5 w.r.t. the value obtained in the first step. As for the α parameter in Eq. 4.3, we tested multiple values of α (0, 0.5, 1, 1.5, 2 and 3) and found that the value of 1 yields the best compromise between image generation quality and pairwise LPIPS distance [231] over all tested datasets. In order to quantitatively evaluate privacy preservation, we also compute the average LPIPS distance between each real image and its closest synthetic sample by means of latent space projection (described in Sect. 4.1.4): the higher value of LPIPS, the lower the possibility to reconstruct real images from the generator.

Table 4.3: **Accuracy convergence among distributed node models**.Each local model is evaluated on all test sets of the federation in order to measure convergence and generalization (lower standard deviation corresponds to higher convergence).

	Dataset	**FedER**	**Standalone**
Tuberculosis	**Shenzhen**	80.54 ± 1.20	66.15 ± 22.84
	Montgomery	85.67 ± 2.36	70.00 ± 28.28
Melanoma	**BCN**	82.87 ± 1.22	65.06 ± 19.68
	HAM	84.45 ± 0.75	59.94 ± 20.47
	MSK4	67.78 ± 1.28	65.43 ± 5.05

Federated learning performance

We first evaluate the performance (in terms of classification accuracy) of FedER in the non-*i.i.d.* setting, and compare it to several centralized baselines, namely:

- **Centralized training**: all datasets are merged in a single node where all training happens. In this setting, no federated learning constraints are applied.
- **Centralized training with synthetic data only**. In this setting, each node trains a privacy-preserving GAN model and shares a synthetic version of its own data with the central node, where global training is performed. In this case, we aim to assess how much information is retained by synthetic data to support classification.
- **Centralized training with synthetic and real data.** This setting is a combination of the previous two: real and synthetic samples are centrally merged and used for training a global classifier. This scenario measures the contribution of synthetic data as a data augmentation approach.

We also compare FedER against standard training of the local node models, referred to as "Standalone" . Classification accuracy is computed using local node models on their own data. The results, reported in Table 4.2, show that standalone training appears to be the most favourable scenario. Centralized strategies perform generally worse than standalone training, because of the non-*i.i.d.* nature of the data. However, when the centralized approach is trained with original data augmented with synthetic samples, its classification accuracy is on par with the standalone training, possibly due to the learned generative latent spaces that likely tend to smooth different modes of non-*i.i.d.* data. FedER, instead, outperforms its centralized counterpart and yields slightly worse performance (1.5 percent points less) than standalone training. Although this may appear, at a first glance, as a shortcoming of FedER, we recall that in a federated learning scenario, we aim at building a model that, leveraging multiple data distributions present in the federation, may generalize better, thus addressing possible future data drifts. In order to assess the capabilities of the trained models to achieve such a generalization, we measure the decision convergence by evaluating how a local node model performs on other node datasets. Results are in Table 4.3 and show a good average accuracy, with a low standard deviation, by FedER, indicating that each node model performs equally well on its own dataset and on the others (i.e., all node models converge to similar decisions). Conversely, standalone training yields significantly lower accuracy and higher standard deviation than ours, demonstrating to be an unsuitable strategy for the sought generalization properties.

Table 4.4: **Comparison with state-of-the-art methods**. In bold, best accuracy values.

	Tuberculosis			Melanoma			
	Shenzhen	**Montgomery**	**Mean**	**BCN**	**HAM**	**MSK4**	**Mean**
FedAvg [134]	72.31	83.33	77.82	77.55	75.15	67.28	73.33
FedProx [116]	78.46	76.67	77.56	78.80	81.87	64.81	75.16
FedBN [118]	63.08	70.00	66.54	**82.19**	81.12	59.26	74.19
FedER (ours)	**80.15**	**86.67**	**83.41**	82.11	**84.58**	**68.40**	**78.36**

Thus, Table 4.2 shows the performance obtained by each node model on its internal test data, while Table 4.3 shows, instead, the performance obtained when each node model is tested again all other nodes' data. The latter results indicate that in FedER, any arbitrary node model can be used for the final evaluation, as all federation models converge to the same decisions. However, we further investigate whether building an ensemble of all node models yields better performance than using one arbitrary model. Results are given in Table 4.5 indeed showing higher accuracy by the ensemble. However, the models' ensemble leads to increased communication overhead (after training, all models have to be shared across the federation) and inference costs (each node needs to make a forward pass for all its available models to make the prediction). For this reason, the following experiments are carried out without using ensemble.

Table 4.5: **Accuracy performance with and without models' ensemble.** Results are computed by testing (first line) each node model with its own data and (second line) creating an ensemble and testing it on all nodes' data.

Method	**Tuberculosis**	**Melanoma**
No ensemble	83.41 ± 4.61	78.36 ± 8.72
Ensemble	$\mathbf{84.77 \pm 4.57}$	$\mathbf{80.35 \pm 9.42}$

We then compare our approach (without ensemble) to state-of-the-art federated learning approaches, namely: a) server-based federated methods, FedAvg [134] and FedProx [116], which have shown to perform generally better than decentralized methods [108, 195], and b) a personalized method, FedBN [118]. As already mentioned, to avoid biased assessment, we use the official code repository[2] of FedBN [118] and hyper-parameter selection on the tested datasets was carried out through grid search on training rounds/epochs, learning rate and μ for FedProx [116] using 5-fold cross validation as for our approach. Results, for the tuberculosis and the melanoma tasks, are reported in Table 4.4 and show that FedER outperforms

[2] https://github.com/med-air/FedBN

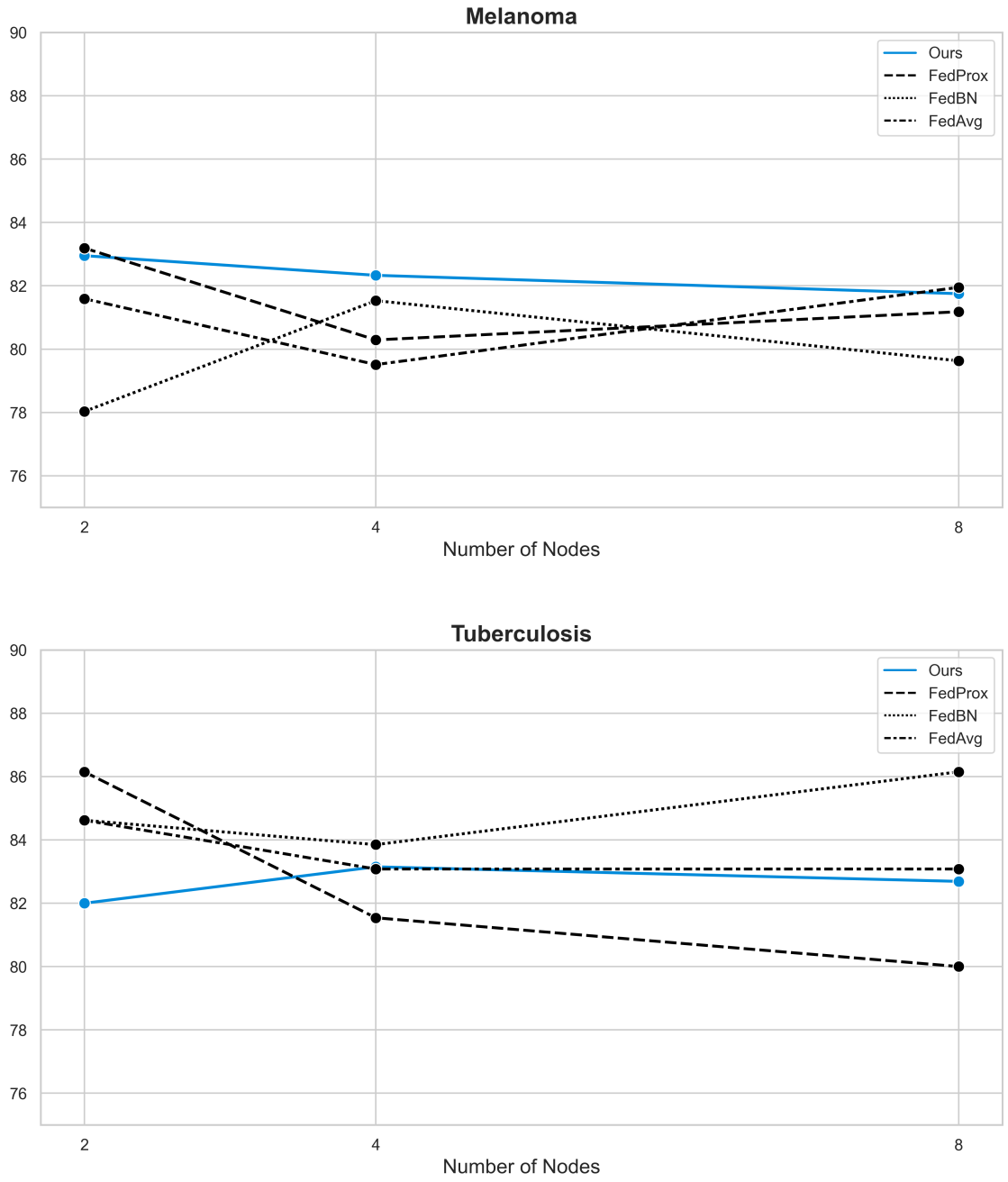

Figure 4.2: **Scalability performance in the *i.i.d.* setting** w.r.t. number of nodes for the proposed approach and state-of-the-art methods.

all methods under comparison. Interestingly, FedER learning strategy does better than: a) *server-based methods*, FedAvg [134] and FedProx [116], suggesting that experience replay is a more effective feature aggregation approach than naive parameter averaging; b) personalized methods, such as FedBN [118], which affects a limited aspect of feature representation (i.e., input layer distributions), while our approach adapts the entire model to local and remote

Table 4.6: **FedER classification accuracy w.r.t. buffer size**. Each local model is evaluated on all test sets of the federation in order to measure convergence and generalization (lower standard deviation corresponds to higher convergence).

	Node Convergence	
Buffer	**Shenzhen**	**Montgomery**
0	70.62 ± 11.97	80.33 ± 10.84
256	80.46 ± 2.96	81.67 ± 4.24
512	80.54 ± 1.20	85.67 ± 2.36
1024	82.23 ± 1.31	86.00 ± 3.01
2048	82.08 ± 1.39	88.67 ± 2.97

tasks.

These above results suggest that experience replay plays a key role in federated models as a principled way to integrate features coming from different data distributions. To further assess its contribution, we evaluate FedER performance when using buffer at different sizes. Results on the tuberculosis task, measured as mean and standard deviation of the local node models over a given dataset, are shown in Table 4.6 and indicate a clear contribution of the buffer in terms of overall performance and models' agreement. Indeed, with no buffer we obtain the lowest average performance and the highest standard deviation. As the buffer is enabled, we can observe a performance gain (mainly for the Shenzhen dataset) and a significant drop in standard deviation. Performance improves as buffer size increases, although gain becomes negligible above 512. Since higher buffer sizes result in more data to be shared among nodes, we use a buffer size of 512, as the best trade-off between accuracy and communication costs.

