Journal of Artificial Intelligence Research 80 (2024) 377-417 Submitted 12/2023; published 06/2024

Similarity-Based Adaptation for Task-Aware and Task-Free Continual Learning

Tameem Adel TAMEEM.HESHAM@GMAIL.COM
National Physical Laboratory (NPL), Maxwell Centre, University of Cambridge
JJ Thomson Avenue, Cambridge, CB3 0HE, United Kingdom

Abstract

Continual learning (CL) is a paradigm which addresses the issue of how to learn from sequentially arriving tasks. The goal of this paper is to introduce a CL framework which can both learn from a global multi-task architecture and locally adapt this learning to the task at hand. In addition to the global knowledge, we conjecture that it is also beneficial to further focus on the most relevant pieces of previous knowledge. Using a prototypical network as a proxy, the proposed framework bases its adaptation on the similarity between the current data stream and the previously encountered data. We develop two algorithms, one for the standard task-aware CL and another for the more challenging task-free setting where boundaries between tasks are unknown. We correspondingly derive a generalization upper bound on the error of an upcoming task. Experiments demonstrate that the introduced algorithms lead to improved performance on several CL benchmarks.

1. Introduction

Continual learning (also called incremental learning or lifelong learning) is an online paradigm where (non i.i.d.) data continuously arrive. The data distribution can potentially change over time (Schlimmer & Fisher, 1986; Sutton & Whitehead, 1993; Ring, 1995, 1997; Kirkpatrick et al., 2017; Lee et al., 2017; Shin et al., 2017; Schmidhuber, 2018; Ahn et al., 2019; Riemer et al., 2019; Buzzega et al., 2020; Mirzadeh et al., 2020; Yoon et al., 2020; Beaulieu et al., 2021; Guo et al., 2022; Mundt et al., 2022; Romero et al., 2022b; Wu et al., 2022; Evron et al., 2023; Wang et al., 2023). Upon learning from upcoming data, the continual learner should not forget the knowledge acquired from previous data, a phenomenon referred to as catastrophic forgetting (McCloskey & Cohen, 1989; Ratcliff, 1990; Robins, 1993, 1995; French, 1999; Pape et al., 2011; Srivastava et al., 2013; Achille et al., 2018; Diaz-Rodriguez et al., 2018; Kemker et al., 2018; Zeno et al., 2018; Parisi et al., 2019; Pfulb & Gepperth, 2019; Ebrahimi et al., 2020a; Gupta et al., 2020b; Banayeeanzade et al., 2021; Ke et al., 2021; Ostapenko et al., 2021; Wang et al., 2021; Kang et al., 2022; Karakida & Akaho, 2022; Lin et al., 2022; Miao et al., 2022; Yasar & Iqbal, 2023). The continual learner should as well adapt to the distributional shift occurring across data streams. A balance must therefore be achieved in continual learning (CL) between stability, to preserve previous knowledge, and adaptation (the stability-plasticity dilemma). In addition, updates must be performed incrementally where the data available at each stage comprise solely the new data. Due to privacy, security and computational constraints, access to the old data is prohibited (Adel et al., 2020; Smith et al., 2023).

Upon encountering experiences, humans are more adept than machines at learning from the past given a limited scale of training data (Taylor & Stone, 2009; Chen & Liu, 2016; Finn et al., 2017; Li et al., 2018; Rostami et al., 2020a). One reason is our sheer aptitude to precisely assign different relevance levels to knowledge accumulated from the past, with respect to the upcoming task[1]. A person asked to carry an object with one hand can efficiently learn which characteristics of the object (e.g. center of gravity) are similar to which objects encountered in the past. Therefore, the latter (i.e. more similar) objects should be more relevant to the current task. We imitate this here by developing a continual learner with general (global) learning capacity as well as a (local) adaptive mechanism customized for each upcoming task via zooming in on a (most) relevant task from the past.

A common approach to CL is based on multi-task transfer. The standard practice in this multi-task learning approach (Caruana, 1997; Heskes, 2000; Bakker & Heskes, 2003; Yoon et al., 2018; Wu et al., 2023) is to divide the architecture into (i) a slowly evolving, shared part (usually at the bottom of the architecture) which is globally accessible by all tasks; and (ii) rapidly evolving task-specific parts, including the top classification layer (head) of each task. In its standard form, this division of the architecture makes it tricky to highlight a specific experience (task) from the past as particularly relevant for the current/upcoming task. In addition, this rigid specification of the shared and task-specific components constrains the form of task-transfer that can take place. Furthermore, this approach is prone to underfitting when the tasks are heterogeneous (Requeima et al., 2019). Hence, an adaptation mechanism is needed to establish a potentially more flexible multi-task architecture.

The architecture of the proposed CL framework, which we refer to as Similarity-based adaptation for Task-Aware and task-Free continual learning (*STAF*), comprises shared, task-specific and adaptive components. The output of the shared components is adapted in *STAF*. Thus, the optimization performed to compute the task-specific parameters depends not only on the shared parameters but also on the adaptation mechanism which takes into consideration the similarity between the task at hand and previous tasks. The adaptation mechanism is based on an auxiliary prototypical network (PN, Snell et al., 2017). We develop a learning procedure for the PN which particularly suits our adaptation purpose. In brief, the PN is responsible for identifying the most relevant (i.e. similar) previous task to the current task.

We propose two algorithms: task-aware *STAF* and task-free *STAF*. In task-aware CL, the data is presented to the learner in the form of tasks between which hard boundaries are observed (Aljundi et al., 2019b). The training procedure is accordingly divided into phases where each phase corresponds to a task. Task-aware CL has led to advances in understanding how to transfer knowledge over sequentially arriving tasks. Nonetheless, since task boundaries are not observed in several real-world applications of CL, the need to learn under the more challenging task-free CL setting recently arose (Caccia et al., 2020b). The adaptation mechanism proposed by *STAF* enables it to discover task boundaries.

We derive a CL generalization upper bound to analyze the error of an upcoming task. Our upper bound corroborates the idea of involving the most relevant task to the upcoming task in the CL optimization. We also empirically evaluate the proposed *STAF*, with both its task-aware and task-free algorithms, on several established benchmarks. Experiments demonstrate that *STAF* achieves state-of-the-art performance in terms of both overall CL classification accuracy and reducing catastrophic forgetting. We also perform an ablation study to gauge the significance of the adaptation mechanism. Information about related works can be found in Section 2.

Our principal contributions can be summarized as follows: (1) A CL framework which reinforces the global utilization of knowledge with the proposed local adaptation mechanism that relies on

1. As clarified later, in task-aware CL, a task encompasses the data stream exposed to the learner at a specific time step. The data distribution can change across tasks, but not within the same task.

evaluating the similarity between each upcoming task and the previous tasks (Section 3). (2) An extension of the proposed framework to task-free CL by providing an algorithm which can function without observing task boundaries (Section 4). (3) A generalization upper bound on the error of an upcoming task (Section 5). (4) State-of-the-art results on several CL benchmarks which demonstrate the effectiveness of the proposed framework in learning both with and without task boundaries, as measured both by the overall classification accuracy and by the degree to which catastrophic forgetting is mitigated (Section 6).

2. Related Work

Numerous CL algorithms have been developed in the literature (Srivastava et al., 2013; Farquhar & Gal, 2018; Kim et al., 2018; Morgado & Vasconcelos, 2019; Benavides-Prado et al., 2020; Parisi & Lomonaco, 2020; Ahn et al., 2021; Ayub & Wagner, 2021; Cha et al., 2021a; Derakhshani et al., 2021; Ehret et al., 2021; Hurtado et al., 2021; Kapoor et al., 2021; Mao et al., 2021; Pham et al., 2021; Tang & Matteson, 2021; Yoon et al., 2021; Benavides-Prado & Riddle, 2022; Madaan et al., 2022; Ramesh & Chaudhari, 2022; Romero et al., 2022a; Skantze & Willemsen, 2022; Wang et al., 2022b; Gaya et al., 2023; Mundt et al., 2023). On a high level, there are three principal approaches to continual learning (CL): memory-based, regularization-based and architecture-based (Pan et al., 2020; Parisi & Lomonaco, 2020; De Lange et al., 2021; De Lange & Tuytelaars, 2021; Krishnan & Balaprakash, 2021; Mehta et al., 2021; Liu & Liu, 2022).

Some CL algorithms are memory-based, which typically store (or generate) previous data in an episodic memory (Ratcliff, 1990; Thrun, 1996; Schmidhuber, 2013; Hattori, 2014; Mocanu et al., 2016; Kamra et al., 2017; Rolnick et al., 2018; Chaudhry et al., 2019a; Smith et al., 2019; Titsias et al., 2019; Ebrahimi et al., 2020b; Cory et al., 2021; Deng et al., 2021; Jin et al., 2021; Shim et al., 2021; Saha et al., 2021; Arani et al., 2022; Masarczyk et al., 2022; Sun et al., 2022; Wang et al., 2022; Ho et al., 2023). Most of the memory-based CL algorithms rely on storing or replaying data from previous tasks, which leads to an overhead in terms of storage (Shin et al., 2017; Choi et al., 2019; Teng & Dasgupta, 2019). In addition, there is a computational overhead on the methods performing optimization to select which previous examples to replay (Titsias et al., 2019). Other methods are based on learning a generative model to generate observations from previous tasks (Schmidhuber, 2013; Mocanu et al., 2016; Kamra et al., 2017; Rebuffi et al., 2017; Shin et al., 2017; van de Ven & Tolias, 2018; Wu et al., 2018; Rostami et al., 2020b). This approach leads to less overhead in terms of storage, albeit at the added cost of the required generative modelling and its training procedure. Based on generative adversarial networks (GANs, Goodfellow et al., 2014b; Goodfellow, 2016), a deep generative model is trained by Shin et al. (2017) to mimic data from previous tasks. Other GAN-based CL algorithms include Khan et al. (2023) where the feature drift, between the generated and original features, is reduced via a distillation and matching latent modelling procedure, which ultimately enhances the ability to learn and generate complex data.

A gradient episodic memory is adopted by Lopez-Paz and Ranzato (2017) to store gradients of the previous tasks. A controlled sampling of memories is developed for replay by Aljundi et al. (2019a). Sample selection (from previous tasks) is cast as a constraint reduction problem by Aljundi et al. (2019c). Methods based on the replay approach have as well been proposed in reinforcement learning (Isele & Cosgun, 2018; Rolnick et al., 2018). Some CL methods are based on selecting coresets from previous tasks via an optimization procedure (Borsos et al., 2020; Nguyen et al., 2018; Tiwari et al., 2022; Yoon et al., 2022; Hao et al., 2023). In the algorithm introduced by Kamra et al.

(2017), catastrophic forgetting is alleviated via a dual memory, which emulates the way the human brain works. Saha et al. (2021) learns upcoming tasks by taking gradient steps in the direction which is orthogonal to the subspaces of the gradients from previous tasks. Kernel ridge regression is utilized to construct an episodic memory by Derakhshani et al. (2021).

Unlike the memory-based approach, our proposed framework, *STAF*, results in less memory and run-time overhead than memory-based CL algorithms since *STAF* does not store, nor generate, any previous data. Instead, we develop a PN which acts as a proxy to *learn* information about previous tasks.

Another CL approach is based on regularization (Li & Hoiem, 2016; Aljundi et al., 2018; Vuorio et al., 2018; Aljundi et al., 2019d; Hung et al., 2019; Park et al., 2019; Benzing, 2020; von Oswald et al., 2020; Kessler et al., 2021; Mirzadeh et al., 2021; von Oswald et al., 2021; Lyu et al., 2023) where some parameters are granted more freedom to change than others, in order to provide a balance between learnability and stability, and to mitigate the impact of catastrophic forgetting. Parameters hugely influencing the prediction are protected against massive changes, whereas the rest of the parameters are allowed to change more freely.

Examples of seminal regularization-based CL algorithms include the Elastic Weight Consolidation (EWC) algorithm by Kirkpatrick et al. (2017) in which a quadratic penalty is imposed on the difference between parameter values of the old and new tasks. However, a high level of hand tuning is required in EWC. Confident fitting to uncertain knowledge is regularized by maximum entropy in the method introduced by Kim et al. (2019). The regularizer introduced by Zenke et al. (2017) is based on synapses in which importance measures are computed during training, based on their corresponding contributions to the variation in the global loss. In order to mitigate catastrophic forgetting, more important synapses are granted less freedom to change. In the Orthogonal Gradient Descent (OGD) algorithm by Farajtabar et al. (2019), catastrophic forgetting is addressed via projecting the gradients of the predictions belonging to the current task onto the subspace of the respective gradients from previous tasks. Cha et al. (2021a) introduced the Classifier-Projection Regularization (CPR) algorithm which is inspired by ideas from information theory. CPR is based on adding a regularization term that maximizes the entropy of the classifier's output probability. In the algorithm developed by Lyu et al. (2023), another regularization term is proposed in order to ultimately warrant a certain degree of stability while training the continual learner.

Several regularization-based CL algorithms are based on probabilistic inference (Nguyen et al., 2018; Adel et al., 2020; Ebrahimi et al., 2020a; Egorov et al., 2021; Henning et al., 2021; Kao et al., 2021; Loo et al., 2021; Wang et al., 2023). A seminal example of this category is the variational continual learning (VCL) algorithm by Nguyen et al. (2018) where sequential variational inference (VI) and Monte Carlo VI are fused to model CL. Kessler et al. (2021) developed a solution based on Bayesian Neural Networks (BNNs) where the complexity of the BNN is automatically controlled via a hierarchical Indian Buffet Process (IBP). The moments of the BNN posterior are matched incrementally in the algorithm proposed by Lee et al. (2017). Another Bayesian framework has been introduced by Kumar et al. (2021) where regularization is based on priors and sparse subsets of weights learned by different tasks. Another sparsity-based regularizer has been developed by Ghosh et al. (2018) where a horseshoe prior is used over some pre-activations of a BNN to turn off the respective network nodes. Other sparsity-based CL regularizers have been developed in the literature (Abati et al., 2020), like Aljundi et al. (2019d) where sparsity is encouraged on the neuron activations to mitigate catastrophic forgetting, and Jung et al. (2020) in which two group sparsity-based penalties are utilized. The method introduced by Ebrahimi et al. (2020a) bases its regularization on the uncertainty in the probability distribution of the network weights. Another BNN-based CL algorithm has been proposed by Kurle et al. (2020). CL frameworks based on Gaussian processes have been proposed by Titsias et al. (2019), Kapoor et al. (2021), among others.

The regularization-based approach is orthogonal to the proposed framework, *STAF*. This means that ideas therein can be combined with the ideas in *STAF*.

The third main approach to CL is the architecture-based approach (Rusu et al., 2016b; Fernando et al., 2017; Kaplanis et al., 2018; Xu & Zhu, 2018; Yoon et al., 2018; Du et al., 2019; Gigante et al., 2019; He et al., 2019; Li et al., 2019a; Ostapenko et al., 2019; Xu et al., 2019; Gao et al., 2020; Deng et al., 2021; Qin et al., 2021; Mirzadeh et al., 2022; Morawiecki et al., 2022) which often balances stability and learnability via a rigid division of the architecture into shared and task-specific components.

A CL architecture is established in PathNet (Fernando et al., 2017) based on agents selected by a genetic algorithm. Neural Architecture Search (NAS) is adopted by Gao et al. (2020) where reinforcement learning (RL) techniques are utilized to search for the best neural architecture for each task. Another RL-based CL framework is the one introduced by Kaplanis et al. (2018) where catastrophic forgetting is mitigated via RL agents with a synaptic model inspired by neuroscience. Another NAS-based algorithm is proposed by Smith et al. (2022) where the idea of architecture search is also extended to regularize architecture forgetting. The CL algorithm presented by Xu and Zhu (2018) is also based on NAS. The optimal number of neurons and filters to add to each layer with each upcoming task is cast as a combinatorial optimization problem optimized by an RL policy. The optimization performed by Javed and White (2019) leads to a sparse representation which limits catastrophic forgetting. Neural structure learning and parameter estimation are separated over two optimization steps in the work by Li et al. (2019b) where NAS is used to optimize for the former step. The method developed by Ostapenko et al. (2019) is based on a dynamic network expansion optimized by a GAN. CL Algorithms based progressive networks have as well been proposed (Rusu et al., 2016a, 2016b). In order to learn a new task in the work by Rusu et al. (2016a), the whole network from the previous task is first copied then augmented. Even though this mitigates catastrophic forgetting, it can be very tricky in terms of scalability since the size of the architecture will grow prohibitively larger with a large number of tasks.

