A Simple Recipe to Meta-Learn Forward and Backward Transfer

Edoardo Cetin
King's College London
Department of Engineering
edoardo.cetin@kcl.ac.uk

Antonio Carta
University of Pisa
Department of Computer Science
antonio.carta@unipi.it

Oya Celiktutan
King's College London
Department of Engineering
oya.celiktutan@kcl.ac.uk

Abstract

Meta-learning holds the potential to provide a general and explicit solution to tackle interference and forgetting in continual learning. However, many popular algorithms introduce expensive and unstable optimization processes with new key hyper-parameters and requirements, hindering their applicability. We propose a new, general, and ***si****mple* ***m****eta-learning algorithm for* ***c****ontinual learning (****SiM4C****) that explicitly optimizes to minimize forgetting and facilitate forward transfer. We show our method is stable, introduces only minimal computational overhead, and can be integrated with any memory-based continual learning algorithm in only a few lines of code. SiM4C meta-learns how to effectively continually learn even on very long task sequences, largely outperforming prior meta-approaches. Naively integrating with existing memory-based algorithms, we also record universal performance benefits and state-of-the-art results across different visual classification benchmarks without introducing new hyper-parameters.*

1. Introduction

Continual learning considers the problem of appropriately updating and consolidating knowledge when learning from data with a continual distribution shift. This problem setting is extremely relevant to achieving lifelong learning systems where an agent is expected to encounter drifts both across and within learning environments, due to the non-stationary nature of the real-world [31]. An ideal continual learning system would re-use prior knowledge to facilitate learning new problems (forward transfer), and new data to refine its prior predictions (backward transfer). However, deep supervised learning algorithms struggle to even *retain* learned information when trained with non-stationary sequences of data, a problem known as catastrophic forgetting [14, 15, 36]. In classical computer vision applications, prior work is still far from addressing this issue [11], which would be crucial to bringing the generality and applicability of deep learning closer to its natural analogue [18, 27].

Traditional algorithms for continual learning can be generally classified into (i) replay methods, (ii) regularization-based methods, and (iii) parameter isolation methods [11]. These approaches entail hand-designed strategies involving (i) storing a small buffer of samples from all prior tasks, (ii) introducing auxiliary regularization terms to consolidate knowledge, and (iii) defining modular architectures to separate task-specific knowledge into independent modules. Given its relevance, most methods only optimize to mitigate forgetting, ignoring other potentially desirable properties of continual learning [12]. Furthermore, they rely on heuristics and strong assumptions about the types of drifts, limiting their applicability to specific settings [48]. A fourth category of methods has recently found success, based on utilizing meta-learning to *emergently* recover many of the wanted properties of an effective continual learning agent [7].

The general principle behind these prior meta-learning approaches has been to emulate the non i.i.d. continual learning optimization *within* an outer meta-optimization to improve the agent's stability (knowledge retention) and plasticity (knowledge acquisition) [16]. However, optimizing these meta-objectives often requires high computational cost [41] and even access to a separate meta pre-training phase where the constraints of continual learning are essentially lifted [23]. Furthermore, different recent advances have focused on aligning the meta and continual optimizations by increasing the number of steps, adding new models, and introducing new parameters to meta-train [4, 17]. The underlying complexity of this line of work comes with non-trivial implications hindering applicability, efficiency, and also introducing the need of heuristic approximations to the meta-objectives for tractability [3, 38].

In this work, we propose a new **si**mple **m**eta-learner for **c**ontinual learning (**SiM4C**) that takes a different approach than prior work. In particular, we propose to purposefully use a lightweight *single-step* inner-loop in the meta-optimization and preserve large amounts of 'unseen' data for each task which can be interpreted as potential *future data*. Then, using both this future data and past stored experiences in the outer meta-loss, we explicitly optimize the

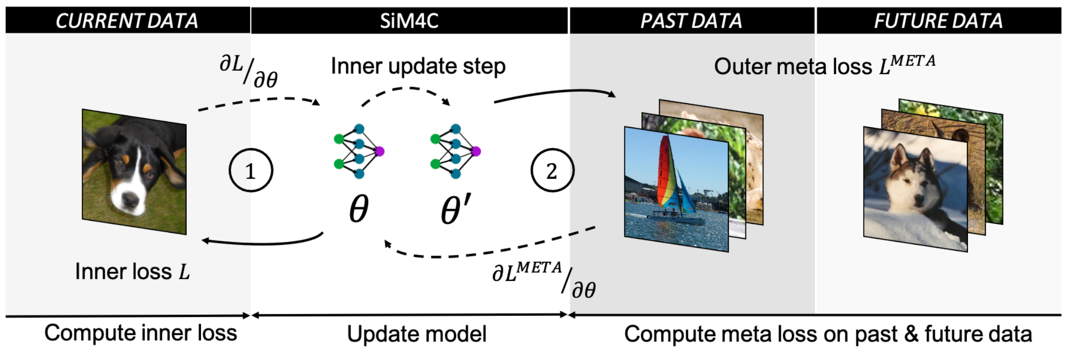
CURRENT DATA
SiM4C
PAST DATA
FUTURE DATA
$\partial L / \partial \theta$
Inner update step
Outer meta loss L^{META}
1
2
θ
θ'
Inner loss L
$\partial L^{META} / \partial \theta$
Compute inner loss
Update model
Compute meta loss on past & future data