We finally evaluate the capability of FedER to scale with the size of the federated network. Accordingly, we quantify this property using an *i.i.d.* setting on both tuberculosis (Shenzhen dataset) and skin lesion classification (BCN dataset) tasks, by equally splitting the available data on multiple nodes. Fig 4.2 shows how the proposed approach is able to keep classification accuracy high and performs on par with state-of-the-art approaches (namely, FedAvg, FedProx and FedBN).

Privacy-preserving performance

In this section we quantify how much information of real samples is retained by our privacy-preserving method, and in particular in the mapping between latent space and synthetic images. To do so, we employ the projection method proposed in [95]: given a real image $\mathbf{x}$, we find an intermediate latent point $\mathbf{w}$ such that the generated image $G(\mathbf{w})$ is most similar

to $\mathbf{x}$, by optimizing $\mathbf{w}$ to minimize the LPIPS distance [231] between $\mathbf{x}$ and $G(\mathbf{w})$.

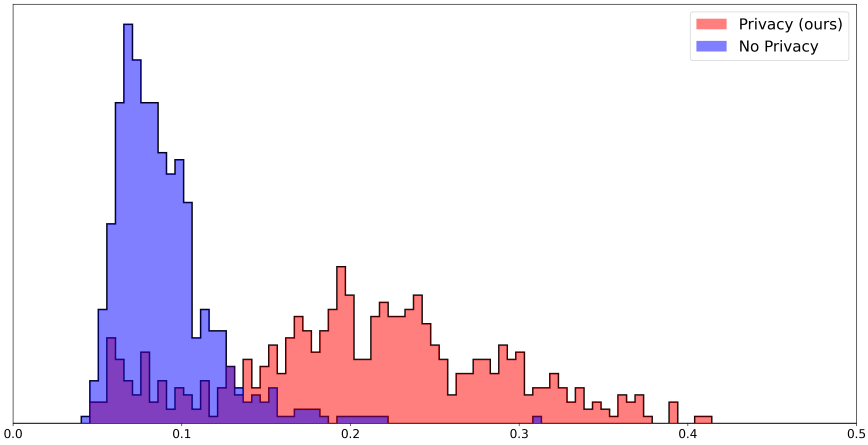

Figure 4.3: **Quantitative analysis of privacy-preserving generation**. In blue, LPIPS distance histogram between real images and the corresponding images obtained through latent space projection using a GAN trained without the proposed privacy-preserving loss. In red, LPIPS distance histogram between real images and the closest images generated with the proposed approach.

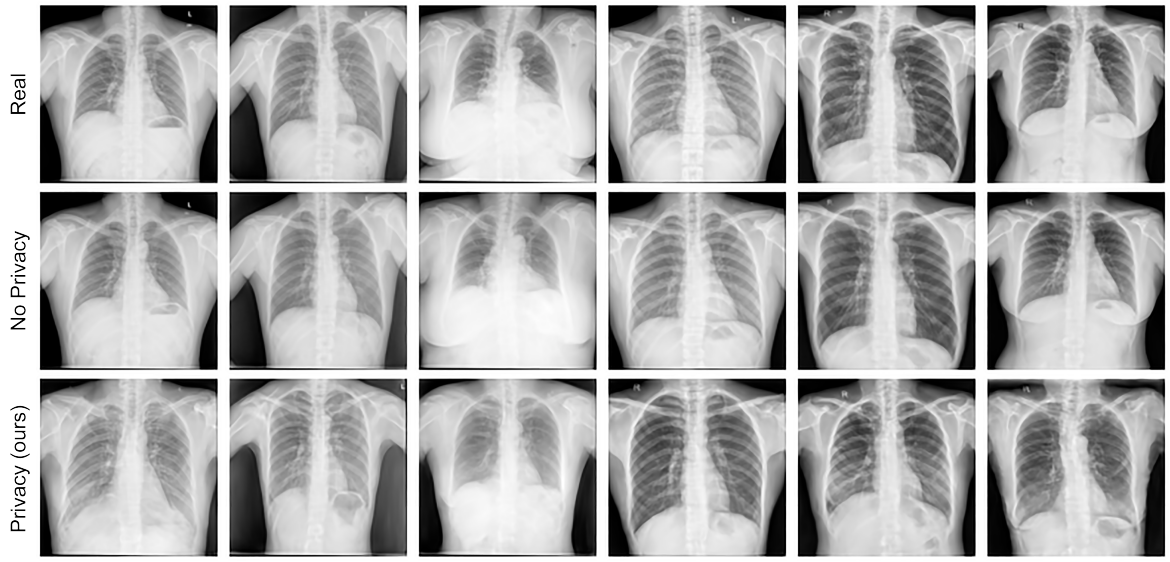

Figure 4.4: **Qualitative samples of our privacy-preserving generation.** Top row: real images from the Shenzhen dataset. Middle row: projection with a standard GAN. Bottom row: projection with our privacy-preserving GAN.

In practice, for each image of the dataset used for GAN training, we perform backprojection to find its most similar synthetic sample, and measure the LPIPS distance between the original and projected images. Fig. 4.3 shows the histograms of the resulting distances on the Shenzhen dataset, using GAN models trained with and without the proposed privacy-

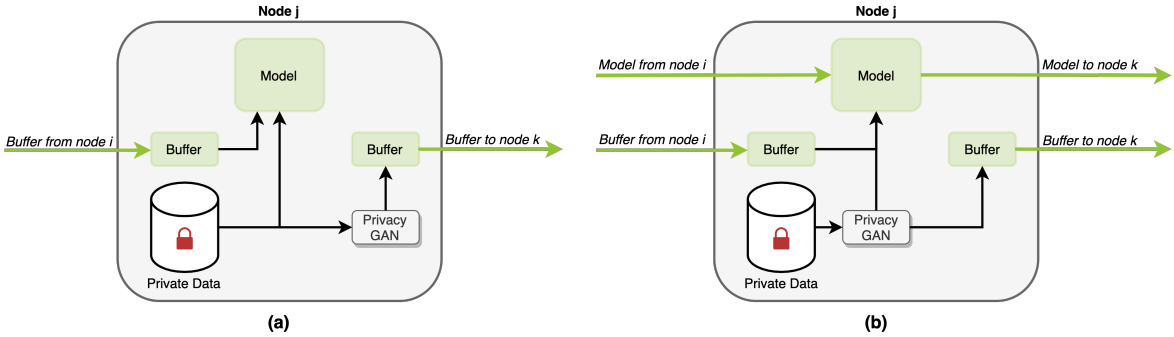

Figure 4.5: **Privacy-enhanced alternative architectures.** *(a) FedER-A* configuration ("Buffer-only sharing"): a local node model is trained on real data, but only a buffer of synthetic samples is shared with other nodes. *(b) FedER-B* ("Synthetic-only training"): Even within the dataset owner node, models are trained on synthetic data only.

preserving loss (both models start from the same $\mathbf{w}$, for fairness). The histograms show that standard GAN training, with no privacy-preserving loss, tends to yield distances closer to 0, demonstrating that real images are indeed included into the generator latent space; while our model significantly mitigates this issue, by synthesizing samples that are substantially different than the original ones. In order to qualitatively substantiate these findings, Fig. 4.4 compares original samples from the Shenzhen dataset with the corresponding projections, generated with and without our privacy-preserving loss[3]. It is easy to notice that generated samples with a traditional GAN highly resemble real data, making it impossible to share such samples, albeit synthetic, in a privacy-safe manner, as they clearly contain patient information. Instead, comparing real images with the projections obtained from privacy-preserving GAN confirms the inability of the generator to find latent representations that recover real images used during training.

Given the high realism of generated samples, we run additional tests by proposing two FedER variants aiming to increase the level of privacy preservation: a) *FedER-A*: models are not shared among nodes — only synthetic buffers are sent and received; b) *FedER-B*: models are trained only using synthetic data, even on local nodes. Fig. 4.5 shows the internal architecture of each node in the two variants. Results obtained with these alternative privacy-enhanced configurations are provided in Table 4.7. It can be noted that FedER-A (i.e., "buffer-only sharing") configuration achieves comparable performance to our standard FedER (82.76 vs 83.41), but, remarkably, it outperforms all existing federated learning methods on the same datasets (compare Table 4.4 with the node performance block in Table 4.7). The FedER-B (i.e., "synthetic-only training") configuration, instead, performs slightly worse

[3]We show only X-Ray synthesized samples, as the effect is of our privacy-preserving strategy, is more appreciable than in skin lesion data.

than the other two configurations, but is still on par with existing federated methods.

Table 4.7: **Classification accuracy of the proposed privacy-enhanced strategies in the *non-i.i.d.* setting**. FedER-A: only buffers are shared (Fig. 4.5-a). FedER-B: models are trained on synthetic data only (Fig. 4.5-b). Node performance measures how each node model performs on its own private dataset, while node convergence assesses how a node model performs on other federation nodes.

Config	Node Performance			Node Convergence	
	Shenzhen	Montgomery	Mean	Shenzhen	Montgomery
FedER	80.15	86.67	83.41	80.54 ±1.20	85.67 ±2.36
FedER-A	83.54	82.00	82.76	78.84 ±6.64	81.00 ±3.30
FedER-B	74.15	81.33	77.74	73.61 ±4.68	80.40 ±3.60

Communication and computational performance

We conclude the experimental analysis by measuring *communication* and *computational* costs. As for *communication costs*, compared to state-of-the-art approaches, FedER requires additional transmission of synthetic images between nodes at each round. Tab. 4.8 reports per-node communication costs for state-of-the-art models (the table reports FedAvg, but the same values apply for FedProx and FedBN) and for FedER, in its full formulation and in the FedER-A variant, where only buffers of synthetic data are shared. The main cost for state-of-the-art models lies in the transfer of the model, and depends on the specific architecture (we included ResNet-18 and ResNet-152 as representative examples of different model scales). Values for our approach are reported for buffers of size 512 containing 256×256 images, and depend on the color space. For our full FedER model, the increment in communication costs is significant but not excessive. However, if we take into consideration the variant where only synthetic data are exchanged (i.e., FedER-A), which still performs better than state-of-the-art methods (Tab. 4.4 and Tab. 4.7), communication overhead becomes significantly less than model-sharing approaches.

As for *computational costs* of federated training, FedER incurs the same overhead for parameter optimization and aggregation as state-of-the-art methods. Additionally, before federated training starts, FedER requires that each node trains a local privacy-preserving GAN off-line; this, however, does not affect online federated learning costs, as it is carried out only once at the very beginning of the whole procedure.

Furthermore, we argue that, in the medical domain, the number of institutions in a federation is relatively low and it is reasonable to assume that nodes can benefit from a powerful communication network and computing infrastructure: thus, the overhead introduced by

FedER is tolerable, in light of the methodological advantages and the obtained performance and generalization capabilities showed by the resulting models.