The two proposed *STAF* algorithms can be considered an adaptive development of the architecture-based approach. In addition to the shared and task-specific components, *STAF* enforces a similarity-based adaptation via establishing an adaptation layer. The proposed *STAF* framework is also related to the multi-task learning paradigm (Caruana, 1997; Heskes, 2000; Bakker & Heskes, 2003; Stickland & Murray, 2019).

The algorithms presented by Chaudhry et al. (2018), Kim et al. (2019) have explored metrics for the inability to adapt to new tasks, which they refer to as intransigence. Other works that explored metrics for particular branches of CL, e.g. continual reinforcement learning, include Powers et al. (2022), Hammoud et al. (2023). A federated CL algorithm has been proposed by Yoon et al. (2021) to address the scenario where each client learns on a sequence of tasks from a private local data stream. The method introduced by Yin et al. (2021) aims at mitigating catastrophic forgetting via neuron calibration. An analysis of how batch normalization fares with CL frameworks is performed by Pham et al. (2022), while Caccia et al. (2022) analyzed how representations shift at task boundaries.

Some CL algorithms, e.g. (Caccia et al., 2020b; Gupta et al., 2020a; Yap et al., 2020), are based on developing variations of MAML (Finn et al., 2017) which are suitable for CL. The work by Nguyen et al. (2020) proposes a new measure to evaluate transferability of CL representations. A derivation is developed by Knoblauch et al. (2020) which promotes the adoption of the memory-based approach compared to the regularization-based approach. A new dataset has been introduced by Lacoste et al. (2020). A few CL algorithms have focussed on the class-incremental setting (Hou et al., 2019; Wu et al., 2019; Yu et al., 2020; Zhang et al., 2020; Akyurek et al., 2021; Cha et al., 2021b; Liu et al., 2021; Shim et al., 2021; Zhu et al., 2021; Cossu et al., 2022; Koh et al., 2022; Goswami et al., 2023; Rymarczyk et al., 2023). Adversarial Shapley values are utilized by Shim et al. (2021) to score the stored data examples according to their ability to avoid catastrophic forgetting.

Previous works on task-free CL include (Aljundi et al., 2019a, 2019c; Rao et al., 2019; Caccia et al., 2020b; Lee et al., 2020; Jin et al., 2021; Pourcel et al., 2022; Wang et al., 2022a; Ye & Bors, 2022; Chrysakis & Moens, 2023; Ye & Bors, 2023). The method by Aljundi et al. (2019c) is based on constrained optimization where sample selection is formulated as a constraint reduction problem. A controlled sampling of memories is adopted for replay by Aljundi et al. (2019a). In the method introduced by Lee et al. (2020), task-free CL is cast as an online variational inference procedure of Dirichlet process mixture models of a set of neural experts where each expert is in charge of a subset of the data. Caccia et al. (2020b) introduced a Continual version of MAML. Jin et al. (2021) proposed a gradient-based memory editing algorithm which continually updates the stored examples via gradient updates in order to improve the utility of such examples over time. An associate memory task-free CL algorithm is introduced by Pourcel et al. (2022) where the data stream is modelled as locally distributed, partially overlapping representation clusters. A memory evolution algorithm is proposed by Wang et al. (2022a) in which the memory data distribution dynamically evolves by forcing the memorization process to become gradually harder. Task-free CL streams are simulated by Chrysakis and Moens (2023) from existing datasets via permuting any labeled dataset into a continuously non-stationary stream. The task-free CL algorithm in (Ye & Bors, 2023) aims to improve the compactness levels of the learned structures via a novelty-aware sample selection approach that increases the diversity among the selected memory samples.

Other adaptation-based CL algorithms have been introduced into the literature (Rosenfeld & Tsotsos, 2018; Ahn et al., 2019; Stickland & Murray, 2019; Caccia et al., 2020a; Lee et al., 2021; Pratama et al., 2021). Adaptable attention layers have been introduced for multi-task learning by Stickland and Murray (2019). Adaptive regularization has been adopted by Ahn et al. (2019) to develop an uncertainty-based CL algorithm. The work by Requeima et al. (2019) has presented an adaptive meta-learning algorithm referred to as CNAPS, which has as well been applied to CL. The proposed *STAF* is different from CNAPS in several aspects, e.g. the adaptation granted by the latter is based on linear modulation. In addition, the similarity-based adaptation provided by the proposed *STAF* directly depends on balancing the global and task-specific sets of parameters with a previous task, which helps mitigate catastrophic forgetting and is therefore a conceptually more inherent continual learning framework where the learner has to obtain knowledge on the fly. On the other hand, CNAPS is originally a meta-learning framework. Moreover, the proposed *STAF* does not have to deal with the overhead resulting from employing a permutation invariant architecture (and corresponding functions).

Other previous works on adaptive CL include the approach pursued by Rebuffi et al. (2017, 2018), which adapts domains with visual adapters, but it is feasible for relatively small numbers of tasks due to the huge computational demands. It also requires expert oversight for tuning the numerous parameters involved, and is prone to overfitting in the low-data regime. The work by Li et al. (2019b) bases its adaptation on neural architecture search, but the computational feasibility can be problematic since the search space may grow exponentially with respect to the number of tasks.

Other previous works which touched on adaptive CL include (Nguyen et al., 2019; Ramasesh et al., 2021; Doan et al., 2021; Andle et al., 2023; Lin et al., 2023). Veniat et al. (2021) used modules and leveraged a task-driven prior over the exponential search space to decide which modules to reuse. A dual memory along with a context transformation network have been developed by Pham et al. (2021) to model task-specific features. A knowledge base has been developed by Ke et al. (2020) to learn from a mixed sequence of similar and dissimilar tasks via hard attention. The work presented by Doan et al. (2021) analyzed catastrophic forgetting under the neural tangent kernel regime and utilized an overlap matrix to assess the similarity between a source and a target task. Other algorithms include the ORACLE algorithm by Yoon et al. (2020) which has established a learner representing the parameters of each task as a summation of task-shared and task-specific parameters.

Prototypical Networks. Prototypical networks (PNs) were originally developed by Snell et al. (2017) for few-shot classification. PNs were used therein to enable the classifier to generalize to new classes not seen during training, given a small set of examples from each new class. Due to the possible limitation in the size of available data, PNs are based on a rather straightforward inductive bias. The main essence is to compute a representative (prototype) of points belonging to the same class by learning a representation where such prototypes can be found. Therefore, a nonlinear mapping from the input space to a common representation space is learned via a neural network. Every test point is then assigned the class of their nearest prototypes.

3. Methodology (Task-Aware *STAF*)

Due to the growth of CL models in complexity and the increasing possibility of encountering a heterogeneous spectrum of tasks, it is essential to establish automatically adaptive CL frameworks. The proposed *STAF* framework consists of: i) the main CL architecture, and; ii) the auxiliary PN.

In the main architecture of *STAF*, the computation of the task-specific components depends on both the shared components and the adaptation mechanism. The auxiliary PN provides the building block for the adaptation mechanism.

3.1 The Main Continual Learning Architecture

A graphical model depicting the network architecture of *STAF* is displayed in Figure 1. Similar to the standard practice in continual multi-task architectures, the shared components (with parameters $\boldsymbol{\theta}_h$) constitute the bottom part. The task-specific part (including the top classification layer) is at the very top. Refer to the task-specific parameters of task t as $\boldsymbol{\theta}_f^t$. The adaptation layer (with parameters $\boldsymbol{\theta}_a^t$) lies in the middle between the shared and task-specific components.

Parameters of the adaptation layer $\boldsymbol{\theta}_a^t$ are initialized based on the relationship/similarity between the current task and the previously encountered tasks. The latter relationship is evaluated via the auxiliary PN whose parameters are ϕ_r.

We have a sequence of m datasets, $D_t = \{\boldsymbol{x}_t^n, \boldsymbol{y}_t^n\}_{n=1}^{N_t}$, where $t \in \{1, 2, \ldots, m\}$ is the task index and N_t is the size of the training dataset of task t. Data points are depicted by $\boldsymbol{x} \in \mathcal{X}$ and their labels are $\boldsymbol{y} \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and label spaces, respectively, and $\mathbf{X}$ and $\mathbf{Y}$ are the input and output variables. For classification[2], $\mathcal{Y}$ is a finite set, i.e. $\mathcal{Y} = \{1, 2, \ldots, L\}$. Denote by T a generic task index value. Overall, two predictions are performed for each data point during training. The first is the label prediction $\boldsymbol{y}_t | \boldsymbol{x}_t$, and the second is the auxiliary task index prediction $T | \boldsymbol{x}_t$.

2. Extension to label spaces like regression is straightforward.

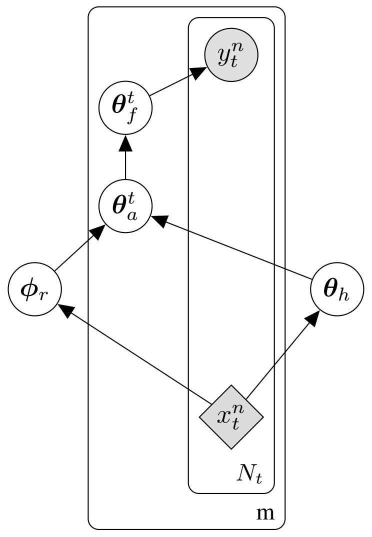


Figure 1: Architecture of the proposed *STAF*. The shared components (with parameters $\boldsymbol{\theta}_h$) are at the bottom. For a task t, the task-specific components (including the classification layer) are at the top (with parameters $\boldsymbol{\theta}_f^t$). The adaptation layer (with parameters $\boldsymbol{\theta}_a^t$) is in the middle between the shared and task-specific components. The adaptation parameters $\boldsymbol{\theta}_a^t$ are initialized based on the similarity between the current task t and the previously encountered tasks. The similarity is evaluated via the auxiliary PN whose parameters are ϕ_r.

Refer to the label prediction loss for task t as $L_f^t(\cdot,\cdot)$. For a data point $(\boldsymbol{x}_t^n, \boldsymbol{y}_t^n)$ from task t, the loss is defined as:

$$J(\boldsymbol{\theta}_h, \boldsymbol{\theta}_a^t, \boldsymbol{\theta}_f^t) = L_f^t(G_f^t(G_a^t(G_h(\boldsymbol{x}_t^n; \boldsymbol{\theta}_h); \boldsymbol{\theta}_a^t); \boldsymbol{\theta}_f^t), \boldsymbol{y}_t^n) \tag{1}$$

The nonlinear mappings performed by the shared, adaptation and task-specific parameters are referred to as G_h, G_a^t (feature extractors) and G_f^t (label predictor) respectively. More details about the auxiliary PN are given in Section 3.2, but let's focus here on its output: Index of the most similar task t^{NN} to the current task t. The adaptation parameters $\boldsymbol{\theta}_a^t$ are initialized with the final values of the adaptation parameters $\boldsymbol{\theta}_a^{\text{NN}}$ which were already learned by data from t^{NN}. One backpropagation gradient update to the task-specific, adaptation and shared parameters of task t, $\boldsymbol{\theta}_f^t$, $\boldsymbol{\theta}_a^t$ and $\boldsymbol{\theta}_h$, is expressed as:

$$\boldsymbol{\theta}_f^t \leftarrow \boldsymbol{\theta}_f^t - \alpha_f^t \frac{1}{N_t} \frac{\partial L_f^t}{\partial \boldsymbol{\theta}_f^t} \tag{2}$$

$$\boldsymbol{\theta}_a^t \leftarrow \boldsymbol{\theta}_a^{\text{NN}} - \alpha_a^t \frac{1}{N_t} \frac{\partial L_f^t}{\partial \boldsymbol{\theta}_a^t} \tag{3}$$

$$\boldsymbol{\theta}_h \leftarrow \boldsymbol{\theta}_h - \alpha_h \frac{1}{N_t} \frac{\partial L_f^t}{\partial \boldsymbol{\theta}_h} \tag{4}$$

The gradient steps displayed in equations (2)-(4) depict the first gradient step. This is why we display the initial condition of the parameters in the first term at the R.H.S., e.g. $\boldsymbol{\theta}_a^{\text{NN}}$.

Similar to the shared components, dimensions of the parameter vectors of the adaptation layer $\boldsymbol{\theta}_a^t$ are identical for all tasks, unlike the task-specific components. Apparently, the task-specific components are solely dedicated to learn from the respective task data and are therefore completely task-dependent. The separation between the adaptation and task-specific components is beneficial for both. On the one hand, the total number of adaptation parameters is not excessively large due to restricting the size of the adaptation components to one layer. This is useful to establish an effective adaptation-stability balance. Adapting an excessively large number of parameters can lead to a framework which is less time-efficient, less robust and more prone to catastrophic forgetting. On the other hand, this separation enables task-specific learning from the current data to operate without detractions.

The value of the adaptation learning rate α_a^t depends on the degree of similarity between the most similar task t^{NN} and the current task t. The learning rate has an impact on how far the adaptation parameters $\boldsymbol{\theta}_a^t$ can move away from their initial values $\boldsymbol{\theta}_a^{\text{NN}}$. A larger learning rate α_a^t permits the adaptation parameters to be more different from the initial values. Value of the adaptation learning rate α_a^t is inversely proportionate to the degree of similarity between the upcoming task t and the most similar task t^{NN}. Further details on setting the value of the adaptation learning rate α_a^t are provided in Section 3.2.

There are two other learning rates, α_f^t and α_h. Since the shared parameters are updated by all tasks, and in order to mitigate catastrophic forgetting, the value of the shared learning rate α_h is smaller than the task-specific learning rate α_f^t.

For a task t, the algorithmic complexity on the main *STAF* architecture is $O(N_t G B^2)$ where N_t is the size of the training dataset of task t, G is the number of layers in the network, and B is the (largest) number of neurons within a single layer.

3.2 The Auxiliary Prototypical Network

The PN is the auxiliary architecture whose output feeds the adaptation layer of the main architecture. The PN learns how to identify the most similar task t^{NN} to the current task t.

Compared to a standard PN, we develop a PN with two principal variations: 1) the manner via which the training and test procedures are performed; and 2) the dependent variable.

The dependent variable learned by our PN is the task index T, which here signifies the most similar task t^{NN}. After learning $t-1$ tasks, the learner is currently presented with data D_t from task t. For each task, the learning procedure of the PN goes through two phases to achieve two purposes. The first phase aims to decide which, out of the previous tasks $1, 2, \ldots, t-1$, is the most similar to the current task t. The second phase pre-emptively prepares the framework for the future tasks, via ensuring that the current task t will be involved when evaluating the most similar task to future tasks, $t+1, t+2,$ etc.

Refer to the sample presented to the PN from an arbitrary task T as $D_T = \{\boldsymbol{x}_T^n, T\}_{n=1}^{N_T}$, where $\boldsymbol{x}_T^n$ is the data, and the task index T is the dependent variable[3]. Refer to the mapping learned by the PN from input space $\mathcal{X}$ as $r = G_r(\boldsymbol{x}; \boldsymbol{\phi}_r)$ where $r \in \mathbb{R}^L$ and $\boldsymbol{\phi}_r$ are the mapping parameters. Each prototype $\boldsymbol{\mu}_T \in \mathbb{R}^L$ is an L-dimensional representation depicting the mean of data points of task T

3. Please note that, as far as the PN is concerned, the dependent variable is the task index T, not the label $\boldsymbol{y}_T^n$.

which are embedded by G_r:

$$\boldsymbol{\mu}_T = \frac{1}{N_T}\sum_{n=1}^{N_T} G_r(\boldsymbol{x}_T^n; \boldsymbol{\phi}_r) \tag{5}$$

When encountering a task $T = t$, the first goal is to evaluate the most similar task to t. The surrogate for this goal is the prediction of the task index of task t, importantly, *prior to training the PN on its data D_t*. In other words, the PN, which has only been trained on data from tasks $T = 1, 2, \ldots, t-1$ thus far, attempts to predict a task index for data from task t, D_t. This conceptually leads to the prediction of the most similar task to t (without needing to store, nor replay, any data points from the previous tasks) since the PN has not yet been trained on D_t.