Figure 1. Schematic depiction of SiM4C, after a single inner optimization step the proposed meta-objective optimizes for forward and backward transfer by utilizing seen *past data* from previous tasks and unseen *future data* of the current task.

model to tackle challenging *future and past* transfer objectives, across and within tasks (as depicted in Figure 1). In practice, our formulation results in SiM4C avoiding harmful meta-overfitting phenomena, providing minimal computational overheads even without approximations, and allowing integrations in just a few lines of code.

Empirically, we thoroughly evaluate and analyze SiM4C for image classification problems, comparing it with different state-of-the-art algorithms. When given access to a meta pre-training phase, SiM4C greatly outperforms other meta-learning work evaluated and designed for tackling long sequences of hundreds of consecutive tasks [4, 23]. Furthermore, even without any pre-training, using SiM4C as a simple auxiliary meta-objective provides near-universal performance benefits to different memory-based algorithms [6, 41]. Our results validate SiM4C's effectiveness, achieving state-of-the-art results across five different continual learning benchmarks, and highlight its improved efficiency and stability as compared to prior meta-learning approaches. We provide additional resources and our full implementation[1] to facilitate future extensions and promote further work towards more scalable and applicable meta-learning for continual learning.

2. Background

2.1. Continual learning

The problem of continual learning is to learn from a stream of T consecutive tasks sampled from a task set $\mathcal{T}$. In supervised continual classification problems, each task is represented by a different distribution D_t, $t = 1, 2, \ldots, T$ over exemplars and their respective target labels $(x, y) \sim D_t$. At any given timestep t, a continual learning model f_θ has only access to a limited number (n_t) of samples $D_t^{\text{train}} = \{(x_i, y_i)\}_{i=1}^{n_t}$ from the current task D_t while losing access to all previous tasks. However, after experiencing t tasks, its objective L_t^{CL} is then to minimize a cross-entropy loss with respect to the distribution of all seen tasks:

$$L_t^{\text{CL}}(\theta) = \sum_{i=1}^{t} {}_{(x,y)\sim D_t} \left[L^{ce}(f_\theta(x), y)\right], \tag{1}$$

where the continual learning model f_θ is assumed to output a set of C total class logits for all tasks and L^{ce} is defined as a standard softmax cross-entropy classification loss:

$$L^{ce}(f_\theta(x), y) = \log\left(\frac{\exp f_\theta(x)_y}{\sum_{j=1}^{C} \exp f_\theta(x)_j}\right).$$

Based on how different classes are partitioned and the information contained in the samples x, we can further divide continual learning into different sub-settings [48]. In particular, **(i) task-incremental learning (Task-IL)** and **(ii) class-incremental learning (Class-IL)** assume different tasks comprise different subsets of the classes, such that $y \neq y'$ if $y \sim D_i, y' \sim D_j$, and $i \neq j$. However, Task-IL further assumes that task information is provided as part of the input features x, allowing the classifier to ignore all non-relevant class logits at training and test time. **(iii) Domain-incremental learning (Domain-IL)** assumes some inherent unknown distribution shift tied to the data points from different classes but no mutual exclusivity of the respective labels. Finally, **(iv) online continual learning** does not assume access to task information and further introduces the constraint that the agent can only see once each example provided from the training set of each task. The online continual learning setting is similar to the **general continual learning** setting described by Buzzega et al. [6].