Table 4.8: **Communication results comparison**

	Tuberculosis		**Melanoma**	
	ResNet-18	**ResNet-152**	**ResNet-18**	**ResNet-152**
FedAvg FedProx FedBN	45 MB	230 MB	45 MB	230 MB
FedER	65 MB	250 MB	105 MB	290 MB
FedER-A	20 MB	20 MB	60 MB	60 MB

4.1.5 Discussion

In this section we presented FedER, a decentralized federated learning framework that replaces traditional parameters averaging with a more principled feature integration approach based on the combination of experience replay, investigated in the second chapter of this thesis, and privacy-preserving generative models, investigated in the previous chapter. In FedER, nodes communicate with each other by sharing local models and buffers of synthetic samples; local model updates are carried out in a way that encourages the reuse and adaptation of features learned by other nodes, thus avoiding potentially disruptive effects due to blind feature averaging. Experimental results show that our method outperforms significantly state-of-the-art server-based approaches in a non-*i.i.d.* scenario, which is a typical setting in the medical domain. Additionally, quantitative and qualitative analysis shows that our privacy-preserving generation approach is able to synthesize samples that are significantly different from real data, while correctly supporting the learning of discriminative features. In the future, we aim at investigating some unexplored properties of our method: for instance, unlike all other existing methods based on parameter averaging is required, our approach does not strictly require that all nodes share the same model architecture. Model heterogeneity could therefore be employed to create a shared ensemble and combine different feature learning capabilities. This section demonstrated that applying Continual Learning (CL) techniques to Federated Learning (FL), with the support of generative models, can lead to significant improvements. It also paves the way for addressing the more challenging scenario of Federated Continual Learning, which will be explored in the next section.

4.1.6 Publications

Pennisi, M., Salanitri, F. P., Bellitto, G., Casella, B., Aldinucci, M., Palazzo, S., & Spampinato, C. (2024). FedER: Federated Learning through Experience Replay and privacy-preserving data synthesis. Computer Vision and Image Understanding, 238, 103882. [156]

4.2 FedRewind: Rewinding Continual Model Exchange for Decentralized Federated Learning

4.2.1 Motivation

The proliferation of data across multiple distributed devices and locations has sparked significant interest in federated learning (FL), a paradigm that enables collaborative model training without the need to centralize data. Federated learning offers numerous benefits, including enhanced privacy and reduced communication costs. However, a fundamental challenge in FL is the non-i.i.d. (independent and identically distributed) nature of data across different nodes, which can lead to performance degradation due to data distribution shifts. This problem becomes even more pronounced in decentralized federated learning, where there is no central server to coordinate and aggregate updates, making the system less robust to heterogeneous data distributions.

Existing solutions in federated learning primarily focus on mitigating the effects of non-i.i.d. distributions through various aggregation and optimization techniques. Centralized federated learning approaches often rely on a central server to aggregate updates from all nodes, thereby smoothing out the differences in local data distributions [116,134,187]. Decentralized methods, instead, employ peer-to-peer communication and model averaging strategies to achieve consensus without a central entity [14, 29, 213]. While these methods have shown promise, they often fall short in addressing the dynamic nature of data distribution shifts, particularly in environments featured by strong data imbalance [218, 233].

Continual learning (CL) [45,133,151], a field that addresses the problem of learning from a stream of data that changes over time, offers valuable insights for handling distribution shifts with strong imbalance. CL methods are designed to prevent *catastrophic forgetting*, which occurs when a model forgets previously learned information upon encountering new data, by maintaining knowledge across sequential learning tasks through either exposing the model to limited past experience [11, 23, 164, 169] or regularizing model parameters [19, 102, 229] using previous knowledge, while learning new tasks. Although CL and FL address similar challenges, they operate in different contexts: CL deals with non-i.i.d. data over time, while FL addresses non-i.i.d. data across distributed nodes.

We propose *FedRewind*, a novel approach that integrates continual learning concepts into federated learning to address the limitations of existing FL methods. Our method involves decentralized nodes periodically exchanging their models and sending them back to the originating nodes for a limited number of iterations during local training. This exchange mechanism, inspired by continual learning strategies, aims to prevent overfitting on local

data and enhance memory retention by periodically re-exposing models to previously seen data.

FedRewind's strategy also aligns with the cognitive neuroscience principle of *testing effect*, which emphasizes the role of active recall and retrieval practice for the enhancement of long-term memory. The testing effect, in particular, demonstrates that memory retrieval processes (similar to our rewind strategy) significantly improve knowledge retention compared to simple re-exposure to information [93, 171]. This phenomenon is underpinned by mechanisms such as elaborative retrieval and spreading activation, where active recall strengthens memory traces and facilitates the integration of new information into existing cognitive frameworks.

By adapting cognitive neuroscience principles and continual learning concepts to the spatial distribution challenges of FL, *FedRewind* aims to reduce the distribution shift between nodes, thus enhancing model performance and robustness. We validate our claims on multiple benchmark datasets, demonstrating how *FedRewind* leads to performance improvement over standard decentralized federated learning methods, as well as those that impose specific routing schemes within the federation. Furthermore, the combination of federated and continual learning concepts enables our method to effectively address the federated continual learning problem, where data shifts occur over both space and time, outperforming existing baselines. Our results, cumulatively, indicate that this decentralized and iterative model exchange approach offers a robust solution to the challenges posed by non-i.i.d. data in federated learning environments.

4.2.2 Related Work

Federated learning (FL) has emerged as a new paradigm within distributed machine learning, addressing the challenge of data privacy. Drawing upon the foundational work of McMahan et al. [134], FL facilitates collaborative model training while ensuring node data remains secure on their local devices.

A typical FL setting features a central server that orchestrates the learning process. This server distributes a global model to a pool of participating nodes, which use their private data for local updates on the received model. Subsequently, the nodes transmit their local updates back to the central server, which aggregates them to refine the global model. This iterative process of distributing, updating, and aggregating the model persists until a satisfactory level of convergence is achieved. The most common aggregation technique is FedAvg [134], that simply averages the local model parameters received from all nodes. More sophisticated aggregation methods have been proposed by adding a regularization term

[116] or leveraging knowledge distillation [236]. Another branch of FL, namely Personalized Federated Learning, has the primary objective to improve the performances w.r.t. only the single node distribution. FedBN [118] achieves this goal by preserving the batch-norm layers of each node while FedProto [197] aggregates only the prototypes while the models are kept on each node.

While a central server simplifies the communication protocols, especially for large-scale deployments, its presence introduces specific limitations. Firstly, it creates a single point of failure, posing a vulnerability to system robustness if the server becomes unavailable. Secondly, as the number of participating nodes increases, the central server can become a bottleneck, hindering communication efficiency [120]. Finally, the very presence of a central server that aggregates data might not be desirable or even feasible in certain collaborative learning scenarios. This is particularly true for scenarios that prioritize robust privacy guarantees or involve geographically dispersed participants with limited or unreliable network connectivity.

This work investigates also decentralized federated learning, which, conversely to centralized approaches, relies on peer-to-peer communication between nodes [29,92]. This approach eliminates the single point of failure and enhances privacy guarantees, but introduces additional complexity in terms of communication protocols and achieving convergence among local models on all devices.

Since *FedRewind* leverages concepts from **continual learning (CL)**, we provide a brief overview of existing methods related to the strategies we employ to retain knowledge from past learning rounds. Continual learning [45, 151] is a field of machine learning that seeks to bridge the gap between the incremental learning observed in humans and the limitations of neural networks. McCloskey and Cohen [133] identified the phenomenon of "catastrophic forgetting", where neural networks lose previously acquired knowledge upon encountering substantial shifts in the input distribution.

To address catastrophic forgetting, various mitigation strategies have been proposed. These include the introduction of appropriate regularization terms [102, 229], the development of specialized network architectures [130, 180], and the use of rehearsal mechanisms that leverage a limited set of previously encountered data points [23, 164, 169].

FedRewind adopts a hybrid approach, combining elements of both regularization and rehearsal strategies. Unlike traditional methods, it does not use any buffer. Instead, during training rounds on local nodes, the model is periodically sent back to previous nodes for regularization. This process helps address data shifts across nodes, thereby mitigating potential forgetting.

Federated continual learning (FCL) combines the paradigms of federated learning (FL) and continual learning (CL), enabling it to address the challenge of distributed data that, at the same time, undergo continual change over time [54, 219, 222]. However, initial efforts in this area compromised data privacy by requiring the storage of training samples on the central server [219]. Recent advancements in FCL prioritize data privacy by advocating for the storage of only perturbed images for replay purposes [54]. FedWEIT [222], instead, tackled the problem by decomposing network weights but necessitates data replay buffers. Recently, FedSpace [185] has been developed to overcome the limitations of current methods. It leverages class prototypes within the feature space and employs contrastive learning to preserve prior knowledge and reduce divergence between the behaviors of different federation nodes.

Our *FedRewind* strategy is complementary to approaches like FedSpace, enhancing their capabilities by mitigating typical overfitting in distributed learning, particularly in cases of high class imbalance and strongly non-iid data distribution among nodes. Specifically, *FedRewind* addresses these issues by transferring models between nodes to enforce iid-ness on data, rather than relying on data storage. This method significantly reduces privacy concerns associated with storing sensitive training data. By periodically sending the model back to previous nodes, we maintain knowledge across sequential tasks and enforce regularization, thus reducing node overfitting, while enhancing both privacy (no need for data sharing) and scalability.

4.2.3 Method

In federated learning, a collection of nodes collaboratively train a shared model while keeping their data localized. This approach ensures data privacy but introduces challenges related to effective knowledge sharing across distributed and sequential learning tasks. To address these challenges, we propose a novel "rewind" strategy. This section introduces the approach and key concepts, describes the rewind method in detail, and provides the corresponding pseudo-code.

Federated Learning. Federated learning is a collaborative machine learning approach where multiple nodes (N nodes) train a shared model without centralizing their data. Each node updates its model using local data and shares the model updates rather than the data itself, ensuring data privacy.