More formally, denote by $d : \mathbb{R}^L \times \mathbb{R}^L \to [0, \infty)$ a distance metric. Given data D_t from task t, the most similar task t^{NN} is computed by:

$$t^{\text{NN}} = \underset{T\in\{1,\ldots,t-1\}}{\text{argmax}} \sum_{\boldsymbol{x}\in D_t} \frac{\exp(-d(G_r(\boldsymbol{x}; \boldsymbol{\phi}_r), \boldsymbol{\mu}_T))}{\sum_{T=1}^{t-1}\exp(-d(G_r(\boldsymbol{x}; \boldsymbol{\phi}_r), \boldsymbol{\mu}_T))} \tag{6}$$

After evaluating t^{NN}, the PN is trained on data D_t in the second phase (such that task t can eventually be included in the comparisons made to indicate most similar task to future tasks). For every point $\boldsymbol{x}$, the PN yields a distribution over all task indices via a softmax over distances to the prototypes $\boldsymbol{\mu}_t$, after applying the mapping G_r. The gradient-based PN learning minimizes the negative log-likelihood (LL) $E(\boldsymbol{\phi}_r) = -\log p_{\boldsymbol{\phi}_r}(T = t|\boldsymbol{x}_t)$, $\boldsymbol{x}_t \in D_t$, of the correct value of the task index ($T = t$). Define $p_{\boldsymbol{\phi}_r}(T = t|\boldsymbol{x}_t)$ as:

$$p_{\boldsymbol{\phi}_r}(T = t|\boldsymbol{x}_t) = \frac{\exp(-d(G_r(\boldsymbol{x}_t; \boldsymbol{\phi}_r), \boldsymbol{\mu}_t))}{\sum_{T=1}^{t}\exp(-d(G_r(\boldsymbol{x}_t; \boldsymbol{\phi}_r), \boldsymbol{\mu}_T))} \tag{7}$$

During the PN training, the mapping G_r attempts to embed the data D_t of task t into a space where it can be as distinguishable as possible from data belonging to tasks other than t. The main steps of the task-aware *STAF* are listed in Algorithm 1.

Regarding the adaptation learning rate α_a^t which was initially discussed in Section 3.1, it is inversely proportionate to the degree of similarity between the upcoming task t and the most similar task t^{NN}. We set the value of the adaptation learning rate α_a^t as follows:

$$\alpha_a^t = 1 - \frac{\exp(-d(G_r(\boldsymbol{x}_t; \boldsymbol{\phi}_r), \boldsymbol{\mu}_{\text{NN}}))}{\sum_{T=1}^{t-1}\exp(-d(G_r(\boldsymbol{x}_t; \boldsymbol{\phi}_r), \boldsymbol{\mu}_T))}, \quad \boldsymbol{x}_t \in D_t \tag{8}$$

4. Task-Free *STAF*

We propose a *STAF* algorithm for task-free CL where there are no observed task boundaries. The adaptive characteristics of *STAF* are preserved in its task-free version.

The learner encounters data which arrive sequentially. Refer to the consecutive pairs of data and labels arriving at time step k as a data stream $D_k = \{\boldsymbol{x}_k^n, \boldsymbol{y}_k^n\}_{n=1}^{N_k}$ with size N_k. Upon encountering data D_k at time step k, there is no given information on whether or not D_k belongs to the same distribution/context as the previous data stream D_{k-1}. Let K refer to an arbitrary index of a data stream. Denote by C the index of a context. A context involves one or more data streams.

Algorithm 1 Training of the task-aware *STAF* Algorithm

```
Input: A sequence of $m$ datasets: $D_t, t = 1, 2, \ldots, m$.
  Initialize the shared and task-specific parameters $\boldsymbol{\theta}_h$ and $\boldsymbol{\theta}_f^t$ for all tasks, and adaptation parameters $\boldsymbol{\theta}_a^1$ for task 1.
  for $t = 2, \ldots, m$ do
    // PN Learning
    Observe sample $D_t = \{\boldsymbol{x}_t^n, \boldsymbol{y}_t^n\}_{n=1}^{N_t}$
    Embed data $x_t$ as $G_r(\boldsymbol{x}_t; \boldsymbol{\phi}_r)$, where $\phi_r$ are the mapping parameters.
    Compute current prototype $\boldsymbol{\mu}_t$ using (5).
    Predict the most similar task $t^{\text{NN}}$ using (6).
    Train $D_t$ via negative LL. Compute prob. using (7).
    // Main CL architecture
    Initialize $\boldsymbol{\theta}_a^t$ via final values of $\boldsymbol{\theta}_a^{\text{NN}}$
    Train parameters $\boldsymbol{\theta}_f^t$, $\boldsymbol{\theta}_a^t$ and $\boldsymbol{\theta}_h$ using (2), (3) and (4).
  end for
```

With no observed context indices, the learner has to identify the boundaries which depict context shifts. We develop the following procedure via which the proposed *STAF* infers the boundary at which data streams begin to follow a different context from that of their precedents.

Refer to the (hidden) context of a data stream D as $C(D)$, and refer to the most similar previous context to D as $C_{\text{NN}}(D)$. Suppose the current data stream is D_k. The intuition of the procedure is that as long as the most similar context to the current data stream is the same as the most similar context of the last data stream, i.e. $C_{\text{NN}}(D_k) = C_{\text{NN}}(D_{k-1})$, this signifies that the current data most probably belongs to the same context as the most recent data, i.e. $C(D_k) = C(D_{k-1})$.

The dependent variable learned by the PN here is the context index C. Upon encountering D_k, the PN first infers the most similar context to D_k, $C_{\text{NN}}(D_k)$. At this point, suppose there are i contexts between which the PN can distinguish, $i \leq k - 1$[4]. The most similar context $C_{\text{NN}}(D_k)$ is computed as follows:

$$C_{\text{NN}}(D_k) = \underset{C \in \{1,\ldots,i\}}{\operatorname{argmax}} \sum_{\boldsymbol{x} \in D_k} \frac{\exp(-d(G_r(\boldsymbol{x}; \boldsymbol{\phi}_r), \boldsymbol{\mu}_C))}{\sum_{C=1}^{i} \exp(-d(G_r(\boldsymbol{x}; \boldsymbol{\phi}_r), \boldsymbol{\mu}_C))} \tag{9}$$

Context prototypes are denoted by $\boldsymbol{\mu}_C$. In case the most similar context to the current data stream is the same as that of the last data stream, $C_{\text{NN}}(D_k) = C_{\text{NN}}(D_{k-1})$, then D_k is considered a continuation of data from the context of the previous stream, $C(D_k) = C(D_{k-1})$. In such case:

i The current data stream D_k is trained as part of the context of the last data stream $C_i = C(D_k) = C(D_{k-1})$. Due to the inclusion of data D_k, the prototype $\boldsymbol{\mu}_{\text{i}}$ of context C_i must be updated. After accumulating $|D_k|$, refer to the updated size of data of C_i as N_i. For $\boldsymbol{x}_i \in C_i$:

$$\boldsymbol{\mu}_i = \frac{1}{N_i} \sum_{n=1}^{N_i} G_r(\boldsymbol{x}_i^n; \boldsymbol{\phi}_r) \tag{10}$$

4. Each context comprises one or more data streams. Prior to D_k, the learner has already encountered $k - 1$ data streams. Therefore, the current number of contexts is $i \leq k - 1$.

ii Data of the current stream $\boldsymbol{x}_k \in D_k$ is then trained via minimizing the negative LL: $E(\boldsymbol{\phi}_r) = -\log p_{\boldsymbol{\phi}_r}(C = C_i|\boldsymbol{x}_k)$:

$$p_{\boldsymbol{\phi}_r}(C = C_i|\boldsymbol{x}_k) = \frac{\exp(-d(G_r(\boldsymbol{x}_k;\boldsymbol{\phi}_r),\boldsymbol{\mu}_i))}{\sum_{C=1}^{i}\exp(-d(G_r(\boldsymbol{x}_k;\boldsymbol{\phi}_r),\boldsymbol{\mu}_C))} \tag{11}$$

iii Since the current data D_k does not belong to a new context, there is no need to re-train the main CL architecture[5].

Otherwise, in case the most similar context to the current data D_k is different from the last stream, i.e. $C_{\text{NN}}(D_k) \neq C_{\text{NN}}(D_{k-1})$, then D_k belongs to a new context C_{i+1}. In such case:

i The current data stream D_k is the first data stream of the new context C_{i+1}. For $\boldsymbol{x}_k \in D_k$:

$$\boldsymbol{\mu}_{i+1} = \frac{1}{|D_k|}\sum_{n=1}^{|D_k|} G_r(\boldsymbol{x}_k^n;\boldsymbol{\phi}_r) \tag{12}$$

ii The PN training proceeds by minimizing the negative LL of the new context C_{i+1}: $E(\boldsymbol{\phi}_r) = -\log p_{\boldsymbol{\phi}_r}(C = C_{i+1}|\boldsymbol{x}_k)$.

$$p_{\boldsymbol{\phi}_r}(C = C_{i+1}|\boldsymbol{x}_k) = \frac{\exp(-d(G_r(\boldsymbol{x}_k;\boldsymbol{\phi}_r),\boldsymbol{\mu}_{\text{i+1}}))}{\sum_{C=1}^{i+1}\exp(-d(G_r(\boldsymbol{x}_k;\boldsymbol{\phi}_r),\boldsymbol{\mu}_C))} \tag{13}$$

iii Training of the new context C_{i+1} on the main CL architecture is then performed in a similar manner to the task-aware setting: The adaptation parameters of the new context C_{i+1} are initialized with the values of its most similar context $C_{\text{NN}}(D_k)$.

Regarding the second context C_2, which can begin at any data stream from the second data stream, $k \geq 2$. This is a special case since there is only one previous stream C_1. To predict whether a data stream D_k belongs to C_1: Rather than using the most similar context in (9) to predict the boundary between the first and second contexts, a threshold is applied over the following expression: $\exp(-d(G_r(\boldsymbol{x}_k;\boldsymbol{\phi}_r),\boldsymbol{\mu}_1))$. If the value of this expression is smaller than the threshold for data points $\boldsymbol{x}_k$ of stream C_k, then such points are far enough from the prototype $\boldsymbol{\mu}_1$ of C_1, and therefore C_k is the first stream of a new context C_2. Otherwise, the current data points D_k are added to the first stream C_1 based on (10) and (11).

The key steps of the task-free *STAF* are listed in Algorithm 2.

5. Theoretical Analysis

Let's first set up the notation. A task $\mathcal{T}$ is characterized by a distribution $\mathcal{D}_{\mathcal{T}}$ on the input space $\mathcal{X}$ and a labeling function $f : \mathcal{X} \to [0,1]$. Note that the output of the labeling function f can be probabilistic. A hypothesis space $\mathcal{H}$ is a space of functions $h : \mathcal{X} \to [0,1]$. For analysis, we mainly focus on binary classification problems, but the extension to multi-class classification is straightforward. Under a distribution $\mathcal{D}_{\mathcal{T}}$, the error of a hypothesis h w.r.t. a labeling function f is:

5. Note that the two previous steps were performed on the PN.

Algorithm 2 The task-free *STAF* Algorithm

```
Input: A sequence of data streams $D_k$. Each data stream $D_k$ belongs to a (hidden) context $C_i$.
  Initialize the shared and context-specific parameters $\boldsymbol{\theta}_h$ and $\boldsymbol{\theta}_f^k$, and adaptation parameters $\boldsymbol{\theta}_a^1$ for context 1.
  for all data streams $D_k$, $k = 2, \dots$ do
    // PN Learning
    Observe sample $D_k = \{\boldsymbol{x}_k^n, \boldsymbol{y}_k^n\}_{n=1}^{N_k}$
    Predict the most similar context $C_{\text{NN}}(D_k)$ via (9).
    if $C_{\text{NN}}(D_k) = C_{\text{NN}}(D_{k-1})$ then
      Let $C_i = C_{\text{NN}}(D_k) = C_{\text{NN}}(D_{k-1})$
      Update $\boldsymbol{\mu}_i$ via (10).
      Train $D_k$ as part of $C_i$. Compute prob. using (11).
    else
      Compute $\boldsymbol{\mu}_{i+1}$ using (12).    //a new context $C_{i+1}$
      Train $D_k$ as (the first) part of $C_{i+1}$. Compute prob. using (13).
      // Main CL architecture
      Initialize $\boldsymbol{\theta}_a^k$ via final values of $\boldsymbol{\theta}_a^{\text{NN}}$
      Train parameters $\boldsymbol{\theta}_f^k$, $\boldsymbol{\theta}_a^k$ and $\boldsymbol{\theta}_h$ using (2), (3) and (4).
    end if
  end for
```

$$\varepsilon_{\mathcal{T}}(h, f) := \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}}}[|h(\mathbf{x}) - f(\mathbf{x})|] \tag{14}$$

We address a continual learning setup with sequentially arriving tasks characterized by clear boundaries. The learner has already encountered m training tasks $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_m\}$ where each task is drawn from a distribution of tasks $\Pr(\mathcal{T})$.

Define the risk of hypothesis h as the error of h w.r.t. the true labeling function $f_{\mathcal{T}}$ under task $\mathcal{T}$: $\varepsilon_{\mathcal{T}}(h) := \varepsilon_{\mathcal{T}}(h, f_{\mathcal{T}})$. As a measure of distance between two distributions, we build on the $\mathcal{H}$-divergence (Ben-David et al., 2010), which is a commonly used distance measure in domain adaptation (Sun et al., 2011; Ajakan et al., 2014; Zhao et al., 2019). Let $\mathcal{H}$ be a hypothesis space defined on $\mathcal{X}$, and $\mathcal{A}_{\mathcal{H}}$ be the collection of subsets of $\mathcal{X}$ representing the support of a hypothesis h in $\mathcal{H}$, i.e. $\mathcal{A}_{\mathcal{H}} := \{h^{-1}(1) \mid h \in \mathcal{H}\}$. We stick to the way the definition of the $\mathcal{H}$-divergence was described by Zhao et al. (2019) which does not include the factor of 2 since, in our work as well, the constant factor is not relevant. Based on $\mathcal{H}$, the distance between two distributions $\mathcal{D}$ and $\mathcal{D}'$ is: $d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}') := \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}}(A) - \Pr_{\mathcal{D}'}(A)|$.

For the sake of lucidity, we follow a rather modular approach in this section. We begin by introducing the theory and its gist in Section 5.1, followed by the proofs and related information in Section 5.2.

5.1 Generalization Upper Bound

The primary motivation of the CL bound derived here is to provide a theoretical corroboration for the proposed *STAF* algorithm based on the bound's degrees of freedom, not to come up with the tightest CL bound. Defining the difference between two tasks as the summation of the distance between their

marginal (data) distributions and the distance between their labeling functions (conditional of the label given data), Theorem 1 states that the error on an upcoming (test) task $\mathcal{T}_i$ is less than or equal to the summation of the following terms:
i) summation of the differences between each pair of training (i.e. already encountered) tasks, multiplied by a factor of two,
ii) the minimum upper bound on the test task $\mathcal{T}_i$ with an individual training task, i.e. the minimum upper bound on $\mathcal{T}_i$ had there been only one training task.

Theorem 1. *Let $\mathcal{T}_i$ refer to a test task with index i, and $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m\}$ refer to the training tasks. The following holds for a hypothesis space $\mathcal{H}$ and any hypothesis $h \in \mathcal{H}$:*

$$\varepsilon_{\mathcal{T}_i}(h) \leq$$

$$\sum_{j,k=1,k>j}^{m} \Big(2min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) + \tag{15}$$

$$\min_{j\in\{1,2,..,m\}} \Big\{ \Big(\varepsilon_{\mathcal{T}_j}(h) + min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\} \tag{16}$$

The bound consists of two kinds of terms, the first in (15) depicts the summation of the differences between pairs of training tasks (marginals + labeling functions), multiplied by two. It is important to note that this kind of term does not include a learning hypothesis h. In other words, this fully depends on the training tasks already encountered by the learner, and is therefore fixed once the training tasks and their respective training data have been observed. There are no degrees of freedom involved herein based on which the continual learner can capture intuition for a CL optimization procedure.