In practice, replay-based continual learning methods relax the constraint of access to previous task data by keeping a *memory buffer* $\mathcal{M}$ of stored samples from past tasks with some constant memory cost. To populate and choose which samples to evict from $\mathcal{M}$, *reservoir sampling* [50] is typically employed, ensuring that the state of the memory buffer

[1] https://sites.google.com/view/sim4c/

after each task $\mathcal{M}_t$ contains i.i.d. samples from all seen data points. To mitigate catastrophic forgetting, these methods then train f_θ with a weighted combination of the cross-entropy loss using available training data from the current task D_t^{train} and an auxiliary loss L^{aux} using $\mathcal{M}_t$:

$$\begin{aligned} L_t^{\text{MB}}(\theta) =& E_{(x,y)\sim D_t^{\text{train}}}\left[L^{ce}(f_\theta(x), y)\right] \\ &+ \alpha E_{(x,y)\sim \mathcal{M}_t}\left[L^{aux}(x, y|\theta)\right]. \end{aligned} \tag{2}$$

For instance, an effective baseline is to simply employ an additional softmax cross-entropy loss for L^{aux}. Dark Experience Replay (DER) [6] is a recent extension using *dark knowledge* [21] and data augmentation to distill past information into the current model. In particular, DER stores the set of past model logits $z = f_\theta(x)$ in $\mathcal{M}$ and uses them to minimize mean-squared error with the current model's predictions, either to substitute (L^{DER}) or complement the standard cross-entropy auxiliary objective ($L^{\text{DER++}}$):

$$\begin{aligned} L_t^{\text{DER}}(\theta) &= E_{(x,y)\sim D_t^{\text{train}}}\left[L^{ce}(f_\theta(x), y)\right] \\ &\quad + \alpha E_{(x,y,z)\sim \mathcal{M}_t}\left[||f_\theta(x) - z||_2^2\right], \end{aligned} \tag{3}$$

$$L_t^{\text{DER++}}(\theta) = L_t^{\text{DER}} + \beta E_{(x,y,z)\sim \mathcal{M}_t}\left[L^{ce}(f_\theta(x), y)\right]. \tag{4}$$

2.2. Meta-learning

The goal of meta-learning [45] is to explicitly learn components influencing the learning process itself to be more effective. The popular MAML [13] algorithm entails fixing the optimization procedure and learning a set of initial parameters θ that perform well across the task set $\mathcal{T}$. In particular, MAML updates θ based on a meta *outer* loss L^{out} after performing gradient descent for a sequence of *inner steps* to optimize a local *inner* loss L^{inn} starting from $\theta_0 = \theta$:

$$L^{\text{MAML}}(\theta) = E_{(D, D^{\text{train}})\sim \mathcal{T}}\left[L^{out}(\theta_k|D)\right], \tag{5}$$

$$\textit{where} \quad \theta_k = \theta_{k-1} - \eta \frac{\partial L^{inn}(\theta_{k-1}|D^{\text{train}})}{\partial \theta_{k-1}}. \tag{6}$$

In Equation 5, L^{inn} can be any differentiable objective computed from sampled sequences $(x_i, y_i)_{i=1}^k$ from the training data in D^{train}[2]. MAML was designed to solve multi-task few-shot classification problems where L^{inn} and L^{out} are cross-entropy losses and D^{train} is a very small buffer of samples. However, computational costs and gradient instabilities become significantly more relevant with increasing k, as meta-optimization requires backpropagating through the whole computation graph performing the inner optimization steps [3]. Hence, k is usually set to some particularly low value, which is not an issue in few-shot learning scenarios since performing a larger number of i.i.d. gradient steps should not expectedly hurt performance.

[2]This dependency was omitted in Eqn. 5 to avoid clutter in the notation.

2.3. Meta-learning for continual learning overview

Meta-learning approaches for continual learning can be divided into two categories based on their relative problem settings and assumptions [7]. In what we will refer to as *meta pre-trained continual learning* the model has access to a separate meta pre-training phase (meta-training) that happens before deployment to a continual learning problem (meta-testing). During meta pre-training, we assume the model has full access to a large number of *meta-training tasks*, sampled from the same task distribution we expect to encounter during meta-testing. Hence, the model can meta-learn how to effectively continually learn with optimization procedures very much analogous to the ones employed in traditional meta-learning designed for the i.i.d. setting [4, 23]. Instead, the second category of methods tackles the more general *continual learning with meta-learning* problem setting, without assuming access to an auxiliary set of meta-training tasks. These methods optimize an outer meta-loss to replace or complement traditional continual learning objectives, making use of memory buffer strategies to obtain i.i.d. distribution of samples [17, 41]. SiM4C can be effectively implemented in *both* problem settings, as we will describe in greater detail throughout the next sections.