Decentralized Federated Learning. In this framework, nodes communicate directly with each other without a central server. We consider two modes of communication:

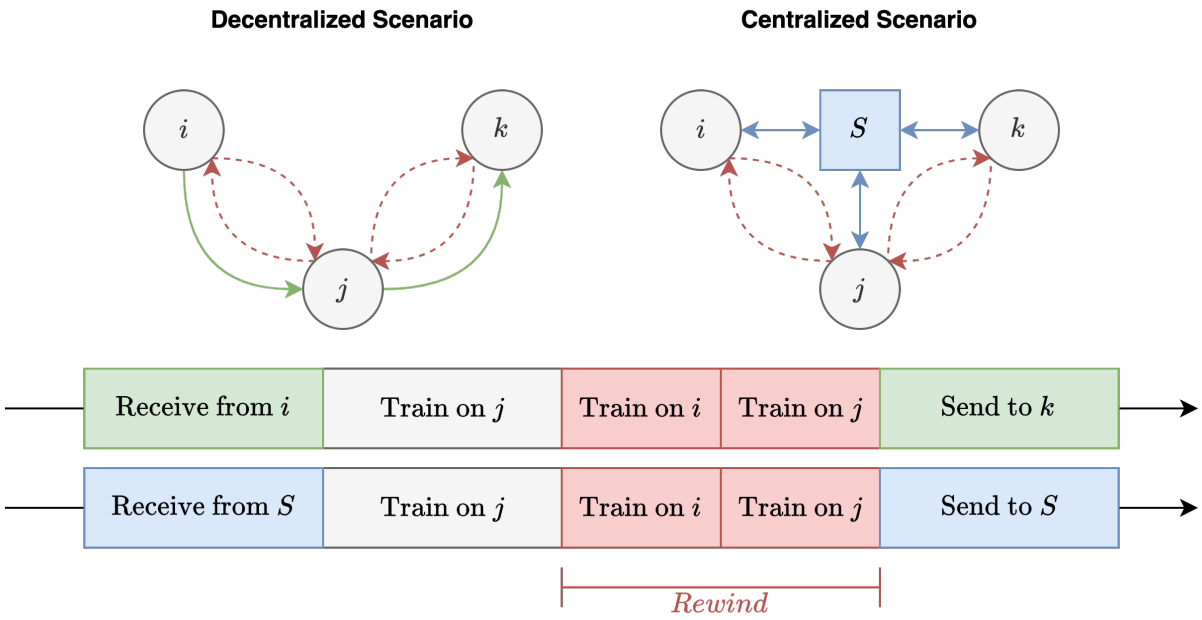

Figure 4.6: **Rewind strategy.** The model received and trained on the current node is sent back to its source node for a brief fine-tuning. The model then returns to the node and continue its training before the start of a new federated round.

- **Random Communication (RWT)**: Nodes select a random source node (for the incoming model) and a random destination node (for the outcoming model) for information exchange in each round.
- **Cyclic Communication (CWT)**: Each node communicates with the same predetermined source and destination nodes in every round, as described in [29].

Centralized Federated Learning. In this framework, a central server coordinates the training process. Nodes send their local model updates to the server, which aggregates them to form a global model.

Training Rounds. Defined as a block of training where all nodes complete training for E epochs. At the end of each round, model exchange across all federated nodes is carried out.

The Rewind Strategy

To improve knowledge sharing in federated learning, we introduce the "rewind" strategy, described in Fig. 4.6. This method involves temporarily reverting the model to a previous node to rehearse prior knowledge, thus preserving data privacy. We apply this strategy on both centralized and decentralized federated learning.

Decentralized federated learning. During each communication round, a generic node C_j receives a model M_i, parameterized by θ_i, from a source node C_i, trains it on its local dataset D_j, and forwards it to another node C_k. The standard training process on node C_j for model M_j parameterized by θ_j at round t on dataset D_j is given by:

$$M_j^{(t)} = \text{Train}_{D_j,E}(M_i^{(t-1)}) = \theta_i^{(t-1)} - \eta \sum_{0}^{E-1} \nabla L(\theta_i^{(t-1)}, D_j) \tag{4.7}$$

where E denotes the number of epochs for a single federation round, and L is a generic loss function.

To enhance knowledge retention, we introduce a fractional computation budget parameterized by λ for retraining the model on its origin node before continuing training on the current node. This modifies the training equation as follows:

$$M_j^{(t)} = \text{Train}_{D_j,\lambda\cdot E}\left(\text{Train}_{D_i,\lambda\cdot E}\left(\text{Train}_{D_j,(1-2\lambda)\cdot E}\left(M_i^{(t-1)}\right)\right)\right) \tag{4.8}$$

where λ denote the fraction of the budget allocated for rewinding.

Centralized Federated Learning. In this scenario, a central server S aggregates models received from nodes at each communication round. The model computed by the server $M_s^{(t)}$ at round t is defined as:

$$M_s^{(t)} = agg(\{M_j^{(t)} \mid j \in \{1, 2, \ldots, N\}\}) \tag{4.9}$$

where $agg(\cdot)$ represents a generic aggregation function employed by the server. Applying the rewind strategy, the training process for a generic node C_j is modified to:

$$M_j^{(t)} = \text{Train}_{D_j,\lambda\cdot E}\left(\text{Train}_{D_i,\lambda\cdot E}\left(\text{Train}_{D_j,(1-2\lambda)\cdot E}\left(M_s^{(t-1)}\right)\right)\right) \tag{4.10}$$

By leveraging inter-node communication and the rewind strategy, federated learning—whether decentralized or centralized—can more effectively retain knowledge across different tasks. This approach ensures that the shared model benefits from the distributed data while maintaining privacy and improving performance.

The pseudo-code of the rewind strategy is reported in Alg. 2 and Alg. 3 respectively for the decentralized and centralized scenario.

Algorithm 2: Decentralized Federated Learning with Rewind Strategy.

Input: N nodes, initial model M_0, epochs E, fractional budget λ

for *each round* t *in* 1 *to* T **do**

  for *each node* C_j *in* N **do**

    // Receive model from source node C_i

    $M_i^{(t-1)} \leftarrow \text{receive\_model}(C_i)$;

    // Train on current node's dataset D_j

    $M_{\text{intermediate}} \leftarrow \text{Train}(M_{\text{rewind}}, D_j, (1-2\lambda)\cdot E)$;

    // Rewind phase: Train on source node's dataset D_i

    $M_{\text{rewind}} \leftarrow \text{Train}(M_i^{(t-1)}, D_i, \lambda \cdot E)$;

    // Finish training on current node's dataset D_j

    $M_j^{(t)} \leftarrow \text{Train}(M_{\text{intermediate}}, D_j, \lambda \cdot E)$;

    // Send model to destination node C_k

    $\text{send\_model}(C_j, M_j^{(t)})$;

  end

end

Algorithm 3: Centralized Federated Learning with Rewind Strategy.

Input: N nodes, initial model M_0, epochs E, fractional budget λ

for *each round* t *in* 1 *to* T **do**

  for *each node* C_j *in* N **do**

    // Receive aggregated model from the server

    $M_s^{(t-1)} \leftarrow \text{receive\_model}(\text{server})$;

    // Train on current node's dataset D_j

    $M_{\text{intermediate}} \leftarrow \text{Train}(M_{\text{rewind}}, D_j, (1-2\lambda)\cdot E)$;

    // Rewind phase: Train on previous node's dataset D_i

    $M_{\text{rewind}} \leftarrow \text{Train}(M_s^{(t-1)}, D_i, \lambda \cdot E)$;

    // Finish training on current node's dataset D_j

    $M_j^{(t)} \leftarrow \text{Train}(M_{\text{intermediate}}, D_j, \lambda \cdot E)$;

    // Send model to the server

    $\text{send\_model}(\text{server}, M_j^{(t)})$;

  end

  // Server aggregates models from all nodes

  $M_s^{(t)} \leftarrow \text{aggregate\_models}(\{M_j^{(t)} \mid j \in 1 \text{ to } N\})$;

end

4.2.4 Experimental Results

Federated Learning Performance

Experimental settings. To evaluate the effectiveness of *FedRewind*, we simulate different federated learning scenarios using three benchmarks (generally employed for testing FL methods), namely MNIST [49], CIFAR10 [106] and CIFAR100 [106]. Data is distributed across nodes according to a non-independent and identically distributed (non-IID) scheme. This distribution is achieved by applying a Dirichlet distribution, as in previous work [115, 206, 226], parameterized by α_{dir}, which serves as a measure of the degree of data heterogeneity across nodes; a lower α_{dir} value indicates a more pronounced imbalance in data distribution across nodes. Our experimental settings include 50 communication rounds and two configurations based on the number of nodes in the federation: one with 10 nodes and another with 50 nodes. During each round, in each node, we perform local training for 10 epochs (E). For the rewind experiments, we set the rewind hyperparameter $\lambda = 0.1$. This configuration determines a training procedure where, within the given E=10 epochs, 8 epochs are dedicated to the local training on the current node's data, followed by one epoch on the previous node's data, and concluding with a final epoch on the current node's data. All experiments are carried out using the ResNet18 architecture [73], pre-trained on ImageNet [48], optimized using Stochastic Gradient Descent (SGD) with a learning rate of 0.001.

Metrics. In decentralized federated learning (FL), each node creates a distinct local model, unlike in the centralized FL paradigm, which results in a single global model at the end of the training phase. To quantify the aggregated performance and generalization capability of the federation, we propose the *Federation Accuracy (FA)* metric. This metric is calculated by testing all the node models within the federation against all the private test sets and then computing the mean accuracy. Given a federation of size N, our metric is defined as follows:

$$\text{FA} := \frac{1}{2 \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} \text{Acc}(M_i, D_j^{test}) \tag{4.11}$$

where $\text{Acc}(M_i, S_j)$ is the accuracy of model M_i on the private test set D_j^{test} of node j.

Moreover, our objective is not only to improve the overall generalization across the federation, but also to enhance performance of each individual node on its private dataset. To measure this performance, we define the *Personalized Federation Accuracy (PFA)* metric as:

$$\text{PFA} := \frac{1}{N} \sum_{i=1}^{N} \text{Acc}(M_i, D_i^{test}). \tag{4.12}$$

These two metrics, PA and PFA, allow us to capture both the generalization capabilities of the entire federation and the performance improvements from the perspective of individual nodes.

Baselines. We test our approach in combination to existing FL strategies, applying it to CWT [29], RWT, and FedAvg [134]. CWT employs a static cyclic model transfer between rounds, while RWT is our modified version of CWT, featuring random communication between nodes in each round.

We also assess our approach in two reference scenarios: the *Joint* and *Standalone* settings. The *Joint* setting represents an optimal condition where all data from the federation is consolidated and utilized for training on a single node, thereby establishing an upper bound on performance.

In contrast, the *Standalone* setting assumes that each node trains its model independently, with no communication or data sharing between nodes. This setting generally sets a lower bound on performance, particularly when the data distribution between nodes is highly non-IID.

Results. We begin our evaluation by testing FedRewind performance on the two scenarios: with 10 nodes and with 50 nodes, both under a strongly non-IID scenario with $\alpha_{dir} = 0.25$. Table 4.9 presents the results in terms of Federation Accuracy (FA) across the three benchmarks, showing that the *rewind* strategy consistently achieves higher accuracy. This improvement is especially pronounced in the 50-node scenario, which poses greater complexity and challenge due to its larger and more heterogeneous node distribution.

The Personalized Federation Accuracy (PFA) results, shown in Table 4.10, further demonstrate the benefits of our rewind strategy at the node level. The strategy's effectiveness is evident, as it consistently enhances PFA across various datasets and federation scales. We speculate that the rewind mechanism acts as an effective regularizer, mitigating overfitting to a node's local dataset. In a non-IID setting, where the tendency to overfit to local data patterns is high, this regularization effect is crucial. By periodically rewinding and retraining with data from other nodes, the models are exposed to a more diverse data distribution, promoting more generalized representation learning. It is also noteworthy that the impact of our rewind strategy on personalization performance for FedAvg is relatively lower compared to CWT and RWT. This might be due to the aggregation step in FedAvg, which tends to smooth out the specificities of local models trained on non-IID data.