The second kind of term in the bound is the minimum term in (16). It consists of m clauses (one per training task). As derived in (27) in Section 5.2, each clause in this minimum term depicts the upper bound on test task $\mathcal{T}_i$ had the learner encountered solely a single training task. Thus, given m training tasks, Theorem 1 notes that finding the least distant (i.e. the most similar) training task to the current test task $\mathcal{T}_i$ tightens the bound and correspondingly minimizes the error on $\mathcal{T}_i$.

A large number of training tasks m might add up to (15). However, regarding the learnability of the model, this can actually be beneficial, since a larger pool of training tasks can potentially be useful in minimizing the latter term in (16).

Thus, Theorem 1 indicates a dependence between the error of an upcoming task and the most similar previous task.

5.2 Proofs

We begin by proving a few lemmata which we will eventually need to prove Theorem 1.

Lemma 1. *The triangle inequality is satisfied by the $\mathcal{H}$-divergence distance measure. More formally, the $\mathcal{H}$-divergence satisfies the following:*

$$d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}') \leq d_{\mathcal{H}}(\mathcal{D}, \mathcal{D}'') + d_{\mathcal{H}}(\mathcal{D}'', \mathcal{D}') \tag{17}$$

for any hypothesis space $\mathcal{H}$ and distributions $\mathcal{D}$, $\mathcal{D}'$ and $\mathcal{D}''$ over the same space.

Proof.

$$\begin{aligned}
d_{\mathcal{H}}(\mathcal{D},\mathcal{D}') &= \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}}(A) - \Pr_{\mathcal{D}'}(A)| \\
&= \sup_{A\in\mathcal{A}_{\mathcal{H}}} \Big(|\Pr_{\mathcal{D}}(A) - \Pr_{\mathcal{D}''}(A) + \Pr_{\mathcal{D}''}(A) - \Pr_{\mathcal{D}'}(A)|\Big) \\
&\leq \sup_{A\in\mathcal{A}_{\mathcal{H}}} \Big(|\Pr_{\mathcal{D}}(A) - \Pr_{\mathcal{D}''}(A)| + |\Pr_{\mathcal{D}''}(A) - \Pr_{\mathcal{D}'}(A)|\Big) \\
&= \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}}(A) - \Pr_{\mathcal{D}''}(A)| + \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}''}(A) - \Pr_{\mathcal{D}'}(A)| \\
&= d_{\mathcal{H}}(\mathcal{D},\mathcal{D}'') + d_{\mathcal{H}}(\mathcal{D}'',\mathcal{D}')
\end{aligned} \tag{18}$$

□

Lemma 2. *For a task $\mathcal{T}$ with a marginal distribution $\mathcal{D}_{\mathcal{T}}$, and for a hypothesis space $\mathcal{H}$, the following triangle inequality is satisfied by the labeling function:*

$$\varepsilon_{\mathcal{T}}(h,h') \leq \varepsilon_{\mathcal{T}}(h,h'') + \varepsilon_{\mathcal{T}}(h'',h') \tag{19}$$

for any three hypotheses h, h', $h'' \in \mathcal{H}$.

Proof.

$$\begin{aligned}
\varepsilon_{\mathcal{T}}(h,h') &= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}},h,h'\sim U}\Big[|h(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}},U}\Big[|h(\mathbf{x}) - h''(\mathbf{x}) + h''(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&\leq \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}},U}\Big[|h(\mathbf{x}) - h''(\mathbf{x})| + |h''(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&= \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}},U}\Big[|h(\mathbf{x}) - h''(\mathbf{x})|\Big] + \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}},U}\Big[|h''(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&= \varepsilon_{\mathcal{T}}(h,h'') + \varepsilon_{\mathcal{T}}(h'',h')
\end{aligned} \tag{20}$$

□

Lemma 3. *Let $\mathcal{D}_{\mathcal{T}_j}$ and $\mathcal{D}_{\mathcal{T}_k}$ be the distributions for tasks $\mathcal{T}_j$ and $\mathcal{T}_k$, respectively. For a hypothesis class $\mathcal{H}$, $\forall h,h' \in \mathcal{H}$:*

$$\varepsilon_{\mathcal{T}_j}(h,h') \leq \varepsilon_{\mathcal{T}_k}(h,h') + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j},\mathcal{D}_{\mathcal{T}_k}) \tag{21}$$

Proof. Given $h,h' \in \mathcal{H}$.

$$\begin{aligned}
&\varepsilon_{\mathcal{T}_j}(h,h') = \mathbb{E}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{T}_j},h,h'\sim U}\Big[|h(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&\quad= \mathbb{E}_{h,h'\sim U}\int_{\mathbf{x}} \Pr_{\mathcal{D}_{\mathcal{T}_j}}(\mathbf{x})\Big[|h(\mathbf{x}) - h'(\mathbf{x})|\Big] = \mathbb{E}_{h,h'\sim U}\int_{\mathbf{x}} \Big(\Pr_{\mathcal{D}_{\mathcal{T}_j}}(\mathbf{x}) - \Pr_{\mathcal{D}_{\mathcal{T}_k}}(\mathbf{x}) + \Pr_{\mathcal{D}_{\mathcal{T}_k}}(\mathbf{x})\Big)\Big[|h(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&\quad= \varepsilon_{\mathcal{T}_k}(h,h') + \mathbb{E}_{h,h'\sim U}\int_{\mathbf{x}} \Big(\Pr_{\mathcal{D}_{\mathcal{T}_j}}(\mathbf{x}) - \Pr_{\mathcal{D}_{\mathcal{T}_k}}(\mathbf{x})\Big)\Big[|h(\mathbf{x}) - h'(\mathbf{x})|\Big] \\
&\quad\leq \varepsilon_{\mathcal{T}_k}(h,h') + \sup_{A\in\mathcal{A}_{\mathcal{H}}} |\Pr_{\mathcal{D}_{\mathcal{T}_j}}(A) - \Pr_{\mathcal{D}_{\mathcal{T}_k}}(A)| \\
&\quad= \varepsilon_{\mathcal{T}_k}(h,h') + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j},\mathcal{D}_{\mathcal{T}_k})
\end{aligned} \tag{22}$$

□

Theorem 1. *Let $\mathcal{T}_i$ refer to an upcoming (test) task with index i, and $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m\}$ refer to the training tasks (i.e. the tasks already encountered by the continual learner). The following holds for a hypothesis space $\mathcal{H}$ and any hypothesis $h \in \mathcal{H}$:*

$$\varepsilon_{\mathcal{T}_i}(h) \leq \sum_{j,k=1,k>j}^{m} \Big(2min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) + \tag{23}$$

$$\min_{j\in\{1,2,\ldots,m\}} \Big\{\Big(\varepsilon_{\mathcal{T}_j}(h) + min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\} \tag{24}$$

Proof. We will formalize the proof using mathematical induction. First, we show that the form holds for one and then for two training tasks[6]. Afterwards, we pursue the induction step.

By applying Lemma 3 to $\varepsilon_{\mathcal{T}_i}(h)$, followed by Lemma 2, we get:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) &= \varepsilon_{\mathcal{T}_i}(h, f_{\mathcal{T}_i}) \\ &\leq \varepsilon_{\mathcal{T}_1}(h, f_{\mathcal{T}_i}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i}) \\ &\leq \varepsilon_{\mathcal{T}_1}(h) + \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\end{aligned} \tag{25}$$

Recall that $\varepsilon_{\mathcal{T}_1}(h, f_{\mathcal{T}_1}) = \varepsilon_{\mathcal{T}_1}(h)$, which we use in (25) above. Meanwhile, by reversing the order, i.e. by applying Lemma 2 first, followed by Lemma 3, we get the following:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) &= \varepsilon_{\mathcal{T}_i}(h, f_{\mathcal{T}_i}) \\ &\leq \varepsilon_{\mathcal{T}_i}(h, f_{\mathcal{T}_1}) + \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}) \\ &\leq \varepsilon_{\mathcal{T}_1}(h) + \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\end{aligned} \tag{26}$$

From (25) and (26):

$$\varepsilon_{\mathcal{T}_i}(h) \leq \varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i}), \tag{27}$$

which forms an upper bound on the error of the test task $\mathcal{T}_i$ given one training task $\mathcal{T}_1$.

Let's now involve another training task $\mathcal{T}_2$. By applying lemmata 2 and 3 in both orders, similar to the operations in (25), (26) and (27), $\varepsilon_{\mathcal{T}_1}(h)$ can be expressed as follows:

$$\varepsilon_{\mathcal{T}_1}(h) \leq \varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1}), \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_1}) \tag{28}$$

From (28) back into (27):

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq& \varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1}), \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1})\} + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} \\ &+ d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_1}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\end{aligned} \tag{29}$$

By the moment we do this analysis, the continual learner has already encountered all the m training tasks, i.e. all the tasks in the following set: $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m\}$. Therefore, we can arbitrarily begin our analysis with any task belonging to this set, e.g. task $\mathcal{T}_2$ rather than task $\mathcal{T}_1$ (as performed above). As such, we are not bound in this analysis by the order via which the continual learner has

6. To further make sure that the form is clear, we begin with two, rather than solely one, base cases.

encountered the training tasks. Above, we began by $\mathcal{T}_1$ in (27) followed by $\mathcal{T}_2$ in (28). Let's switch the order so that we begin with $\mathcal{T}_2$ followed by $\mathcal{T}_1$. The outcome in such case would be as follows:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq & \varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1}), \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1})\} + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} \\ & + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_1}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i}) \end{aligned} \tag{30}$$

From (29) and (30):

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq\ & min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1}), \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_1}) \\ & + min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \qquad \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{31}$$

Note that, similar to what was carried out above to obtain (28) for $\mathcal{T}_1$, we can as well apply lemmata 2 and 3 in both orders to obtain a similar bound for $\mathcal{T}_2$ by reversing the roles of $\mathcal{T}_1$ and $\mathcal{T}_2$. This leads to the following for $\mathcal{T}_2$:

$$\varepsilon_{\mathcal{T}_2}(h) \leq \varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2}), \varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_2}) \tag{32}$$

Using the inequalities in (28) and (32) to express $\varepsilon_{\mathcal{T}_1}(h)$ and $\varepsilon_{\mathcal{T}_2}(h)$, respectively, in (31), we get the upper bound on the error of an upcoming task $\mathcal{T}_i$, $\varepsilon_{\mathcal{T}_i}(h)$, given two training tasks $\mathcal{T}_1$ and $\mathcal{T}_2$ as:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq\ & 2\, min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1}), \varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_2}, f_{\mathcal{T}_1})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_1}) \\ & + min\Big\{\Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \qquad \Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{33}$$

By simply rearranging the clauses of the third (minimum) term on the R.H.S., we get:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq\ & 2\, min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2}), \varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_2}) \\ & + min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \qquad \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{34}$$

In (27) we have an upper bound on the error $\varepsilon_{\mathcal{T}_i}(h)$ for the case when there is only one training task involved. This one task happened to be $\mathcal{T}_1$ in (27), but had there been another training task involved rather than $\mathcal{T}_1$, as the sole training task, e.g. $\mathcal{T}_2$, the same bound in (27) could then be applied via replacing $\mathcal{T}_1$ with such a task (e.g. $\mathcal{T}_2$) in (27).

Using this fact for both clauses of the third (minimum) term on the R.H.S. of (34), we get the following upper bound on the error given two training tasks $\mathcal{T}_1$ and $\mathcal{T}_2$:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq\ & 2\, min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2}), \varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_1}, f_{\mathcal{T}_2})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_2}) \\ & + min\Big\{\text{bound with one training task } \mathcal{T}_1, \\ & \qquad \text{bound with one training task } \mathcal{T}_2\Big\}\end{aligned} \tag{35}$$

Let's now move on to the induction step. Assume that the form of the generalization bound on the error of a test task $\mathcal{T}_i$, given two training tasks $\mathcal{T}_1$ and $\mathcal{T}_2$ in (34) equivalently holds for $m-1$ training tasks, i.e. assume that the following holds:

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) \\ & + \min_{j\in\{1,2,\ldots,m-1\}} \Big\{\Big(\varepsilon_{\mathcal{T}_j}(h) + min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{36}$$

Let's involve $\mathcal{T}_m$. Similar to the inequality form in (27), the following is an upper bound on the error of task $\mathcal{T}_{m-1}$:

$$\varepsilon_{\mathcal{T}_{m-1}}(h) \leq \varepsilon_{\mathcal{T}_m}(h) + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m}) \tag{37}$$

From (37) back into (36):

$$\begin{aligned}\varepsilon_{\mathcal{T}_i}(h) \leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) \\ & + \min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \ldots, \\ & \Big(\varepsilon_{\mathcal{T}_{m-2}}(h) + min\{\varepsilon_{\mathcal{T}_{m-2}}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-2}}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \Big(\varepsilon_{\mathcal{T}_m}(h) + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i})\} \\ & + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{38}$$

$$\begin{aligned}\leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) \\ & + \min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \ldots, \\ & \Big(\varepsilon_{\mathcal{T}_{m-2}}(h) + min\{\varepsilon_{\mathcal{T}_{m-2}}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-2}}, \mathcal{D}_{\mathcal{T}_i})\Big), \\ & \Big(\varepsilon_{\mathcal{T}_{m-1}}(h) + 2 min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i})\} \\ & + 2 d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}\end{aligned} \tag{39}$$

The switch from (38) to (39) is due to the inequality in (27). Recall again that the order of the tasks in this analysis is arbitrary and is independent from the order via which the continual learner has encountered the tasks. Assuming that $\mathcal{T}_m$ was involved in the analysis prior to the involvement of $\mathcal{T}_{m-1}$, i.e. the order of these two tasks has been switched, the outcome in such case would be as follows:

$$\begin{aligned}
\varepsilon_{\mathcal{T}_i}(h) \leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) \\
& + \min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \\
& \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \ldots, \\
& \Big(\varepsilon_{\mathcal{T}_{m-2}}(h) + min\{\varepsilon_{\mathcal{T}_{m-2}}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-2}}, \mathcal{D}_{\mathcal{T}_i})\Big), \\
& \Big(\varepsilon_{\mathcal{T}_{m-1}}(h) + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + min\{\varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i})\} \\
& + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_m}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\} \qquad (40) \\
\leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) \\
& + \min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \\
& \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \ldots, \\
& \Big(\varepsilon_{\mathcal{T}_{m-2}}(h) + min\{\varepsilon_{\mathcal{T}_{m-2}}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-2}}, \mathcal{D}_{\mathcal{T}_i})\Big), \\
& \Big(\varepsilon_{\mathcal{T}_m}(h) + 2min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + min\{\varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i})\} \\
& + 2d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m}) + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_m}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\} \qquad (41)
\end{aligned}$$

Taking the common terms out from (39) and from (41), we get the following:

$$\begin{aligned}
\varepsilon_{\mathcal{T}_i}(h) \leq & \sum_{j,k=1,k>j}^{m-1} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) + \qquad (42) \\
& \Big(2\, min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m}), \varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_m})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_m})\Big) + \qquad (43) \\
& \min\Big\{\Big(\varepsilon_{\mathcal{T}_1}(h) + min\{\varepsilon_{\mathcal{T}_1}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_1}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_1}, \mathcal{D}_{\mathcal{T}_i})\Big), \qquad (44) \\
& \Big(\varepsilon_{\mathcal{T}_2}(h) + min\{\varepsilon_{\mathcal{T}_2}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_2}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_2}, \mathcal{D}_{\mathcal{T}_i})\Big), \ldots, \qquad (45) \\
& \Big(\varepsilon_{\mathcal{T}_{m-2}}(h) + min\{\varepsilon_{\mathcal{T}_{m-2}}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-2}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-2}}, \mathcal{D}_{\mathcal{T}_i})\Big), \qquad (46) \\
& \Big(\varepsilon_{\mathcal{T}_{m-1}}(h) + min\{\varepsilon_{\mathcal{T}_{m-1}}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_{m-1}}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_{m-1}}, \mathcal{D}_{\mathcal{T}_i})\Big), \qquad (47) \\
& \Big(\varepsilon_{\mathcal{T}_m}(h) + min\{\varepsilon_{\mathcal{T}_m}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_m}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_m}, \mathcal{D}_{\mathcal{T}_i})\Big)\Big\}, \qquad (48)
\end{aligned}$$

which concludes the induction step and thus leads to the upper bound on the error of a test task $\mathcal{T}_i$ given m training tasks $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m\}$:

$$\varepsilon_{\mathcal{T}_i}(h) \leq \sum_{j,k=1,k>j}^{m} \Big(2\, min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k}), \varepsilon_{\mathcal{T}_k}(f_{\mathcal{T}_j}, f_{\mathcal{T}_k})\} + 2\, d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_k})\Big) + \tag{49}$$

$$\min_{j\in\{1,2,\ldots,m\}} \Big\{ \Big(\varepsilon_{\mathcal{T}_j}(h) + min\{\varepsilon_{\mathcal{T}_j}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i}), \varepsilon_{\mathcal{T}_i}(f_{\mathcal{T}_j}, f_{\mathcal{T}_i})\} + d_{\mathcal{H}}(\mathcal{D}_{\mathcal{T}_j}, \mathcal{D}_{\mathcal{T}_i})\Big) \Big\} \tag{50}$$

□

6. Experiments

In this section, we empirically evaluate the performance of the proposed *STAF*. We chiefly aim at evaluating the following aspects: i) the task-aware CL performance of *STAF*, measured by the average classification accuracy over all tasks encountered by the learner; ii) the degree to which catastrophic forgetting can be reduced with *STAF*; iii) the task-free CL performance of the corresponding *STAF* version; and iv) an ablation study gauging the impact of the adaptation layer on the obtained results. State-of-the-art results obtained in both the task-aware and task-free settings demonstrate the efficacy of the proposed *STAF* and its adaptation mechanism.

We perform CL experiments on six datasets: Split MNIST (Goodfellow et al., 2014a; Zenke et al., 2017), Fashion-MNIST (Xiao et al., 2017), Omniglot (Lake et al., 2011), Split CIFAR-10 (Krizhevsky & Hinton, 2009), Split CIFAR-100 (Krizhevsky & Hinton, 2009; Rebuffi et al., 2017) and Split mini-ImageNet (Deng et al., 2009; Vinyals et al., 2016; Chaudhry et al., 2019b).

We compare with the most powerful variation of every competing algorithm. The reported results are averages of 15 runs. Statistical Significance (highlighted in bold) is identified using a paired t-test with $p = 0.05$. Adam (Kingma & Ba, 2015) is the optimizer used with a learning rate decayed by a factor of 3 if there is no improvement in the validation loss for 5 consecutive epochs, similar to the works by Serra et al. (2018), Joseph and Balasubramanian (2020), Jung et al. (2020).

6.1 Task-Aware Classification

We evaluate how *STAF* fares compared to multiple state-of-the-art CL algorithms: (VCL, Nguyen et al., 2018), progress and compress (P&C, Schwarz et al., 2018), (RPS-Net, Rajasegaran et al., 2019), (CCLL, Singh et al., 2020), (ORTHOG-SUBSPACE, Chaudhry et al., 2020), CL with hypernetworks (HNET, von Oswald et al., 2020), (AGS-CL, Jung et al., 2020), (La-MAML, Gupta et al., 2020a), (MERLIN, Joseph & Balasubramanian, 2020), (CBRS, Chrysakis & Moens, 2020), (Gating, Abati et al., 2020), (GCL, Tang & Matteson, 2021), (CRNet, Li & Zeng, 2023), (NNA, Madireddy et al., 2023) and (RN18, Lee et al., 2023). The average classification accuracy is measured over (test samples from) all the encountered tasks, not only the latest task. This is the main performance metric of CL algorithms. All methods are trained with a minibatch size of 256 and for 100 epochs. We adopt a 60/20/20$\%$ allocation for training, validation and test, respectively. Initial values assigned to the shared learning rate α_h and the task-specific learning rate α_f^t are 0.02 and 0.05, respectively.

We provide a brief description of the six datasets in use.

Split MNIST The handwritten digit MNIST dataset (LeCun et al., 1998) is split into 5 disjoint subsets with non-overlapping classes to form the 5 tasks of Split MNIST.

Fashion-MNIST A dataset of 10 classes which is split into 5 binary classification tasks: Pullover/Dress, Shirt/Sneaker, T-shirt/Trouser, Bag/Ankle boots and Coat/Sandals.

Omniglot It consists of more than $1,600$ handwritten characters from 50 alphabets (20 examples per character). There are 50 tasks (one task per alphabet). It comprises a considerably larger number of tasks than the previous 2 datasets.

Split CIFAR-10 It is split into 5 different tasks with 2 non-overlapping classes per task. It contains $50,000$ examples.

Split CIFAR-100 Similar to (Lopez-Paz & Ranzato, 2017), CIFAR-100 is split into 20 tasks with 5 disjoint classes per task. CIFAR-100 consists of $50,000$ training examples.

Split mini-ImageNet It is based on a subset of ImageNet (Russakovsky et al., 2015) with a total of 100 classes and 600 images per class. Similar to (Aljundi et al., 2019a; Chaudhry et al., 2019b), it is split into 20 tasks with 5 classes each.

Like (Chrysakis & Moens, 2020), and to compare on common ground, choices of the model architecture in every experiment heavily depend on previous works. A 2-layer perceptron (MLP) with 250 neurons per layer and ReLU activations is used for Split MNIST and Fashion-MNIST. A ResNet-18 is utilized for the other 4 datasets.

Results of the average classification accuracy are displayed in Table 1. The proposed *STAF* achieves significantly higher classification results than all the previous state-of-the-art algorithms in all the 6 experiments. The relevance of the similarity-based adaptation adopted by *STAF* is more significant with the latter (and larger) 4 datasets, which also shows that *STAF* is scalable.

Due to the smooth nature of computations performed in *STAF*, it is also efficient in terms of wall-clock run-time[7]. On average, both *STAF* and La-MAML reach the accuracy levels reported in Table 1 more rapidly than the other methods.

6.2 Task-Free Classification

Based on the six benchmarks listed above, we evaluate the task-free *STAF* and how it addresses problems where task boundaries are not observed. No information about task boundaries is used in the learning procedure of any method under this setting. A shared head is used in the task-free *STAF*. We compare to several task-free CL algorithms: (GSS, Aljundi et al., 2019c), (ER-MIR, Aljundi et al., 2019a), (GMED, Jin et al., 2020), (ASER, Shim et al., 2020), (OSAKA, Caccia et al., 2020b), (CN-DPM, Lee et al., 2020), (WGF, Wang et al., 2022a), (ODDL, Ye & Bors, 2022) and (SEDEM, Ye & Bors, 2023). Similar to the bulk of previous works on task-free CL, e.g. (Aljundi et al., 2019c; Jin et al., 2020; Lee et al., 2020; Shim et al., 2020), a minibatch size of 10 is used in all the experiments on task-free CL. Architectures used in every experiment are similar to previous methods, to compare on common ground. For Split MNIST, a 2-hidden-layer MLP classifier with ReLU activation is used. The dimension of each layer is 400. For Fashion-MNIST and Omniglot, we use a 4-layer CNN with 64 hidden units. A ResNet-18 is used for Split CIFAR-10, Split CIFAR-100 and Split mini-ImageNet. Similar to Aljundi et al. (2019c), an online streaming setting is adopted.

As displayed in Table 2, the task-free *STAF* significantly achieves the highest classification results in 5 out of the 6 experiments, and the joint highest in one experiment. Results of the latter 3 datasets are illustrated with more details in Figures 2-4. Task-free *STAF* is capable of achieving high performance levels due to its similarity-based adaptation as well as its ability to continually learn without task boundaries.

7. This is a brief (yet straight to the point) summary of the wall-clock run-time results.

Table 1: Average test classification accuracy (%) after learning all tasks of the 6 task-aware CL experiments, followed by the standard error. Bold entries indicate significance. Classification accuracy of the proposed *STAF* is significantly higher than the previous state-of-the-art on the six experiments.

Method	S-MNIST	F-MNIST	Omniglot	CIFAR-10	CIFAR-100	mini-ImageNet
P&C	96.4 ± 1.1	90.8 ± 0.9	82.8 ± 1.4	72.4 ± 0.8	65.5 ± 1.3	56.8 ± 1.6
VCL	97.1 ± 0.8	88.0 ± 1.1	80.7 ± 0.7	71.9 ± 1.3	80.0 ± 0.8	53.1 ± 0.7
CCLL	98.8 ± 0.7	92.1 ± 0.4	84.5 ± 0.6	83.2 ± 0.8	90.9 ± 0.9	71.6 ± 1.2
ORTHOG	87.3 ± 0.9	86.8 ± 1.2	80.9 ± 0.8	72.1 ± 0.7	64.3 ± 0.6	51.4 ± 1.4
HNET	99.8 ± 0.02	92.0 ± 0.04	83.4 ± 0.1	83.8 ± 0.1	82.9 ± 0.2	72.2 ± 0.2
AGS-CL	99.4 ± 0.3	91.8 ± 0.3	82.8 ± 1.2	83.5 ± 1.1	64.1 ± 1.0	70.4 ± 1.2
La-MAML	98.3 ± 0.5	91.1 ± 0.5	83.7 ± 0.6	80.7 ± 0.8	70.1 ± 0.7	69.5 ± 0.8
MERLIN	97.4 ± 0.3	91.6 ± 0.2	84.2 ± 0.3	82.9 ± 1.2	48.5 ± 0.3	41.9 ± 1.2
CBRS	95.9 ± 1.2	76.4 ± 2.1	80.3 ± 1.1	72.9 ± 1.1	38.9 ± 1.2	53.2 ± 2.3
GCL	98.9 ± 0.4	91.4 ± 0.4	81.9 ± 0.3	49.6 ± 1.9	74.5 ± 1.0	61.5 ± 0.6
RPS-Net	99.5 ± 0.2	92.2 ± 0.2	84.0 ± 0.4	83.3 ± 0.5	81.8 ± 0.4	75.7 ± 0.6
Gating	99.7 ± 0.1	90.4 ± 0.3	83.6 ± 0.2	96.4 ± 0.2	89.3 ± 0.2	75.2 ± 0.5
RN18	98.3 ± 0.2	90.1 ± 0.2	83.9 ± 0.4	88.6 ± 0.3	56.7 ± 0.23	72.4 ± 0.52
CRNet	99.6 ± 0.1	94.7 ± 0.75	84.7 ± 0.6	93.6 ± 0.4	89.1 ± 0.3	74.2 ± 0.6
NNA	99.6 ± 0.04	85.4 ± 0.5	84.1 ± 0.2	94.5 ± 0.15	83.2 ± 0.15	75.5 ± 0.5
STAF	**99.9** ± 0.02	**97.2** ± 0.2	**89.5** ± 0.09	**98.6** ± 0.13	**95.7** ± 0.19	**83.4** ± 0.4

Table 2: Average classification accuracy (%) of the task-free CL setting for the 6 experiments, followed by the standard error. Significance is identified by bold entries. *STAF* achieves the highest classification accuracy in 5 out of the 6 experiments, and is joint highest in one.

Method	S-MNIST	F-MNIST	Omniglot	CIFAR-10	CIFAR-100	mini-ImageNet
GSS	86.9 ± 1.5	85.7 ± 1.6	82.5 ± 1.1	46.8 ± 0.7	21.2 ± 1.8	22.6 ± 1.4
ER-MIR	87.6 ± 0.7	87.4 ± 0.9	83.4 ± 1.0	49.6 ± 0.2	35.3 ± 1.4	25.2 ± 0.6
GMED + MIR	88.5 ± 1.1	86.2 ± 1.3	80.8 ± 1.4	35.5 ± 1.9	20.8 ± 1.4	27.8 ± 0.7
ASER	86.7 ± 1.3	83.8 ± 1.1	79.1 ± 1.3	43.5 ± 1.4	21.7 ± 0.5	18.2 ± 1.1
CN-DPM	94.4 ± 0.3	93.0 ± 0.4	87.2 ± 1.9	47.0 ± 0.2	21.1 ± 0.2	24.3 ± 0.9
OSAKA	92.5 ± 1.8	90.6 ± 0.9	86.0 ± 0.8	43.4 ± 0.5	32.9 ± 1.1	22.1 ± 1.3
WGF	94.5 ± 1.1	93.8 ± 1.7	85.3 ± 2.2	47.9 ± 2.5	21.8 ± 1.5	32.2 ± 1.5
ODDL	95.8 ± 0.05	95.4 ± 0.09	91.5 ± 0.1	52.7 ± 0.11	27.2 ± 0.87	28.9 ± 1.5
SEDEM	98.4 ± 0.15	96.1 ± 1.2	90.8 ± 1.4	55.3 ± 1.32	24.9 ± 1.16	29.6 ± 1.9
STAF	98.7 ± 0.2	**98.6** ± 0.2	**95.2** ± 0.3	**63.9** ± 0.1	**56.8** ± 0.2	**49.1** ± 0.5

6.3 Catastrophic Forgetting

We evaluate catastrophic forgetting (CF) in a manner similar to the technique presented by Schwarz et al. (2018), which is based checking how the classification accuracy on the first task changes as the learner keeps encountering other tasks, until the end of the training procedure. The CF results on the six benchmarks are displayed in Figure 5. The proposed *STAF* achieves considerably higher retention levels. This empirically demonstrates that the proposed task similarity mechanism manages to mitigate catastrophic forgetting. Reinforcing the (global) CL learning with (local) focussed information from previous relevant tasks has improved the performance retention capability of *STAF*. The results achieved by *STAF*, in terms of the overall classification accuracy as well as catastrophic forgetting reduction, demonstrate its ability to address the stability-plasticity dilemma.

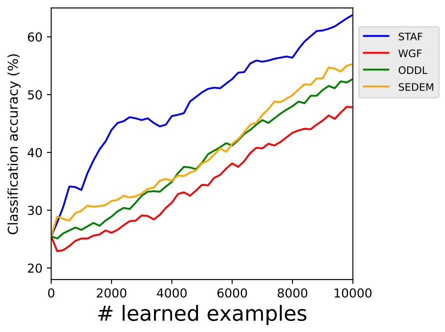


Figure 2: Comparison between state-of-the-art task-free CL algorithms on the Split CIFAR-10 data. Task-free *STAF* outpertforms the previous state-of-the-art.

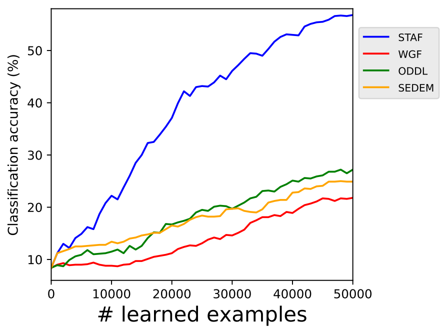


Figure 3: Comparison between task-free CL algorithms on the Split CIFAR-100 data. Highest accuracy is achieved by *STAF*.