The results from the standalone setting highlight the limitations of training models in isolation, especially under non-IID conditions. As the number of nodes increases, standalone

models generally perform poorly, struggling to learn representative features without the diversity of data from other nodes. This aligns with our expectations, as the standalone setting misses the collaborative benefits of federated learning.

Method	**10 Nodes**			**50 Nodes**		
	MNIST	**C10**	**C100**	**MNIST**	**C10**	**C100**
Joint	99.22	78.53	53.13	99.22	78.53	53.13
Standalone	69.19	33.23	19.07	49.91	26.85	11.08
FedAVG	95.43	51.19	39.89	87.47	53.26	37.29
↪**Rewind**	**97.59**	**59.27**	**41.06**	**91.77**	**58.84**	**39.90**
CWT	93.09	45.81	34.02	88.27	40.00	24.90
↪**Rewind**	**96.19**	**55.57**	**37.44**	**92.79**	**47.83**	**27.51**
RWT	94.93	44.16	32.08	86.66	38.75	24.42
↪**Rewind**	**97.42**	**52.34**	**36.72**	**87.01**	**45.85**	**27.40**

Table 4.9: **Federation Accuracy** in a non-IID setting ($\alpha_{dir} = 0.25$) for the MNIST, CIFAR-10, and CIFAR-100 benchmarks, organized across 10 and 50 nodes.

Method	**10 Nodes**			**50 Nodes**		
	MNIST	**C10**	**C100**	**MNIST**	**C10**	**C100**
Joint	99.22	78.53	53.13	99.22	78.53	53.13
Standalone	98.69	83.96	53.49	96.20	77.65	41.73
FedAVG	97.09	**56.15**	37.94	83.83	45.88	32.54
↪**Rewind**	**97.38**	54.39	**38.71**	**85.79**	**47.99**	**35.62**
CWT	94.82	38.01	30.76	86.74	38.90	24.72
↪**Rewind**	**95.65**	**48.85**	**34.34**	**93.40**	**46.29**	**26.58**
RWT	92.95	43.48	29.36	85.43	33.64	23.90
↪**Rewind**	**96.61**	**45.95**	**35.50**	**85.79**	**47.62**	**27.59**

Table 4.10: **Personalized Federation Accuracy** in a non-IID setting ($\alpha_{dir} = 0.25$) for the MNIST, CIFAR-10, and CIFAR-100 benchmarks, organized across 10 and 50 nodes.

We further evaluated the robustness of our rewind strategy under varying degrees of data heterogeneity. Using the CIFAR-10 dataset, we measured federation accuracy with the Dirichlet coefficient (α_{dir}) ranging from 0.1, indicating extreme non-IID conditions, to 0.5, representing a less skewed data distribution among nodes. The results, shown in Fig. 4.7, demonstrate that our rewind strategy consistently enhances FA, with the highest gain obtained at $\alpha_{dir} = 0.1$, the most challenging setting. This suggests that the rewind strategy is particularly effective in environments with high data distribution skewness. As α_{dir} increases, *FedRewind* continues to provide benefits, though they are less pronounced. These findings collectively demonstrate that the rewind strategy is a robust method for federated

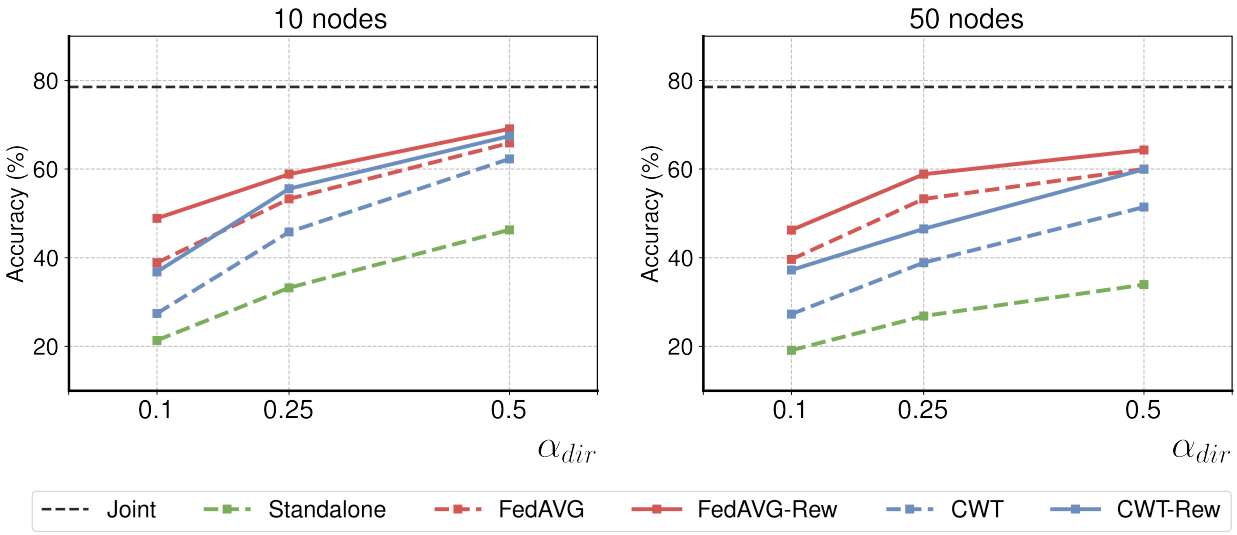

Figure 4.7: **Performance at different degrees of data heterogeneity** (α_{dir}) on CIFAR-10 for 10 (left) and 50 (right) nodes.

learning, capable of enhancing model performance in diverse data conditions. Its consistent performance across different levels of non-IIDness, as shown in Fig. 4.7, suggests reliable applicability in real-world federated settings where data distributions vary widely.

We finally verify whether the enhanced performance is due to rewinding to the previous node or to any other node. Thus, we compared our rewind strategy to a random rewinding one, where the model is sent to a random node of the federation. Table 4.11 shows the performance of the two strategies when combined to CWT and RWT, highlighting how rewinding to the previous node in the communication chain is more effective than using a random node. It has to be noted that, in RWT, the performance increase is slightly lower because the preceding node changes at each communication round, slightly reducing the benefits of rewinding.

Method	**CIFAR-10**		**CIFAR-100**	
	10 Nodes	**50 Nodes**	**10 Nodes**	**50 Nodes**
CWT	45.81	40.00	34.02	24.90
↪**RandRewind**	51.59	45.76	36.58	**27.60**
↪**Rewind**	**55.57**	**47.83**	**37.44**	27.51
RWT	44.16	38.75	32.08	24.42
↪**RandRewind**	50.83	45.62	36.62	**27.40**
↪**Rewind**	**52.83**	**45.85**	**36.72**	27.09

Table 4.11: **Comparison between different rewinding strategies**. RandRewind sends the model back to a random done instead of the previous node as Rewind does.

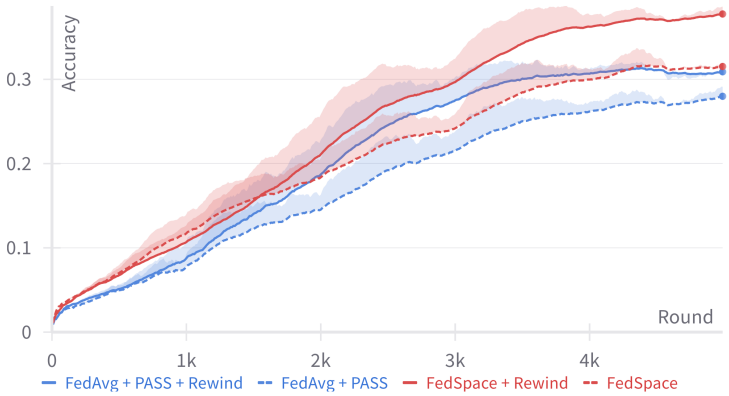

Figure 4.8: **Training trend of rewind strategy in AFCL**

Method	**Accuracy**
FedAvg + PASS	29.70
↪**Rewind**	**33.46**
FedSpace	35.53
↪**Rewind**	**39.40**

Table 4.12: **Results of rewind strategy in AFCL**

Continual Federated Learning

We also evaluate *FedRewind* within the complex context of Asynchronous Federated Continual Learning (AFCL) [185], where data is not only distributed across multiple nodes (as in federated learning) but also subject to changing distributions over time (as in continual learning). In this asynchronous setting, each node independently progresses through its continual learning tasks, creating unique distribution shifts at different times. We argue that the *rewind* strategy is particularly advantageous in AFCL scenarios, as rewinding on another node can mitigate the exacerbated problem of forgetting.

To test this hypothesis, we implement our strategy on top of the current state-of-the-art approach for AFCL, FedSpace [185]. We replicate their experimental setup, using CIFAR100 divided into 10 tasks of 10 classes each, and maintain the same hyperparameters, except for the number of epochs per round, and without any pretraining. Specifically, we rerun their experiments with $E = 3$ because the default value of 1 was incompatible with the rewind strategy. The experimental results and trends are detailed in Table 4.12 and Figure 4.8, respectively, where the *rewind* strategy is integrated into FedSpace. Additionally, we applied our proposed strategy to the same baseline used in [185], where PASS [234], a continual learning strategy, is adapted to the federated scenario by combining it with FedAvg. In both cases, incorporating the rewind strategy results in enhanced performance, while maintaining computational costs low.

4.2.5 Discussion

This secton presented *FedRewind*, a novel approach that incorporates the *rewind* technique into federated learning (FL) scenarios to address challenges arising from non-i.i.d. data distributions across distributed nodes. By periodically exchanging and rewinding models among

nodes, *FedRewind* mitigates issues related to overfitting on locally skewed data, which can hinder model generalizability and lead to catastrophic forgetting. This method significantly enhances performance by promoting robustness against class imbalances and improving overall model generalization, even in the complex context of Asynchronous Federated Continual Learning (AFCL) which was the overall objective of the thesis.

We first validated *FedRewind* on standard federated learning scenarios, demonstrating significant improvements in performance and generalization over existing methods such as FedAVG, CWT, and RWT. Importantly, these improvements were achieved without increasing computational costs, facilitating seamless integration into existing FL frameworks. We further evaluated our approach in the more extreme context of AFCL, surpassing existing methods. By integrating concepts from continual learning and leveraging cognitive neuroscience principles, *FedRewind* reduces the impact of distribution shifts, providing a robust solution to the challenges posed by non-i.i.d. data in distributed learning environments.