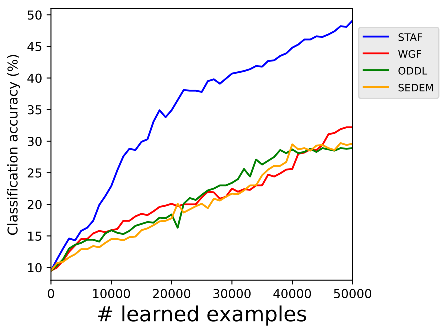
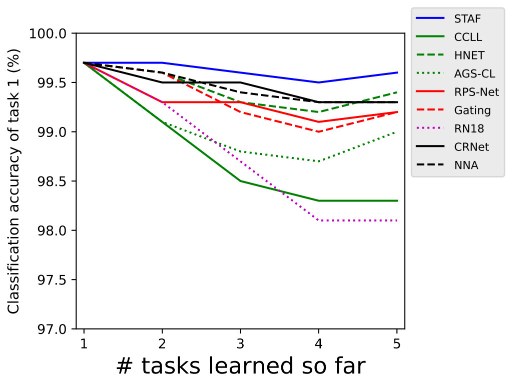
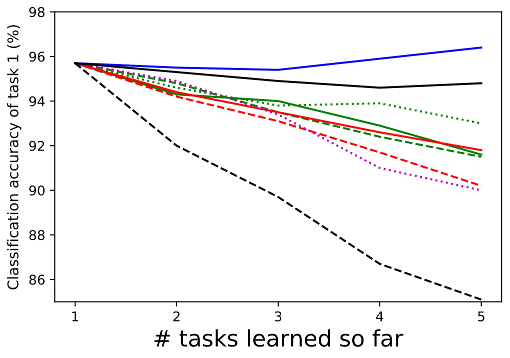


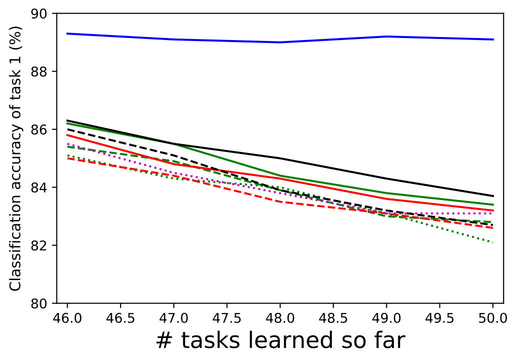
Figure 4: Comparison between task-free CL algorithms on the Split mini-ImageNet data. *STAF* achieves the highest classification accuracy.



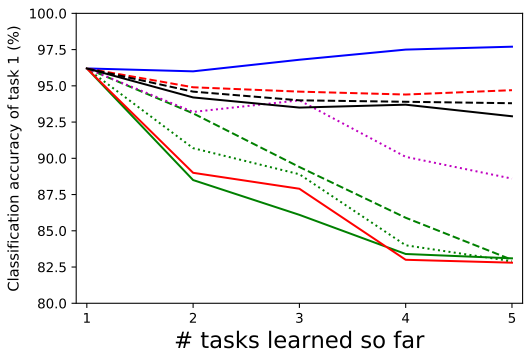
(a) Split MNIST



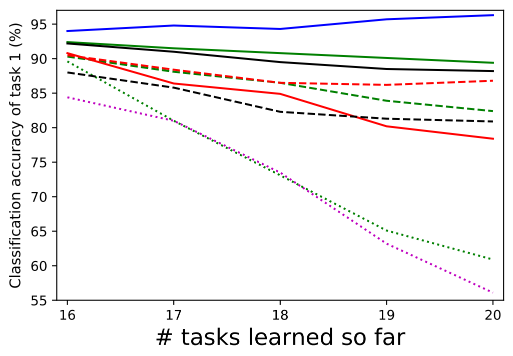
(b) Fashion-MNIST



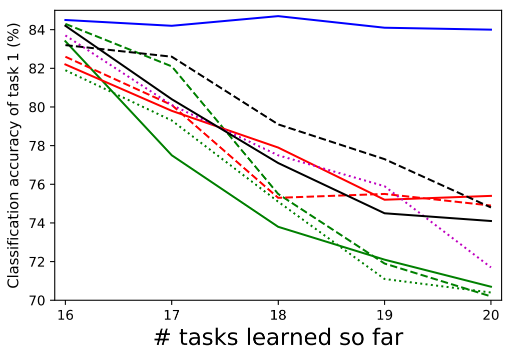
(c) Omniglot



(d) Split CIFAR-10



(e) Split CIFAR-100



(f) Split mini-ImageNet

Figure 5: Catastrophic Forgetting Evaluation. Performance retention is assessed via monitoring how the classification accuracy of Task 1 of each benchmark changes along with the sequential arrival of the other tasks. The proposed *STAF* significantly achieves the highest levels of performance retention in the 6 experiments. Better viewed in color.

6.4 Ablation Study

Results of the performed ablation study for task-aware *STAF* are displayed in Table 3. Such results empirically demonstrate the significance of the proposed adaptation mechanism in achieving the average accumulated classification results obtained by *STAF*. The classification performance of *STAF* after learning all tasks in the six experiments is compared to the following two scenarios: 1) when there is neither an adaptation layer in the main architecture, nor an adaptation mechanism at all, i.e. no PN learning. The main architecture in such case is the vanilla multi-task continual learning architecture solely consisting of shared components and task-specific components. 2) when there is an adaptation mechanism, yet the task based on which adaptation is performed is *randomly* selected. This zooms in further on the value added by the proposed similarity-based adaptation strategy to achieve the classification performance levels obtained by *STAF*. As illustrated in Table 3, significant differences in the overall classification accuracy levels between *STAF* and the other two scenarios demonstrate the relevance of the proposed adaptation strategy.

For task-free *STAF*, the plots in Figures 6-8 depict the ablations for the Split CIFAR-10, Split CIFAR-100 and Split mini-ImageNet datasets, respectively. The significant differences in performance between task-free *STAF* and the other two adaptation scenarios demonstrate the importance of the proposed adaptation strategy.

Table 3: Average test classification accuracy (%) after learning all tasks of the following six experiments: Split MNIST, Fashion-MNIST, Omniglot, Split CIFAR-10, Split CIFAR-100 and Split mini-ImageNet, followed by the standard error. Significance is identified by bold entries. The results obtained by the proposed *STAF* massively depend on its adaptation strategy. This is why *STAF* achieves significantly higher classification levels than the following two scenarios: 1) no adaptation mechanism at all, 2) adaptation via a *randomly* selected task (rather than similarity-based adaptation).

Method	Split MNIST	Fashion-MNIST	Omniglot	CIFAR-10	CIFAR-100	mini-ImageNet
STAF	**99.9** $\pm$ 0.02	**97.2** $\pm$ 0.2	**89.5** $\pm$ 0.09	**98.6** $\pm$ 0.13	**95.7** $\pm$ 0.19	**83.4** $\pm$ 0.4
No Adaptation	95.7 $\pm$ 0.02	80.2 $\pm$ 0.1	79.5 $\pm$ 0.2	71.5 $\pm$ 0.2	35.4 $\pm$ 0.2	32.2 $\pm$ 0.4
Rnd. Sel. Adaptation	96.1 $\pm$ 0.4	78.7 $\pm$ 0.9	78.8 $\pm$ 1.1	72.1 $\pm$ 0.6	31.0 $\pm$ 1.3	30.9 $\pm$ 1.8

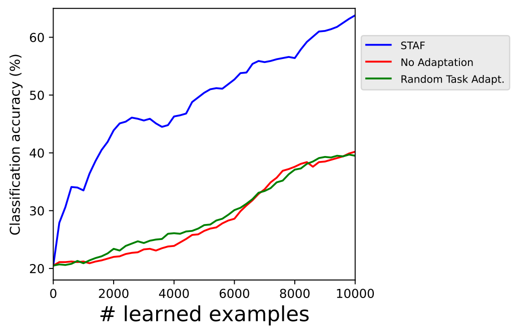


Figure 6: Ablations for task-free CL on Split CIFAR-10. The significant difference in accuracy between task-free *STAF* and the other two adaptation scenarios demonstrate the importance of the proposed adaptation strategy.

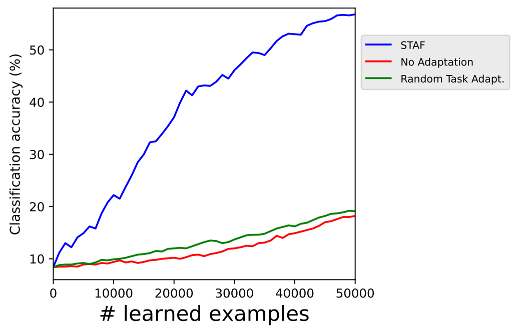


Figure 7: Ablations for task-free CL on Split CIFAR-100. The proposed adaptation strategy massively improves the performance of task-free *STAF*, compared to the other two scenarios.

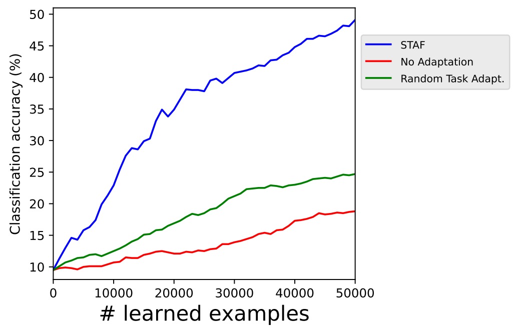


Figure 8: Ablations for task-free CL on the Split mini-ImageNet dataset. *STAF* is considerably more accurate thanks to the proposed adaptation strategy.

7. Conclusion

We introduced a continual learning framework consisting of two algorithms, one for task-aware continual learning, and the other for the more challenging task-free setting. The introduced algorithms learn global (shared across tasks) information, and locally adapt this learning. The proposed similarity-based adaptation is integrated in the model via a proxy prototypical network. We also derived a generalization upper bound on the error of an upcoming task, providing a theoretical corroboration of the proposed methodology. Efficacy of the proposed adaptation mechanism in both the task-aware and task-free settings (measured by the overall classification accuracy and by reducing catastrophic forgetting) is demonstrated via powerful empirical performance over the two continual learning settings.

References

Abati, D., Tomczak, J., Blankevoort, T., Calderara, S., Cucchiara, R., & Bejnordi, B. (2020). Conditional channel gated networks for task-aware continual learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Achille, A., Eccles, T., Matthey, L., Burgess, C., Watters, N., Lerchner, A., & Higgins, I. (2018). Life-long disentangled representation learning with cross-domain latent homologies. *Advances* *in Neural Information Processing Systems (NIPS).*

Adel, T., Zhao, H., & Turner, R. (2020). Continual learning with adaptive weights (CLAW). *International Conference on Learning Representations (ICLR).*

Ahn, H., Kwak, J., Lim, S., Bang, H., Kim, H., & Moon, T. (2021). SS-IL: Separated softmax for incremental learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision.*

Ahn, H., Lee, D., Cha, S., & Moon, T. (2019). Uncertainty-based continual learning with adaptive regularization. *Advances in Neural Information Processing Systems (NeurIPS).*

Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., & Marchand, M. (2014). Domain adversarial neural networks. *CoRR, abs/1412.4446.*

Akyurek, A., Akyurek, E., Wijaya, D., & Andreas, J. (2021). Subspace regularizers for few-shot class incremental learning. *arXiv preprint arXiv:2110.07059.*

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., & Tuytelaars, T. (2018). Memory aware synapses: Learning what (not) to forget. *European Conference on Computer Vision (ECCV).*

Aljundi, R., Caccia, L., Belilovsky, E., Caccia, M., Lin, M., Charlin, L., & Tuytelaars, T. (2019a). Online continual learning with maximally interfered retrieval. *Advances in Neural Information Processing Systems (NeurIPS).*

Aljundi, R., Kelchtermans, K., & Tuytelaars, T. (2019b). Task-free continual learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Aljundi, R., Lin, M., Goujaud, B., & Bengio, Y. (2019c). Gradient based sample selection for online continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Aljundi, R., Rohrbach, M., & Tuytelaars, T. (2019d). Selfless sequential learning. *International Conference on Learning Representations (ICLR).*

Andle, J., Payani, A., & Sekeh, S. (2023). Investigating the impact of weight sharing decisions on knowledge transfer in continual learning. *arXiv preprint arXiv:2311.09506.*

Arani, E., Sarfraz, F., & Zonooz, B. (2022). Learning fast, learning slow: A general continual learning method based on complementary learning system. *International Conference on Learning Representations (ICLR).*

Ayub, A., & Wagner, A. (2021). EEC: Learning to encode and regenerate images for continual learning. *International Conference on Learning Representations (ICLR).*

Bakker, B., & Heskes, T. (2003). Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research (JMLR).*

Banayeeanzade, M., Mirzaiezadeh, R., Hasani, H., & Baghshah, M. (2021). Generative vs discriminative: Rethinking the meta-continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Beaulieu, S., Clune, J., & Cheney, N. (2021). Continual learning under domain transfer with sparse synaptic bursting. *arXiv preprint arXiv:2108.12056.*

Ben-David, S., Blitzer, S., Crammer, K., Kulesza, A., Pereira, F., & Vaughan, J. (2010). A theory of learning from different domains. *Machine learning*, *79*(2), 151–175.

Benavides-Prado, D., Koh, Y., & Riddle, P. (2020). Towards knowledgeable supervised lifelong learning systems. *Journal of Artificial Intelligence Research (JAIR)*, *68*, 159–224.

Benavides-Prado, D., & Riddle, P. (2022). A theory for knowledge transfer in continual learning. *Conference on Lifelong Learning Agents (CoLLAs)*.

Benzing, F. (2020). Understanding regularisation methods for continual learning. *arXiv preprint arXiv:2006.06357*.

Borsos, Z., Mutny, M., & Krause, A. (2020). Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems (NeurIPS)*.

Buzzega, P., Boschini, M., Porrello, A., Abati, D., & Calderara, S. (2020). Dark experience for general continual learning: a strong, simple baseline. *Advances in Neural Information Processing Systems (NeurIPS)*.

Caccia, L., Aljundi, R., Asadi, N., Tuytelaars, T., Pineau, J., & Belilovsky, E. (2022). New insights on reducing abrupt representation change in online continual learning. *International Conference on Learning Representations (ICLR)*.

Caccia, L., Belilovsky, E., Caccia, M., & Pineau, J. (2020a). Online learned continual compression with adaptive quantization modules. *International Conference on Machine Learning (ICML)*.

Caccia, M., Rodriguez, P., Ostapenko, O., Normandin, F., Lin, M., Caccia, L., Laradji, I., Rish, I., Lacoste, A., Vazquez, D., & Charlin, L. (2020b). Online fast adaptation and knowledge accumulation (OSAKA): A new approach to continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Caruana, R. (1997). Multi-task learning. *Machine Learning*.

Cha, S., Hsu, H., Hwang, T., Calmon, F., & Moon, T. (2021a). CPR: Classifier-projection regularization for continual learning. *International Conference on Learning Representations (ICLR)*.

Cha, S., Kim, B., Yoo, Y., & Moon, T. (2021b). SSUL: Semantic segmentation with unknown label for exemplar-based class-incremental learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Chaudhry, A., Dokania, P., Ajanthan, T., & Torr, P. (2018). Riemannian walk for incremental learning: Understanding forgetting and intransigence. *arXiv preprint arXiv:1801.10112*.

Chaudhry, A., Khan, N., Dokania, P., & Torr, P. (2020). Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems (NeurIPS)*.

Chaudhry, A., Ranzato, M., Rohrbach, M., & Elhoseiny, M. (2019a). Efficient lifelong learning with A-GEM. *International Conference on Learning Representations (ICLR)*.

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P., Torr, P., & Ranzato, M. (2019b). Continual learning with tiny episodic memories. *arXiv preprint arXiv:1902.10486*.

Chen, Z., & Liu, B. (2016). Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, *10*, 1–145.

Choi, E., Lee, K., & Choi, K. (2019). Autoencoder-based incremental class learning without retraining on old data. *arXiv preprint arXiv:1907.07872*.

Chrysakis, A., & Moens, M. (2020). Online continual learning from imbalanced data. *International Conference on Machine Learning (ICML).*

Chrysakis, A., & Moens, M. (2023). Simulating task-free continual learning streams from existing datasets. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2516–2524.

Cory, C., Benavides-Prado, D., & Koh, Y. (2021). Continual correction of errors using smart memory replay. *IEEE International Joint Conference on Neural Networks (IJCNN).*

Cossu, A., Graffieti, G., Pellegrini, L., Maltoni, D., Bacciu, D., Carta, A., & Lomonaco, V. (2022). Is class-incremental enough for continual learning?. *Frontiers in Artificial Intelligence.*

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., & Tuytelaars, T. (2021). A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

De Lange, M., & Tuytelaars, T. (2021). Continual prototype evolution: Learning online from non-stationary data streams. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).*

Deng, D., Chen, G., Hao, J., Wang, Q., & Heng, P. (2021). Flattening sharpness for dynamic gradient projection memory benefits continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Deng, J., Dong, W., Socher, R., Li, L., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Deng, Z., Fryer, Z., Juba, B., Panigrahy, R., & Wang, X. (2021). Provable hierarchical lifelong learning with a sketch-based modular architecture. *arXiv preprint arXiv:2112.10919.*

Derakhshani, M., Zhen, X., Shao, L., & Snoek, C. (2021). Kernel continual learning. *International Conference on Machine Learning (ICML).*

Diaz-Rodriguez, N., Lomonaco, V., Filliat, D., & Maltoni, D. (2018). Don't forget, there is more than forgetting: new metrics for Continual Learning. *NIPS Continual Learning Workshop.*

Doan, T., Bennani, M., Mazoure, B., Rabusseau, G., & Alquier, P. (2021). A theoretical analysis of catastrophic forgetting through the NTK overlap matrix. *Artificial Intelligence and Statistics (AISTATS).*

Du, X., Charan, G., Liu, F., & Cao, Y. (2019). Single-net continual learning with progressive segmented training (PST). *arXiv preprint arXiv:1905.11550.*

Ebrahimi, S., Elhoseiny, M., Darrell, T., & Rohrbach, M. (2020a). Uncertainty-guided continual learning with Bayesian neural networks. *International Conference on Learning Representations (ICLR).*

Ebrahimi, S., Meier, F., Calandra, R., Darrell, T., & Rohrbach, M. (2020b). Adversarial continual learning. *European Conference on Computer Vision (ECCV).*

Egorov, E., Kuzina, A., & Burnaev, E. (2021). BooVAE: Boosting approach for continual learning of VAE. *Advances in Neural Information Processing Systems (NeurIPS).*

Ehret, B., Henning, C., Cervera, M., Meulemans, A., von Oswald, J., & Grewe, B. (2021). Continual learning in recurrent neural networks. *International Conference on Learning Representations (ICLR)*.

Evron, I., Moroshko, E., Buzaglo, G., Khriesh, M., Marjieh, B., Srebro, N., & Soudry, D. (2023). Continual learning in linear classification on separable data. *International Conference on Machine Learning (ICML)*.

Farajtabar, M., Azizan, N., Mott, A., & Li, A. (2019). Orthogonal gradient descent for continual learning. *arXiv preprint arXiv:1910.07104*.

Farquhar, S., & Gal, Y. (2018). Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*.

Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A., Pritzel, A., & Wierstra, D. (2017). PathNet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*.

Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. *International Conference on Machine Learning (ICML)*, *34*.

French, R. (1999). Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, *3*, 128–135.

Gao, G., Luo, Z., & Klabjan, D. (2020). Efficient architecture search for continual learning. *arXiv preprint arXiv:2006.04027*.

Gaya, J., Doan, T., Caccia, L., Soulier, L., Denoyer, L., & Raileanu, R. (2023). Building a subspace of policies for scalable continual learning. *International Conference on Learning Representations (ICLR)*.

Ghosh, S., Yao, J., & Doshi-Velez, F. (2018). Structured variational learning of bayesian neural networks with horseshoe priors. *International Conference on Machine Learning (ICML)*.

Gigante, S., Charles, A., Krishnaswamy, S., & Mishne, G. (2019). Visualizing the PHATE of neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*.

Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.

Goodfellow, I., Mirza, M., Xiao, D., Courville, A., & Bengio, Y. (2014a). An empirical investigation of catastrophic forgetting in gradient-based neural networks. *International Conference on Learning Representations (ICLR)*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014b). Generative adversarial nets. *Advances in Neural Information Processing Systems (NIPS)*, 2672–2680.

Goswami, D., Liu, Y., Twardowski, B., & van de Weijer, J. (2023). FeCAM: Exploiting the heterogeneity of class distributions in exemplar-free continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Guo, Y., Liu, B., & Zhao, D. (2022). Online continual learning through mutual information maximization. *International Conference on Machine Learning (ICML)*.

Gupta, G., Yadav, K., & Paull, L. (2020a). La-MAML: Look-ahead meta-learning for continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Gupta, P., Chaudhary, Y., Runkler, T., & Schutze, H. (2020b). Neural topic modeling with continual lifelong learning. *International Conference on Machine Learning (ICML)*.

Hammoud, H., Prabhu, A., Lim, S., Torr, P., Bibi, A., & Ghanem, B. (2023). Rapid adaptation in online continual learning: Are we evaluating it right?. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Hao, J., Ji, K., & Liu, M. (2023). Bilevel coreset selection in continual learning: A new formulation and algorithm. *Advances in Neural Information Processing Systems (NeurIPS)*.

Hattori, M. (2014). A biologically inspired dual-network memory model for reduction of catastrophic forgetting. *Neurocomputing*, *134*, 262–268.

He, X., Sygnowski, J., Galashov, A., Rusu, A., Teh, Y. W., & Pascanu, R. (2019). Task agnostic continual learning via meta learning. *arXiv preprint arXiv:1906.05201*.

Henning, C., Cervera, M., D'Angelo, F., von Oswald, J., Traber, R., Ehret, B., Kobayashi, S., Grewe, B., & Sacramento, J. (2021). Posterior meta-replay for continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Heskes, T. (2000). Empirical Bayes for learning to learn..

Ho, S., Liu, M., Du, L., Gao, L., & Xiang, Y. (2023). Prototype-guided memory replay for continual learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Hou, S., Pan, X., Loy, C., Wang, Z., & Lin, D. (2019). Learning a unified classifier incrementally via rebalancing. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Hung, S., Tu, C., Wu, C., Chen, C., Chan, Y., & Chen, C. (2019). Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Hurtado, J., Raymond-Saez, A., & Soto, A. (2021). Optimizing reusable knowledge for continual learning via meta-learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Isele, D., & Cosgun, A. (2018). Selective experience replay for lifelong learning. *arXiv preprint arXiv:1802.10269*.

Javed, K., & White, M. (2019). Meta-learning representations for continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Jin, X., Du, J., & Ren, X. (2020). Gradient based memory editing for task-free continual learning. *arXiv preprint arXiv:2006.15294*.

Jin, X., Sadhu, A., Du, J., & Ren, X. (2021). Gradient-based editing of memory examples for online task-free continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Joseph, K., & Balasubramanian, V. (2020). Meta-consolidation for continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Jung, S., Ahn, H., Cha, S., & Moon, T. (2020). Continual learning with node-importance based adaptive group sparse regularization. *Advances in Neural Information Processing Systems (NeurIPS)*.

Kamra, N., Gupta, U., & Liu, Y. (2017). Deep generative dual memory network for continual learning. *arXiv preprint arXiv:1710.10368*.

Kang, H., Mina, R., Madjid, S., Yoon, J., Hasegawa-Johnson, M., Hwang, S., & Yoo, C. (2022). Forget-free continual learning with winning subnetworks. *International Conference on Machine Learning (ICML)*.

Kao, T., Jensen, K., van de Ven, G., Bernacchia, A., & Hennequin, G. (2021). Natural continual learning: Success is a journey, not(just) a destination. *Advances in Neural Information Processing Systems (NeurIPS)*.

Kaplanis, C., Shanahan, M., & Clopath, C. (2018). Continual reinforcement learning with complex synapses. *International Conference on Machine Learning (ICML)*.

Kapoor, S., Karaletsos, T., & Bui, T. (2021). Variational auto-regressive Gaussian processes for continual learning. *International Conference on Machine Learning (ICML)*.

Karakida, R., & Akaho, S. (2022). Learning curves for continual learning in neural networks: Self-knowledge transfer and forgetting. *International Conference on Learning Representations (ICLR)*.

Ke, Z., Liu, B., & Huang, X. (2020). Continual learning of a mixed sequence of similar and dissimilar tasks. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ke, Z., Liu, B., Ma, N., Xu, H., & Shu, L. (2021). Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Kemker, R., McClure, M., Abitino, A., Hayes, T., & Kanan, C. (2018). Measuring catastrophic forgetting in neural networks. *AAAI Conference on Artificial Intelligence*, *32*.

Kessler, S., Nguyen, V., Zohren, S., & Roberts, S. (2021). Hierarchical Indian Buffet neural networks for Bayesian continual learning. *Uncertainty in Aritifical Intelligence (UAI)*.

Khan, V., Cygert, S., Twardowski, B., & Trzcinski, T. (2023). Looking through the past: Better knowledge retention for generative replay in continual learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Kim, D., Bae, J., Jo, Y., & Choi, J. (2019). Incremental learning with maximum entropy regularization: Rethinking forgetting and intransigence. *arXiv preprint arXiv:1902.00829*.

Kim, H., Kim, S., & Lee, J. (2018). Keep and learn: Continual learning by constraining the latent space for knowledge preservation in neural networks. *MICCAI*.

Kingma, D., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR)*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., & Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences (PNAS)*.

Knoblauch, J., Husain, H., & Diethe, T. (2020). Optimal continual learning has perfect memory and is NP-HARD. *International Conference on Machine Learning (ICML)*.

Koh, H., Kim, D., Ha, J., & Choi, J. (2022). Online continual learning on class incremental blurry task configuration with anytime inference. *International Conference on Learning Representations (ICLR)*.

Krishnan, R., & Balaprakash, P. (2021). Formalizing the generalization-forgetting trade-Off in continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*.

Kumar, A., Chatterjee, S., & Rai, P. (2021). Bayesian structural adaptation for continual learning. *International Conference on Machine Learning (ICML)*.

Kurle, R., Cseke, B., Klushyn, A., van der Smagt, P., & Gunnemann, S. (2020). Continual learning with Bayesian neural networks for non-stationary data. *International Conference on Learning Representations (ICLR)*.

Lacoste, A., Rodríguez, P., Branchaud-Charron, F., Atighehchian, P., Caccia, M., Laradji, I., Drouin, A., Craddock, M., Charlin, L., & Vazquez, D. (2020). Synbols: Probing learning algorithms with synthetic datasets. *Advances in Neural Information Processing Systems (NeurIPS)*.

Lake, B., Salakhutdinov, R., Gross, J., & Tenenbaum, J. (2011). One shot learning of simple visual concepts. *Proceedings of the Cognitive Science Society*, *33*.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *In Proceedings of the IEEE*, *86*(11), 2278–2324.

Lee, K., Zhong, Y., & Wang, Y. (2023). Do pre-trained models benefit equally in continual learning?. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Lee, S., Goldt, S., & Saxe, A. (2021). Continual learning in the teacher-student setup: Impact of task similarity. *International Conference on Machine Learning (ICML)*.

Lee, S., Ha, J., Zhang, D., & Kim, G. (2020). A neural Dirichlet process mixture model for task-free continual learning. *International Conference on Learning Representations (ICLR)*.

Lee, S., Kim, J., Jun, J., Ha, J., & Zhang, B. (2017). Overcoming catastrophic forgetting by incremental moment matching. *Advances in Neural Information Processing Systems (NIPS)*.

Li, D., Yang, Y., Song, Y., & Hospedales, T. (2018). Learning to generalize: Meta-learning for domain generalization. *AAAI Conference on Artificial Intelligence*, *32*.

Li, D., & Zeng, Z. (2023). CRNet: A fast continual learning framework with random theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(9).

Li, H., Enshaeifar, S., Ganz, F., & Barnaghi, P. (2019a). Continual learning in deep neural network by using a Kalman optimiser. *ICML Workshop*.

Li, X., Zhou, Y., Wu, T., Socher, R., & Xiong, C. (2019b). Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. *International Conference on Machine Learning (ICML)*.

Li, Z., & Hoiem, D. (2016). Learning without forgetting. *European Conference on Computer Vision (ECCV)*.

Lin, S., Ju, P., Liang, Y., & Shroff, N. (2023). Theory on forgetting and generalization of continual learning. *International Conference on Machine Learning (ICML)*.

Lin, S., Yang, L., Fan, D., & Zhang, J. (2022). TRGP: Trust region gradient projection for continual learning. *International Conference on Learning Representations (ICLR)*.

Liu, H., & Liu, H. (2022). Continual learning with recursive gradient optimization. *International Conference on Learning Representations (ICLR).*

Liu, Y., Schiele, B., & Sun, Q. (2021). RMM: Reinforced memory management for class-incremental learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Loo, N., Swaroop, S., & Turner, R. (2021). Generalized variational continual learning. *International Conference on Learning Representations (ICLR).*

Lopez-Paz, D., & Ranzato, M. (2017). Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems (NIPS).*

Lyu, Y., Wang, L., Zhang, X., Sun, Z., Su, H., Zhu, J., & Jing, L. (2023). Overcoming recency bias of normalization statistics in continual learning: Balance and adaptation. *Advances in Neural Information Processing Systems (NeurIPS).*

Madaan, D., Yoon, J., Li, Y., Liu, Y., & Hwang, S. (2022). Rethinking the representational continuity: Towards unsupervised continual learning. *International Conference on Learning Representations (ICLR).*

Madireddy, S., Yanguas-Gil, A., & Balaprakash, P. (2023). Improving performance in continual learning tasks using bio-inspired architectures. *Conference on Lifelong Learning Agents (CoLLAs).*

Mao, F., Weng, W., Pratama, M., & Yee, E. (2021). Continual learning via inter-task synaptic mapping. *Knowledge-Based Systems*, *222*.

Masarczyk, W., Wawrzynski, P., Marczak, D., Deja, K., & Trzcinski, T. (2022). Logarithmic continual learning. *IEEE Access*, *10*.

McCloskey, M., & Cohen, N. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*.

Mehta, S., Patil, D., Chandar, S., & Strubell, E. (2021). An empirical investigation of the role of pre-training in lifelong learning. *arXiv preprint arXiv:2112.09153*.

Miao, Z., Wang, Z., Chen, W., & Qiu, Q. (2022). Continual learning with filter atom swapping. *International Conference on Learning Representations (ICLR).*

Mirzadeh, S., Chaudhry, A., Yin, D., Nguyen, T., Pascanu, R., Gorur, D., & Farajtabar, M. (2022). Architecture matters in continual learning. *arXiv preprint arXiv:2202.00275*.

Mirzadeh, S., Farajtabar, M., Gorur, D., Pascanu, R., & Ghasemzadeh, H. (2021). Linear mode connectivity in multitask and continual learning. *International Conference on Learning Representations (ICLR).*

Mirzadeh, S., Farajtabar, M., Pascanu, R., & Ghasemzadeh, H. (2020). Understanding the role of training regimes in continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Mocanu, D., Vega, M., Eaton, E., Stone, P., & Liotta, A. (2016). Online contrastive divergence with generative replay: Experience replay without storing data. *arXiv preprint arXiv:1610.05555*.

Morawiecki, P., Wolczyk, M., Krutsylo, A., & Smieja, M. (2022). Hebbian continual representation learning. *arXiv preprint arXiv:2207.04874*.

Morgado, P., & Vasconcelos, N. (2019). NETTAILOR: Tuning the architecture, not just the weights. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mundt, M., Hong, Y., Pliushch, I., & Ramesh, W. (2023). A wholistic view of continual learning with deep neural networks: Forgotten lessons and the bridge to active and open world learning. *Neural Networks*.

Mundt, M., Lang, S., Delfosse, Q., & Kersting, K. (2022). CLEVA-compass: A continual learning evaluation assessment compass to promote research transparency and comparability. *International Conference on Learning Representations (ICLR)*.

Nguyen, C., Achille, A., Lam, M., Hassner, T., Mahadevan, V., & Soatto, S. (2019). Toward understanding catastrophic forgetting in continual learning. *arXiv preprint arXiv:1908.01091*.

Nguyen, C., Hassner, T., Seeger, M., & Archambeau, C. (2020). LEEP: A New measure to evaluate transferability of learned representations. *International Conference on Machine Learning (ICML)*.

Nguyen, C., Li, Y., Bui, T., & Turner, R. (2018). Variational continual learning. *International Conference on Learning Representations (ICLR)*.

Ostapenko, O., Puscas, M., Klein, T., Jahnichen, P., & Nabi, M. (2019). Learning to remember: A synaptic plasticity driven framework for continual learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ostapenko, O., Rodriguez, P., Caccia, M., & Charlin, L. (2021). Continual learning via local module composition. *Advances in Neural Information Processing Systems (NeurIPS)*.