4.2.6 Publications

Palazzo, L., Pennisi M., Salanitri, F. P., Bellitto, G., Palazzo, S., & Spampinato, C. (2024). FedRewind: Rewinding Continual Model Exchange for Decentralized Federated Learning. 2024 27th International Conference on Pattern Recognition (ICPR). [148]

Chapter 5

Conclusions

The process presented in this thesis highlights the importance of integrating Federated Learning (FL) and Continual Learning (CL) approaches to support the practical adoption of AI in healthcare. Although Continual Learning may initially appear unrelated to Federated Learning, this work demonstrates how these strategies together address critical challenges in real-world medical applications.

We started by focusing on Continual Learning to address catastrophic forgetting. Drawing on bio-inspired concepts, such as the role of auxiliary information, we developed methods to support ongoing learning. In the first study, we used an auxiliary, unrelated dataset to prepare the model for upcoming tasks, reducing performance drops between tasks. In the second study, we leveraged knowledge from a fixed, pretrained network to stabilize learning. Finally, we introduced an auxiliary, low-level task (such as saliency prediction), which is less prone to forgetting enhancing CL performance in online continual learning. This initial part of the thesis poses foundations in handling non-i.i.d. data streams.

The thesis then shifted focus to learning in non-i.i.d. environments from a spatial perspective, where data is distributed across multiple nodes. To tackle this, we developed data-oriented solutions that enable data sharing without compromising privacy. In the first study, we achieved this by clustering the latent space of a generative model in way to be complaint to the k-same aggregation method. We then expanded this approach to improve sample diversity by finding privacy-safe pathways through the latent space, ensuring that synthetic data remains distinct from original datasets (thus, preserving patients' privacy).

Finally, we bridged Continual and Federated Learning, applying CL techniques within a fully decentralized federated learning framework. In the first study, we used Experience Replay in a decentralized setting, where each node receives both the model and a buffer of synthetic images to reduce forgetting of previously learned information. In the final study, we adapted Experience Replay to a Federated Continual Learning setting, simulating a "rewind"

mechanism that revisits previous nodes in a round, enhancing retention over time. These results illustrate the complementary strengths of FL and CL, demonstrating that solutions for one challenge can often be adapted for the other, or even for both.

Future research will explore the integration of the developed methods in an unified framework and its valuation on more realistic clinical scenarios where users can not only withdraw data access but also request complete removal from AI models. Addressing this need, the emerging field of Machine Unlearning [124] presents promising solutions for enhancing privacy in deployed AI systems.

In summary, this thesis establishes a strong foundation for future research, offering new directions for applying FL and CL in medical contexts. The methods and results presented here open pathways for both further theoretical exploration and practical implementations in better and more robust healthcare AI.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[2] Davide Abati, Jakub Tomczak, Tijmen Blankevoort, Simone Calderara, Rita Cucchiara, and Babak Ehteshami Bejnordi. Conditional channel gated networks for task-aware continual learning. In *CVPR*, 2020.

[3] David Abramian and Anders Eklund. Refacing: Reconstructing anonymized facial features using GANS. In *16th IEEE International Symposium on Biomedical Imaging, ISBI 2019, Venice, Italy, April 8-11, 2019*, pages 1104–1108. IEEE, 2019.

[4] Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. Knowledge distillation from internal representations. In *AAAI*, 2020.

[5] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720, 2021.

[6] Rahaf Aljundi, Eugene Belilovsky, Tinne Tuytelaars, Laurent Charlin, Massimo Caccia, Min Lin, and Lucas Page-Caccia. Online continual learning with maximal interfered retrieval. In *ANeurIPS*, 2019.

[7] Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11254–11263, 2019.

[8] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In *ANeurIPS*, 2019.

[9] Stefano Allegretti, Federico Bolelli, Federico Pollastri, Sabrina Longhitano, Giovanni Pellacani, and Costantino Grana. Supporting Skin Lesion Diagnosis with Content-Based Image Retrieval. In *ICPR*, 2021.

[10] Guangji Bai, Qilong Zhao, Xiaoyang Jiang, and Liang Zhao. Saliency-guided hidden associative replay for continual learning. In *Associative Memory & Hopfield Networks in 2023*, 2023.

[11] Giovanni Bellitto, Matteo Pennisi, Simone Palazzo, Lorenzo Bonicelli, Matteo Boschini, and Simone Calderara. Effects of auxiliary knowledge on continual learning. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1357–1363. IEEE, 2022.

[12] Giovanni Bellitto, Federica Proietto Salanitri, Simone Palazzo, Francesco Rundo, Daniela Giordano, and Concetto Spampinato. Hierarchical domain-adapted feature learning for video saliency prediction. *International Journal of Computer Vision*, 129:3216–3232, 2021.

[13] Giovanni Bellitto, Federica Proietto Salanitri, **Pennisi, Matteo**, Matteo Boschini, Angelo Porrello, Simone Calderara, Simone Palazzo, and Concetto Spampinato. Saliency-driven experience replay for continual learning. *arXiv preprint arXiv:2403.20086*, 2024. Accepted at NIPS 2024.

[14] Enrique Tomás Martínez Beltrán, Mario Quiles Pérez, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*, 2023.

[15] Ari S Benjamin, David Rolnick, and Konrad Kording. Measuring and regularizing networks in function space. In *ICLR*, 2019.

[16] A. Bischoff-Grethe, I. B. Ozyurt, E. Busa, B. T. Quinn, C. Fennema-Notestine, C. P. Clark, S. Morris, M. W. Bondi, T. L. Jernigan, A. M. Dale, G. G. Brown, and B. Fischl. A technique for the deidentification of structural brain MR images. *Hum Brain Mapp*, 28(9):892–903, Sep 2007.

[17] Ali Borji. Saliency prediction in the deep learning era: Successes, limitations, and future challenges, 2018.

[18] Matteo Boschini, Lorenzo Bonicelli, Pietro Buzzega, Angelo Porrello, and Simone Calderara. Class-incremental continual learning into the extended der-verse. *arXiv preprint arXiv:2201.00766*, 2022.

[19] Matteo Boschini, Lorenzo Bonicelli, Angelo Porrello, Giovanni Bellitto, Matteo Pennisi, Simone Palazzo, Concetto Spampinato, and Simone Calderara. Transfer without forgetting. In *European Conference on Computer Vision*, pages 692–709. Springer, 2022.

[20] Matteo Boschini, Pietro Buzzega, Lorenzo Bonicelli, Angelo Porrello, and Simone Calderara. Continual semi-supervised learning through contrastive interpolation consistency. *arXiv preprint arXiv:2108.06552*, 2021.

[21] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019.

[22] Samuel Budd, Emma C. Robinson, and Bernhard Kainz. A survey on active learning and human-in-the-loop deep learning for medical image analysis. *Medical Image Analysis*, 71:102062, 2021.

[23] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In *ANeurIPS*, 2020.

[24] Pietro Buzzega, Matteo Boschini, Angelo Porrello, and Simone Calderara. Rethinking Experience Replay: a Bag of Tricks for Continual Learning. In *ICPR*, 2020.

[25] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[26] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New Insights on Reducing Abrupt Representation Change in Online Continual Learning. In *ICLR*, 2022.

[27] Sema Candemir et al. Lung segmentation in chest radiographs using anatomical atlases with nonrigid registration. *IEEE TMI*, 33(2):577–590, 2013.

[28] Hyuntak Cha, Jaeho Lee, and Jinwoo Shin. Co2l: Contrastive continual learning. In *ICCV*, 2021.

[29] Ken Chang, Niranjan Balachandar, Carson Lam, Darvin Yi, James Brown, Andrew Beers, Bruce Rosen, Daniel L Rubin, and Jayashree Kalpathy-Cramer. Distributed deep learning networks among institutions for medical imaging. *Journal of the American Medical Informatics Association*, 25(8):945–954, 2018.

[30] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *ECCV*, 2018.

[31] Arslan Chaudhry, Albert Gordo, Puneet Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. In *AAAI*, 2021.

[32] Arslan Chaudhry, Albert Gordo, Puneet Kumar Dokania, Philip Torr, and David Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. In *AAAI*, 2021.

[33] Arslan Chaudhry, Naeemullah Khan, Puneet K Dokania, and Philip HS Torr. Continual learning in low-rank orthogonal subspaces. In *ANeurIPS*, 2020.

[34] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient Lifelong Learning with A-GEM. In *ICLR*, 2019.

[35] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient Lifelong Learning with A-GEM. In *ICLR*, 2019.

[36] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. In *ICML Workshops*, 2019.

[37] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. Gs-wgan: A gradient-sanitized approach for learning differentially private generators. *Advances in Neural Information Processing Systems*, 33:12673–12684, 2020.

[38] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[39] Xuxin Chen, Ximin Wang, Ke Zhang, Kar-Ming Fung, Theresa C. Thai, Kathleen Moore, Robert S. Mannel, Hong Liu, Bin Zheng, and Yuchen Qiu. Recent advances and clinical applications of deep learning in medical image analysis. *Medical Image Analysis*, 79:102444, 2022.

[40] Joseph Cichon and Wen-Biao Gan. Branch-specific dendritic ca 2+ spikes cause persistent synaptic plasticity. *Nature*, 2015.

[41] Noel CF Codella et al. Skin lesion analysis toward melanoma detection. In *IEEE ISBI*, 2018.

[42] Joseph Paul Cohen, Paul Morrison, Lan Dao, Karsten Roth, Tim Q Duong, and Marzyeh Ghassemi. Covid-19 image data collection: Prospective predictions are the future. *arXiv 2006.11988*, 2020.

[43] Marc Combalia et al. Bcn20000: Dermoscopic lesions in the wild. *arXiv:1908.02288*, 2019.

[44] Ittai Dayan et al. Federated learning for predicting clinical outcomes in patients with COVID-19. *Nature medicine*, 27, 2021.

[45] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE TPAMI*, 2021.

[46] Matthias De Lange and Tinne Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In *ICCV*, 2021.

[47] Harm De Vries, Florian Strub, Jérémie Mary, Hugo Larochelle, Olivier Pietquin, and Aaron C Courville. Modulating early visual processing by language. In *ANeurIPS*, 2017.

[48] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[49] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[50] Mohammad Mahdi Derakhshani, Xiantong Zhen, Ling Shao, and Cees Snoek. Kernel continual learning. In *ICML*, 2021.

[51] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[52] Prithviraj Dhar, Rajat Vikram Singh, Kuan-Chuan Peng, Ziyan Wu, and Rama Chellappa. Learning without memorizing. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 5138–5146, 2019.

[53] J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 2012.

[54] Jiahua Dong, Lixu Wang, Zhen Fang, Gan Sun, Shichao Xu, Xiao Wang, and Qi Zhu. Federated class-incremental learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10164–10173, 2022.

[55] Richard Droste, Jianbo Jiao, and J Alison Noble. Unified image and video saliency modeling. In *European Conference on Computer Vision*, 2020.

[56] Sayna Ebrahimi, Franziska Meier, Roberto Calandra, Trevor Darrell, and Marcus Rohrbach. Adversarial continual learning. In *ECCV*, 2020.

[57] Sayna Ebrahimi, Suzanne Petryk, Akash Gokul, William Gan, Joseph E Gonzalez, Marcus Rohrbach, and Trevor Darrell. Remembering for the right reasons: Explanations reduce catastrophic forgetting. *Applied AI Letters*, 2021.

[58] David Evans et al. A pragmatic introduction to secure multi-party computation. *Foundations and Trends in Privacy and Security*, 2(2-3):70–246, 2018.

[59] Mehrdad Farajtabar, Navid Azizan, Alex Mott, and Ang Li. Orthogonal gradient descent for continual learning. In *AISTATS*, 2020.

[60] Sebastian Farquhar and Yarin Gal. Towards Robust Evaluations of Continual Learning. In *ICML Workshops*, 2018.

[61] Ines Feki et al. Federated learning for COVID-19 screening from chest x-ray images. *Applied Soft Computing*, 106:107330, 2021.

[62] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

[63] Robert M French. Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In *Proc. Annu. Conf. Cogn. Sci. Soc.*, 1991.

[64] Tommaso Furlanello, Zachary C Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *ICML*, 2018.

[65] Qiankun Gao, Chen Zhao, Yifan Sun, Teng Xi, Gang Zhang, Bernard Ghanem, and Jian Zhang. A unified continual learning framework with general parameter-efficient tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11483–11493, October 2023.

[66] Zheyao Gao, Fuping Wu, Weiguo Gao, and Xiahai Zhuang. A new framework of swarm learning consolidating knowledge from multi-center non-iid data for medical image segmentation. *IEEE TMI*, pages 1–1, 2022.

[67] Jonas Geiping et al. Inverting gradients-how easy is it to break privacy in federated learning? *NeurIPS*, 2020.

[68] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *ANeurIPS*, 2014.

[69] Ian J. Goodfellow, Mehdi Mirza, Xia Da, Aaron C. Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgeting in gradient-based neural networks. *ICLR*, 2014.

[70] Shangwei Guo, Tianwei Zhang, Guowen Xu, Han Yu, Tao Xiang, and Yang Liu. Topology-aware differential privacy for decentralized image classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[71] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017.

[72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[73] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *ICPR*, 2016.

[74] Dan Hendrycks, Kimin Lee, and Mantas Mazeika. Using pre-training can improve model robustness and uncertainty. In *ICML*, 2019.

[75] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *ICCV*, 2019.

[76] Martin Heusel et al. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 2017.

[77] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NeurIPS Workshops*, 2015.

[78] Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *CVPR*, 2019.

[79] Feiyan Hu, Simone Palazzo, Federica Proietto Salanitri, Giovanni Bellitto, Morteza Moradi, Concetto Spampinato, and Kevin McGuinness. Tinyhd: Efficient video saliency prediction with heterogeneous decoders using hierarchical maps distillation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023.

[80] Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In *ICML*, 2018.

[81] Intersoft Consulting. GDPR, 2016. Accessed: 2024-9-27. Available at: https://gdpr-info.eu.

[82] Jeremy Irvin et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *AAAI*, 2019.

[83] Stefan Jaeger et al. Automatic tuberculosis screening using chest radiographs. *IEEE TMI*, 33(2):233–245, 2013.

[84] Stefan Jaeger et al. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative imaging in medicine and surgery*, 4(6):475, 2014.

[85] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017.

[86] Yunhun Jang, Hankook Lee, Sung Ju Hwang, and Jinwoo Shin. Learning what and where to transfer. In *ICML*, 2019.

[87] Minkyu Jeon, Hyeonjin Park, Hyunwoo J Kim, Michael Morley, and Hyunghoon Cho. k-salsa: k-anonymous synthetic averaging of retinal images via local style alignment. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXI*, pages 661–678. Springer, 2022.

[88] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. Memguard: Defending against black-box membership inference attacks via adversarial examples. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 259–274, 2019.

[89] Ming Jiang, Shengsheng Huang, Juanyong Duan, and Qi Zhao. Salicon: Saliency in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1072–1080, 2015.

[90] Sangwon Jung, Hongjoon Ahn, Sungmin Cha, and Taesup Moon. Continual learning with node-importance based adaptive group sparse regularization. In *ANeurIPS*, 2020.

[91] Peter Kairouz et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14, 2021.

[92] Shivam Kalra, Junfeng Wen, Jesse C Cresswell, Maksims Volkovs, and Hamid R Tizhoosh. Decentralized federated learning through proxy model sharing. *Nature communications*, 14(1):2899, 2023.

[93] Jeffrey D Karpicke and Janell R Blunt. Retrieval practice produces more learning than elaborative studying with concept mapping. *Science*, 331(6018):772–775, 2011.

[94] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems*, 33:12104–12114, 2020.

[95] Tero Karras et al. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020.

[96] Tero Karras et al. Training generative adversarial networks with limited data. In *NeurIPS*, 2020.

[97] Sohier Dane Karthik, Maggie. Aptos 2019 blindness detection, 2019.

[98] Amirhossein Kazerouni, Ehsan Khodapanah Aghdam, Moein Heidari, Reza Azad, Mohsen Fayyaz, Ilker Hacihaliloglu, and Dorit Merhof. Diffusion models in medical imaging: A comprehensive survey. *Medical Image Analysis*, 88:102846, 2023.

[99] Ronald Kemker and Christopher Kanan. Fearnet: Brain-inspired model for incremental learning. *arXiv preprint arXiv:1711.10563*, 2017.

[100] Chris Dongjoo Kim, Jinseo Jeong, Sangwoo Moon, and Gunhee Kim. Continual learning on noisy data streams via self-purified replay. In *ICCV*, 2021.

[101] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath,

D. Kumaran, and R. Hadsell. Overcoming catastrophic forgetting in neural networks. *Proc Natl Acad Sci U S A*, 114(13):3521–3526, Mar 2017.

[102] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *PNAS*, 2017.

[103] A. Kohn. Visual adaptation: physiology, mechanisms, and functional benefits. *J Neurophysiol*, 2007.

[104] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *ICML*, 2019.

[105] Jai Kotia, Adit Kotwal, Rishika Bharti, and Ramchandra Mangrulkar. Few shot learning for medical imaging. *Machine learning algorithms for industrial applications*, pages 107–132, 2021.

[106] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[107] D. Kumaran, D. Hassabis, and J. L. McClelland. What Learning Systems do Intelligent Agents Need? Complementary Learning Systems Theory Updated. *Trends Cogn Sci*, 20(7):512–534, Jul 2016.

[108] Anusha Lalitha et al. Peer-to-peer federated learning on graphs. *arXiv:1901.11173*, 2019.

[109] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory G. Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *IEEE TPAMI*, 2021.

[110] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[111] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

[112] Timothée Lesort. Continual feature selection: Spurious features in continual learning. *arXiv preprint arXiv:2203.01012*, 2022.

[113] Stephan Lewandowsky and Shu-Chen Li. Catastrophic interference in neural networks: Causes, solutions, and data. In *Interference and inhibition in cognition*. Elsevier, 1995.

[114] N. Li, D. D. Cox, D. Zoccolan, and J. J. DiCarlo. What response properties do individual neurons need to underlie position and clutter "invariant" object recognition? *J Neurophysiol*, 102(1):360–376, Jul 2009.

[115] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722, 2021.

[116] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

[117] Wenqi Li et al. Privacy-preserving federated brain tumour segmentation. In *International workshop on machine learning in medical imaging*, pages 133–141. Springer, 2019.

[118] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv:2102.07623*, 2021.

[119] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 2017.

[120] Xiangru Lian et al. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *NeurIPS*, 2017.

[121] Akis Linardos, Matthias Kümmerer, Ori Press, and Matthias Bethge. Deepgaze iie: Calibrated prediction in and out-of-domain for state-of-the-art saliency modeling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12919–12928, 2021.

[122] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60–88, 2017.

[123] Xialei Liu, Jiang-Tian Zhai, Andrew D. Bagdanov, Ke Li, and Mingg-Ming Cheng. Task-adaptive saliency guidance for exemplar-free class incremental learning. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[124] Ziyao Liu, Yu Jiang, Jiyuan Shen, Minyi Peng, Kwok-Yan Lam, Xingliang Yuan, and Xiaoning Liu. A survey on federated unlearning: Challenges, methods, and future directions. *ACM Computing Surveys*, 57(1):1–38, 2024.

[125] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *ANeurIPS*, 2018.

[126] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017.

[127] David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *ANeurIPS*, 2017.

[128] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[129] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *Neurocomputing*, 2022.

[130] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *CVPR*, 2018.

[131] J. C. Martinez-Trujillo and S. Treue. Feature-based attention increases the selectivity of population responses in primate visual cortex. *Curr Biol*, 14(9):744–751, May 2004.

[132] J. L. McClelland, B. L. McNaughton, and R. C. O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychol Rev*, 102(3):419–457, Jul 1995.

[133] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychol. Learn. Motiv.*, volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989.

[134] Brendan McMahan et al. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[135] Ken McRae and Phil A Hetherington. Catastrophic interference is eliminated in pre-trained networks. In *Proc. Annu. Conf. Cogn. Sci. Soc.*, 1993.

[136] Blaž Meden, Žiga Emeršič, Vitomir Štruc, and Peter Peer. k-same-net: k-anonymity with generative deep neural networks for face deidentification. *Entropy*, 20(1):60, 2018.

[137] Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. An empirical investigation of the role of pre-training in lifelong learning. In *ICML*, 2021.

[138] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014.

[139] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Razvan Pascanu, and Hassan Ghasemzadeh. Understanding the role of training regimes in continual learning. In *ANeurIPS*, 2020.

[140] Alessio Monti, Angelo Porrello, Simone Calderara, Pasquale Coscia, Lamberto Ballan, and Rita Cucchiara. How many observations are enough? knowledge distillation for trajectory forecasting. In *CVPR*, 2022.

[141] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. Subclass distillation. *arXiv preprint arXiv:2002.03936*, 2020.

[142] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 634–646, 2018.

[143] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *ANeurIPS*, 2011.

[144] Joshua New, Leda Cosmides, and John Tooby. Category-specific attention for animals reflects ancestral priorities, not expertise. *Proceedings of the National Academy of Sciences*, 104(42):16598–16603, 2007.

[145] Andrew Ng, Michael Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14, 2001.

[146] Alex Nichol and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[147] Luke Oakden-Rayner. Exploring large-scale public medical image datasets. *Academic radiology*, 27(1):106–112, 2020.

[148] Luca Palazzo, **Pennisi, Matteo**, Federica Proietto Salanitri, Giovanni Bellitto, Simone Palazzo, and Concetto Spampinato. Fedrewind: Rewinding continual model exchange for decentralized federated learning. In *2024 27th International Conference on Pattern Recognition (ICPR)*, 2024.

[149] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 2009.

[150] Junjie Pang et al. Collaborative city digital twin for the covid-19 pandemic: A federated learning solution. *Tsinghua science and technology*, 26(5):759–771, 2021.

[151] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Netw.*, 2019.

[152] Jongchan Park, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Bam: Bottleneck attention module. In *British Machine Vision Conference*, 2018.