Pan, P., Swaroop, S., Immer, A., Eschenhagen, R., Turner, R., & Khan, M. (2020). Continual deep learning by functional regularisation of memorable past. *Advances in Neural Information Processing Systems (NeurIPS)*.

Pape, L., Gomez, F., Ring, M., & Schmidhuber, J. (2011). Modular deep belief networks that do not forget. *IEEE International Joint Conference on Neural Networks (IJCNN)*.

Parisi, G., Kemker, R., Part, J., Kanan, C., & Wermter, S. (2019). Continual lifelong learning with neural networks: A review. *Neural Networks*.

Parisi, G., & Lomonaco, V. (2020). Online continual learning on sequences. *Recent Trends in Learning From Dat*.

Park, D., Hong, S., Han, B., & Lee, K. (2019). Continual learning by asymmetric loss approximation with single-side overestimation. *arXiv preprint arXiv:1908.02984*.

Pfulb, B., & Gepperth, A. (2019). A comprehensive, application-oriented study of catastrophic forgetting in DNNs. *International Conference on Learning Representations (ICLR)*.

Pham, Q., Liu, C., & Hoi, S. (2021). DualNet: Continual learning, fast and slow. *Advances in Neural Information Processing Systems (NeurIPS)*.

Pham, Q., Liu, C., & Hoi, S. (2022). Continual normalization: Rethinking batch normalization for online continual learning. *International Conference on Learning Representations (ICLR)*.

Pham, Q., Liu, C., Sahoo, D., & Hoi, S. (2021). Contextual transformation networks for online continual learning. *International Conference on Learning Representations (ICLR)*.

Pourcel, J., Vu, N., & French, R. (2022). Online task-free continual learning with dynamic sparse distributed memory. *European Conference on Computer Vision (ECCV)*.

Powers, S., Xing, E., Kolve, E., Mottaghi, R., & Gupta, A. (2022). CORA: Benchmarks, baselines, and metrics as a platform for continual reinforcement learning agents. *Conference on Lifelong Learning Agents (CoLLAs)*.

Pratama, M., Ashfahani, A., & Lughofer, E. (2021). Unsupervised continual learning via self-adaptive deep clustering approach. *arXiv:2106.14563*.

Qin, Q., Peng, H., Hu, W., Zhao, D., & Liu, B. (2021). BNS: Building network structures dynamically for continual learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Rajasegaran, J., Hayat, M., Khan, S., Khan, F., & Shao, L. (2019). Random path selection for incremental learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ramasesh, V., Dyer, E., & Raghu, M. (2021). Anatomy of catastrophic forgetting: Hidden representations and task semantics. *International Conference on Learning Representations (ICLR)*.

Ramesh, R., & Chaudhari, P. (2022). Model zoo: A growing brain that learns continually. *International Conference on Learning Representations (ICLR)*.

Rao, D., Visin, F., Rusu, A., Teh, Y., Pascanu, R., & Hadsell, R. (2019). Continual unsupervised representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Ratcliff, R. (1990). Connectionist models of recognition memory: Constraints imposed by learning and forgetting functions. *Psychological Review*.

Rebuffi, S., Bilen, H., & Vedaldi, A. (2017). Learning multiple visual domains with residual adapters. *Advances in Neural Information Processing Systems (NIPS)*.

Rebuffi, S., Bilen, H., & Vedaldi, A. (2018). Efficient parametrization of multi-domain deep neural networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Rebuffi, S., Kolesnikov, A., Sperl, G., & Lampert, C. (2017). iCaRL: Incremental classifier and representation learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Requeima, J., Gordon, J., Bronskill, J., Nowozin, S., & Turner, R. (2019). Fast and flexible multi-task classification using conditional neural adaptive processes. *Advances in Neural Information Processing Systems (NeurIPS)*.

Riemer, M., Cases, I., Ajemian, R., Liu, M., I.Rish, Tu, Y., & Tesauro, G. (2019). Learning to learn without forgetting by maximizing transfer and minimizing interference. *International Conference on Learning Representations (ICLR)*.

Ring, M. (1995). *Continual learning in reinforcement environments*. Ph.D. thesis, University of Texas, Austin.

Ring, M. (1997). CHILD: A first step towards continual learning. *Machine Learning*.

Robins, A. (1993). Catastrophic forgetting in neural networks: The role of rehearsal mechanisms. *IEEE Artificial Neural Networks and Expert Systems*, 65–68.

Robins, A. (1995). Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, *7*, 123–146.

Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T., & Wayne, G. (2018). Experience replay for continual learning. *arXiv preprint arXiv:1811.11682*.

Romero, D., Bruintjes, R., Bekkers, E., Tomczak, J., Hoogendoorn, M., & van Gemert, J. (2022a). FLEXCONV: Continuous kernel convolutions with differentiable kernel sizes. *International Conference on Learning Representations (ICLR)*.

Romero, D., Kuzina, A., Bekkers, E., Tomczak, J., & Hoogendoorn, M. (2022b). CKCONV: Continuous kernel convolution for sequential data. *International Conference on Learning Representations (ICLR)*.

Rosenfeld, A., & Tsotsos, J. (2018). Incremental learning through deep adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *42*(3), 651–663.

Rostami, M., Isele, D., & Eaton, E. (2020a). Using task descriptions in lifelong machine learning for improved performance and zero-shot transfer. *Journal of Artificial Intelligence Research (JAIR)*, *67*, 673–703.

Rostami, M., Kolouri, S., McClelland, J., & Pilly, P. (2020b). Generative continual concept learning. *AAAI Conference on Artificial Intelligence*, *34*.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A., & Fei-Fei, L. (2015). ImageNet large scale visual recognition challenge. *IJCV*.

Rusu, A., Rabinowitz, N., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., & Hadsell, R. (2016a). Progressive neural networks. *arXiv preprint arXiv:1606.04671*.

Rusu, A., Vecerik, M., Rothoerl, T., Heess, N., Pascanu, R., & Hadsell, R. (2016b). Sim-to-Real robot learning from pixels with progressive nets. *arXiv preprint arXiv:1610.04286*.

Rymarczyk, D., van de Weijer, J., Zielinski, B., & Twardowski, B. (2023). ICICLE: Interpretable Class Incremental Continual Learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Saha, G., Garg, I., & Roy, K. (2021). Gradient projection memory for continual learning. *International Conference on Learning Representations (ICLR)*.

Schlimmer, J., & Fisher, D. (1986). A case study of incremental concept induction. *The National Conference on Artificial Intelligence*.

Schmidhuber, J. (2013). Powerplay: Training an increasingly general problem solver by continually searching for the simplest still unsolvable problem. *Frontiers in Psychology*, *4*.

Schmidhuber, J. (2018). One big net for everything. *arXiv preprint arXiv:1802.08864*.

Schwarz, J., Luketina, J., Czarnecki, W., Grabska-Barwinska, A., Teh, Y. W., Pascanu, R., & Hadsell, R. (2018). Progress & compress: A scalable framework for continual learning. *International Conference on Machine Learning (ICML)*.

Serra, J., Suris, D., Miron, M., & Karatzoglou, A. (2018). Overcoming catastrophic forgetting with hard attention to the task. *International Conference on Machine Learning (ICML)*.

Shim, D., Mai, Z., Jeong, J., Sanner, S., Kim, H., & Jang, J. (2020). Adversarial Shapley value experience replay for task-free continual learning. *arXiv preprint avrXiv:2009.00093*.

Shim, D., Mai, Z., Jeong, J., Sanner, S., Kim, H., & Jang, J. (2021). Online class-incremental continual learning with adversarial Shapley value. *arXiv preprint arXiv:2009.00093*.

Shin, H., Lee, J., Kim, J., & Kim, J. (2017). Continual learning with deep generative replay. *Advances in Neural Information Processing Systems (NIPS)*.

Singh, P., Verma, V., Mazumder, P., Carin, L., & Rai, P. (2020). Calibrating CNNs for lifelong learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Skantze, G., & Willemsen, B. (2022). CoLLIE: Continual learning of language grounding from language-image embeddings. *Journal of Artificial Intelligence Research (JAIR)*, *74*, 1201–1223.

Smith, J., Seymour, Z., & Chiu, H. (2022). Incremental learning with differentiable architecture and forgetting search. *IEEE International Joint Conference on Neural Networks (IJCNN)*.

Smith, J., Taylor, C., Baer, S., & Dovrolis, C. (2019). Unsupervised progressive learning and the STAM architecture. *arXiv:1904.02021*.

Smith, J., Tian, J., Halbe, S., Hsu, Y., & Kira, Z. (2023). A closer look at rehearsal-free continual rearning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems (NIPS)*.

Srivastava, R., Masci, J., Kazerounian, S., Gomez, F., & Schmidhuber, J. (2013). Compete to compute. *Advances in Neural Information Processing Systems (NIPS)*.

Stickland, A., & Murray, I. (2019). BERT and PALs: Projected attention layers for efficient adaptation in multi-task learning. *International Conference on Machine Learning (ICML)*.

Sun, Q., Chattopadhyay, R., Panchanathan, S., & Ye, J. (2011). A two-stage weighting framework for multi-source domain adaptation. *Advances in Neural Information Processing Systems (NIPS)*.

Sun, S., Calandriello, D., Hu, H., Li, A., & Titsias, M. (2022). Information-theoretic online memory selection for continual learning. *International Conference on Learning Representations (ICLR)*.

Sutton, R., & Whitehead, S. (1993). Online learning with random representations. *International Conference on Machine Learning (ICML)*.

Tang, B., & Matteson, D. (2021). Graph-based continual learning. *International Conference on Learning Representations (ICLR)*.

Taylor, M., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research (JMLR)*, 1633–1685.

Teng, D., & Dasgupta, S. (2019). Continual learning via online leverage score sampling. *arXiv preprint arXiv:1908.00355*.

Thrun, S. (1996). Explanation-based neural network learning: A lifelong learning approach. *Springer Science & Business Media*, *357*.

Titsias, M., Schwarz, J., Matthews, A., Pascanu, R., & Teh, Y. W. (2019). Functional regularisation for continual learning using Gaussian processes. *arXiv preprint arXiv:1901.11356*.

Tiwari, R., Killamsetty, K., Iyer, R., & Shenoy, P. (2022). GCR: Gradient coreset based replay buffer selection for continual learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

van de Ven, G., & Tolias, A. (2018). Generative replay with feedback connections as a general strategy for continual learning. *arXiv preprint arXiv:1809.10635.*

Veniat, T., Denoyer, L., & Ranzato, M. (2021). Efficient continual learning with modular networks and task-driven priors. *International Conference on Learning Representations (ICLR).*

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., & Wierstra, D. (2016). Matching networks for one shot learning. *Advances in Neural Information Processing Systems (NIPS).*

von Oswald, J., Henning, C., Sacramento, J., & Grewe, B. (2020). Continual learning with hypernetworks. *International Conference on Learning Representations (ICLR).*

von Oswald, J., Zhao, D., Kobayashi, S., Schug, S., Caccia, M., Zucchet, N., & Sacramento, J. (2021). Learning where to learn: Gradient sparsity in meta and continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Vuorio, R., Cho, D., Kim, D., & Kim, J. (2018). Meta continual learning. *arXiv preprint arXiv:1806.06928.*

Wang, L., Zhang, M., Jia, Z., Li, Q., Ma, K., Bao, C., Zhu, J., & Zhong, Y. (2021). AFEC: Active forgetting of negative transfer in continual learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Wang, L., Zhang, X., Yang, K., Yu, L., Li, C., Hong, L., Zhang, S., Li, Z., Zhong, Y., & Zhu, J. (2022). Memory replay with data compression for continual learning. *International Conference on Learning Representations (ICLR).*

Wang, X., Yao, L., Wang, X., Paik, H., & Wang, S. (2023). Uncertainty estimation with neural processes for meta-continual learning. *IEEE Transactions on Neural Networks and Learning Systems.*

Wang, Z., Shen, L., Fang, L., Suo, Q., Duan, T., & Gao, M. (2022a). Improving task-free continual learning by distributionally robust memory Evolution. *International Conference on Machine Learning (ICML).*

Wang, Z., Zhang, Z., Ebrahimi, S., Sun, R., Zhang, H., Lee, C., Ren, X., Su, G., Perot, V., Dy, J., & Pfister, T. (2022b). DualPrompt: Complementary prompting for rehearsal-free continual learning. *European Conference on Computer Vision (ECCV).*

Wang, Z., Zhang, Z., Lee, C., Zhang, H., Run, R., Ren, X., Su, G., Perot, V., Dy, J., & Pfister, T. (2023). Learning to prompt for continual learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 24214–24223.

Wu, C., Herranz, L., Liu, X., Wang, Y., van de Weijer, J., & Raducanu, B. (2018). Memory replay GANs: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems (NIPS).*

Wu, T., Caccia, M., Li, Z., Li, Y., Qi, G., & Haffari, G. (2022). Pretrained language model in continual learning: A comparative study. *International Conference on Learning Representations (ICLR).*

Wu, Y., Chen, Y., Wang, L., Ye, Y., Liu, Z., Guo, Y., & Fu, Y. (2019). Large scale incremental learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Wu, Z., Tran, H., Pirsiavash, H., & Kolouri, S. (2023). Is multi-task learning an upper bound for continual learning?. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*

Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747.*

Xu, J., Ma, J., & Zhu, Z. (2019). Bayesian Optimized Continual Learning with Attention Mechanism. *arXiv preprint arXiv:1905.03980.*

Xu, J., & Zhu, Z. (2018). Reinforced continual learning. *Advances in Neural Information Processing Systems (NIPS).*

Yap, P., Ritter, H., & Barber, D. (2020). Bayesian online meta-learning with Laplace approximation. *arXiv preprint arXiv:2005.00146.*

Yasar, M., & Iqbal, T. (2023). CoRaL: Continual representation learning for overcoming catastrophic forgetting. *International Conference on Autonomous Agents and Multiagent Systems (AAMAS).*

Ye, F., & Bors, A. (2022). Task-free continual learning via online discrepancy distance learning. *Advances in Neural Information Processing Systems (NeurIPS).*

Ye, F., & Bors, A. (2023). Self-evolved dynamic expansion model for task-free continual learning. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).*

Yin, H., Yang, P., & Li, P. (2021). Mitigating forgetting in online continual learning with neuron calibration. *Advances in Neural Information Processing Systems (NeurIPS).*

Yoon, J., Jeong, W., Lee, G., Yang, E., & Hwang, S. (2021). Federated continual learning with weighted inter-client transfer. *International Conference on Machine Learning (ICML).*

Yoon, J., Kim, S., Yang, E., & Hwang, S. (2020). Scalable and order-robust continual learning with additive parameter decomposition. *International Conference on Learning Representations (ICLR).*

Yoon, J., Madaan, D., Yang, E., & Hwang, S. (2022). Online coreset selection for rehearsal-based continual learning. *International Conference on Learning Representations (ICLR).*

Yoon, J., Yang, E., Lee, J., & Hwang, S. (2018). Lifelong learning with dynamically expandable networks. *International Conference on Learning Representations (ICLR).*

Yu, L., Twardowski, B., Liu, X., Herranz, L., Wang, K., Cheng, Y., Jui, S., & van de Weijer, J. (2020). Semantic drift compensation for class-incremental learning. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

Zenke, F., Poole, B., & Ganguli, S. (2017). Continual learning through synaptic intelligence. *International Conference on Machine Learning (ICML).*

Zeno, C., Golan, I., Hoffer, E., & Soudry, D. (2018). Task agnostic continual learning using online variational Bayes. *NIPS Bayesian Deep Learning Workshop.*

Zhang, J., Zhang, J., Ghosh, S., Li, D., Tasci, S., Heck, L., Zhang, H., & Kuo, C. (2020). Class-incremental learning via deep model consolidation. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Zhao, H., des Combes, R. T., Zhang, K., & Gordon, G. (2019). On learning invariant representations for domain adaptation. *International Conference on Machine Learning (ICML)*.

Zhu, F., Cheng, Z., Zhang, X., & Liu, C. (2021). Class-incremental learning via dual augmentation. *Advances in Neural Information Processing Systems (NeurIPS)*.