[153] Matteo Pennisi, Simone Palazzo, and Concetto Spampinato. Self-improving classification performance through gan distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1640–1648, October 2021.

[154] Matteo Pennisi, Federica Proietto Salanitri, Giovanni Bellitto, Simone Palazzo, Ulas Bagci, and Concetto Spampinato. A privacy-preserving walk in the latent space of generative models for medical applications. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 422–431. Springer, 2023.

[155] Matteo Pennisi, Federica Proietto Salanitri, Simone Palazzo, Carmelo Pino, Francesco Rundo, Daniela Giordano, and Concetto Spampinato. Gan latent space manipulation and aggregation for federated learning in medical imaging. In *International Workshop on Distributed, Collaborative, and Federated Learning*, pages 68–78. Springer, 2022.

[156] Matteo Pennisi, Federica Proietto Salanitri, Giovanni Bellitto, Bruno Casella, Marco Aldinucci, Simone Palazzo, and Concetto Spampinato. Feder: Federated learning through experience replay and privacy-preserving data synthesis. *Computer Vision and Image Understanding*, 238:103882, 2024.

[157] Quang Pham, Chenghao Liu, and Steven Hoi. Dualnet: Continual learning, fast and slow. In *ANeurIPS*, 2021.

[158] Angelo Porrello, Stefano Vincenzi, Pietro Buzzega, Simone Calderara, Annamaria Conte, Carla Ippoliti, Luca Candeloro, Alessio Di Lorenzo, and Andrea Capobianco Dondona. Spotting insects from satellites: modeling the presence of culicoides imicola through deep cnns. In *SITIS*, 2019.

[159] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. GDumb: A simple approach that questions our progress in continual learning. In *ECCV*, 2020.

[160] Jathushan Rajasegaran, Munawar Hayat, Salman H Khan, Fahad Shahbaz Khan, and Ling Shao. Random path selection for continual learning. In *ANeurIPS*, 2019.

[161] Jean-Francois Rajotte, Sumit Mukherjee, Caleb Robinson, Anthony Ortiz, Christopher West, Juan M Lavista Ferres, and Raymond T Ng. Reducing bias and increasing utility by federated generative modeling of medical images using a centralized adversary. In *Proceedings of the Conference on Information Technology for Social Good*, pages 79–84, 2021.

[162] Vinay Venkatesh Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. In *ICLR*, 2021.

[163] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychol. Rev.*, 1990.

[164] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *CVPR*, 2017.

[165] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *ANeurIPS*, 2015.

[166] Slobodan Ribaric and Nikola Pavesic. An overview of face de-identification in still images and videos. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 04, pages 1–6, 2015.

[167] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference. In *ICLR*, 2019.

[168] Matthew Riemer, Tim Klinger, Djallel Bouneffouf, and Michele Franceschini. Scalable recollections for continual lifelong learning. In *AAAI*, 2019.

[169] Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Conn. Sci.*, 1995.

[170] A Harry Robinson and Colin Cherry. Results of a prototype television bandwidth compression scheme. In *IEEE*, 1967.

[171] Henry L Roediger and Andrew C Butler. The critical role of retrieval practice in long-term retention. *Trends in cognitive sciences*, 15(1):20–27, 2011.

[172] David Rolnick et al. Experience replay for continual learning. *NeurIPS*, 2019.

[173] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. In *ICLR*, 2015.

[174] G. Rothschild, E. Eban, and L. M. Frank. A cortical-hippocampal-cortical loop of information processing during memory consolidation. *Nat. Neurosci.*, 2017.

[175] Abhijit Guha Roy et al. Braintorrent: A peer-to-peer environment for decentralized federated learning. *arXiv:1905.06731*, 2019.

[176] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[177] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016.

[178] Gobinda Saha and Kaushik Roy. Saliency guided experience packing for replay in continual learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 5273–5283, 2023.

[179] Sofia Sakellaridi, Vassilios N. Christopoulos, Tyson Aflalo, Kelsie W. Pejsa, Emily R. Rosario, Debra Ouellette, Nader Pouratian, and Richard A. Andersen. Intrinsic variable learning for brain-machine interface control by human anterior intraparietal cortex. *Neuron*, 2019.

[180] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *ICML*, 2018.

[181] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. *Int. J. Comput. Vision*, 2019.

[182] Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. Overcoming catastrophic forgetting with hard attention to the task. In *ICML*, 2018.

[183] Khadija Shaheen, Muhammad Abdullah Hanif, Osman Hasan, and Muhammad Shafique. Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. *Journal of Intelligent & Robotic Systems*, 2022.

[184] Micah J Sheller, G Anthony Reina, Brandon Edwards, Jason Martin, and Spyridon Bakas. Multi-institutional deep learning modeling without sharing patient data: A feasibility study on brain tumor segmentation. In *International MICCAI Brainlesion Workshop*, pages 92–104. Springer, 2018.

[185] Donald Shenaj, Marco Toldo, Alberto Rigon, and Pietro Zanuttigh. Asynchronous federated continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5054–5062, 2023.

[186] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *ANeurIPS*, 2017.

[187] Neta Shoham et al. Overcoming forgetting in federated learning on non-iid data. *arXiv:1910.07796*, 2019.

[188] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[189] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.

[190] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[191] James Smith, Jonathan Balloch, Yen-Chang Hsu, and Zsolt Kira. Memory-efficient semi-supervised continual learning: The world is its own replay buffer. In *IJCNN*, 2021.

[192] James Seale Smith, Leonid Karlinsky, Vyshnavi Gutta, Paola Cascante-Bonilla, Donghyun Kim, Assaf Arbelle, Rameswar Panda, Rogerio Feris, and Zsolt Kira. Coda-prompt: Continual decomposed attention-based prompting for rehearsal-free continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11909–11919, 2023.

[193] Joonyoung Song and Jong Chul Ye. Federated cyclegan for privacy-preserving image-to-image translation. *arXiv:2106.09246*, 2021.

[194] Stanford. Tiny ImageNet Challenge (CS231n), 2015. https://www.kaggle.com/c/tiny-imagenet.

[195] Tao Sun, Dongsheng Li, and Bao Wang. Decentralized federated averaging. *arXiv:2104.11375*, 2021.

[196] Latanya Sweeney. k-anonymity: A model for protecting privacy. *International journal of uncertainty, fuzziness and knowledge-based systems*, 10(05):557–570, 2002.

[197] Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8432–8440, 2022.

[198] S. Treue and J. C. nez Trujillo. Feature-based attention influences motion processing gain in macaque visual cortex. *Nature*, 399(6736):575–579, Jun 1999.

[199] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

[200] Gido M van de Ven and Andreas S Tolias. Three continual learning scenarios. In *NeurIPS Workshops*, 2018.

[201] Gido M van de Ven, Tinne Tuytelaars, and Andreas S Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 2022.

[202] Bas H.M. van der Velden, Hugo J. Kuijf, Kenneth G.A. Gilhuijs, and Max A. Viergever. Explainable artificial intelligence (xai) in deep learning-based medical image analysis. *Medical Image Analysis*, 79:102470, 2022.

[203] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 2019.

[204] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *ANeurIPS*, 2016.

[205] Jeffrey S Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 1985.

[206] Hongyi Wang et al. Federated learning with matched averaging. *arXiv:2002.06440*, 2020.

[207] Kafeng Wang, Xitong Gao, Yiren Zhao, Xingjian Li, Dejing Dou, and Cheng-Zhong Xu. Pay attention to features, transfer learn faster cnns. In *ICLR*, 2019.

[208] Xiaosong Wang et al. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *CVPR*, 2017.

[209] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pages 2512–2520. IEEE, 2019.

[210] Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVI*, page 631–648. Springer-Verlag, 2022.

[211] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 139–149, June 2022.

[212] Ziqiang Wang, Zhi Liu, Gongyang Li, Yang Wang, Tianhong Zhang, Lihua Xu, and Jijun Wang. Spatio-temporal self-attention network for video saliency prediction. *IEEE Transactions on Multimedia*, 2021.

[213] Tobias Wink and Zoltan Nochta. An approach for peer-to-peer federated learning. In *2021 51st Annual IEEE/IFIP DSN-W*, 2021.

[214] Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *CVPR*, 2019.

[215] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv:1802.06739*, 2018.

[216] Chugui Xu, Ju Ren, Deyu Zhang, Yaoxue Zhang, Zhan Qin, and Kui Ren. Ganobfuscator: Mitigating information leakage under gan via differential privacy. *IEEE Trans. Inf. Forensics Secur.*, 14(9):2358–2371, 2019.

[217] Qiang Yang et al. Federated machine learning: Concept and applications. *ACM TIST*, 10(2):1–19, 2019.

[218] Xin Yang, Hao Yu, Xin Gao, Hao Wang, Junbo Zhang, and Tianrui Li. Federated continual learning via knowledge fusion: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[219] Xin Yao and Lifeng Sun. Continual local training for better initialization of federated models. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1736–1740. IEEE, 2020.

[220] Xin Yi, Ekta Walia, and Paul Babyn. Generative adversarial network in medical imaging: A review. *Medical Image Analysis*, 58:101552, 2019.

[221] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *CVPR*, 2017.

[222] Jaehong Yoon, Wonyong Jeong, Giwoong Lee, Eunho Yang, and Sung Ju Hwang. Federated continual learning with weighted inter-client transfer. In *International Conference on Machine Learning*, pages 12073–12086. PMLR, 2021.

[223] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

[224] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *ANeurIPS*, 2014.

[225] Lu Yu, Bartlomiej Twardowski, Xialei Liu, Luis Herranz, Kai Wang, Yongmei Cheng, Shangling Jui, and Joost van de Weijer. Semantic drift compensation for class-incremental learning. In *CVPR*, 2020.

[226] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International conference on machine learning*, pages 7252–7261. PMLR, 2019.

[227] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

[228] John R Zech et al. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11), 2018.

[229] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *ICML*, 2017.

[230] Longling Zhang, Bochen Shen, Ahmed Barnawi, Shan Xi, Neeraj Kumar, and Yi Wu. Feddpgan: federated differentially private generative adversarial networks framework for the detection of covid-19 pneumonia. *Information Systems Frontiers*, 23(6):1403–1415, 2021.

[231] Richard Zhang et al. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[232] Bo Zhao et al. IDLG: Improved deep leakage from gradients. *arXiv:2001.02610*, 2020.

[233] Chen Zhu, Zheng Xu, Mingqing Chen, Jakub Konečnỳ, Andrew Hard, and Tom Goldstein. Diurnal or nocturnal? federated learning of multi-branch networks from periodically shifting distributions. In *International Conference on Learning Representations*, 2022.

[234] Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5871–5880, 2021.

[235] Ligeng Zhu et al. Deep leakage from gradients. *NeurIPS*, 2019.

[236] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *International conference on machine learning*, pages 12878–12889. PMLR, 2021.

[237] Jacob Ziv and Abraham Lempel. A universal algorithm for sequential data compression. *IEEE Trans. Inf. Theory*, 1977.