3. SiM4C design and motivation

3.1. Meta pre-trained continual learning

Online-aware meta-learning (OML) [23] and neuromodulated meta-learning (ANML) [4] are popular approaches tackling the aforementioned meta pre-trained continual learning setting for image classification. They consider performing k fine-tuning steps on a subset of the network's weights θ_p (prediction weights) using the non-stationary continual learning online data stream as the inner steps in Equation 5, while the rest of the weights θ_r (representation weights) are kept fixed. Their proposed meta-optimization procedure then maximizes performance with respect to the whole set of the classifier's parameters $\theta = \theta_r \cup \theta_p$ using a mini-batch estimate of the continual learning objective in Equation 7 as the outer loss. This estimate is made by a combination of the correlated online samples used during the inner steps and an additional set of i.i.d. data from all tasks called the remember set D^{REM}. The purpose of this objective is to find initial weights θ that incentivize *plasticity* on the inner task data and also alleviate catastrophic forgetting on the i.i.d. distribution of samples from D^{REM}.

Online meta continual learning with Omniglot. OML and ANML perform the described meta-optimization as part of their meta pre-training phase. During meta-testing, the prediction weights θ_p are then continually finetuned with a standard cross-entropy objective on a sequence of test tasks while the rest of the weights of f_θ are instead frozen. The main benchmark considered in both works is based on the Omniglot dataset [28], consisting of 1,623 classes, each treated as a separate task. The classes are split into 963 meta-training tasks $\mathcal{T}^{\text{train}}$ and 660 meta-testing classes $\mathcal{T}^{\text{test}}$. During meta-testing, this benchmark follows the online continual learning problem statement with the training data for the t^{th} task in $\mathcal{T}^{\text{test}}$, denoted D_t^{train}, consisting of $n_t = 15$ training samples from its corresponding class. Hence, f_θ can access each data point only a single time before moving to the next task. Continual learning performance is then evaluated on 5 held-out samples from each distribution of the previously seen classes $D_{1:t}$ in $\mathcal{T}^{\text{test}}$. To tackle this problem setting, OML and ANML meta pre-train by performing $k \leq 20$ online inner updates with data from a randomly sampled task from $\mathcal{T}^{\text{train}}$, producing $\theta^k = \theta_r \cup \theta_p^k$. Then, they compute an estimate of the outer meta-loss by additionally sampling a remember set D^{REM} from all samples in the 963 $\mathcal{T}^{\text{train}}$ tasks:

$$L^{\text{OML}}(\theta) = E_{D^{\text{train}}, D^{\text{REM}}} \left[\sum_{(x,y) \in D^{\text{train}} \cup D^{\text{REM}}} L^{ce}(f_{\theta^k}(x), y) \right]. \tag{7}$$

Limitations and approximations. OML and ANML have been solely designed for the meta pre-trained continual learning setting which assumes unrestricted access to a large number of meta-training tasks. Hence, they cannot be directly applied to problem settings that always impose the canonical restrictions of continual learning. Furthermore, at meta-testing time, their meta pre-trained classifiers need to retain the knowledge of each task D_t after observing its n_t samples for all subsequent $T - t$ tasks. To optimally optimize for retention and plasticity for the t^{th} task, we would ideally embed the full $\sum_t^T n_t$ steps in the meta inner loop. However, even just using as many as 15/20 steps (the number of steps in OML/ANML) can lead to great computational costs and optimization issues. These practical issues are caused by the repeated forward and backward passes through the same layers of a neural network, requiring to store all intermediate activations at each step [3, 13, 39]. In fact, by inspecting all the shared repositories of the aforementioned algorithms, we see they *do not actually compute* the exact second-order meta gradients but rather a first-order approximation known as first-order MAML (FOMAML). However, Antoniou et al. [3] showed that while utilizing FOMAML leads to lower instabilities and slightly faster computation, it comes at the cost of model performance due to introducing further considerable approximation errors.

3.2. A very simple yet challenging meta-objective

In our work, we do not try to align the inner and continual objectives by increasing k or by meta-optimizing additional models and parameters. While these directions lead to performance gains in previous work trying to improve OML [4, 17], the costs and instabilities of current meta-learning approaches constrain their actual scalability and introduce the need for limiting approximations (such as FOMAML). Furthermore, we argue that increasing complexity often confounds actual algorithmic improvements, making adoption harder for practitioners and the rest of the research community. Instead, our technical contribution goes in the opposite direction as we focus on designing a *minimal, efficient, and practical* objective to meta-learn how to effectively tackle arbitrary continual learning problems. Our designed simplification entails emphasizing forward transfer and generalization in the meta-objective and allows foregoing hindering approximations to the meta-gradient.

A principle guiding our design motivation comes from noting that optimizing for a lower number of inner optimization steps k does not necessarily entail an easier continual optimization problem. Most prior algorithms focused on explicitly reducing catastrophic forgetting (performance from past tasks in D^{REM}) and incentivizing plasticity (performance on training data D^{train}) for each task independently. However, they do not consider the effects of k on the difficulty of mitigating forgetting the current task *across future tasks* and its implications to incentivize *forward transfer*. In addition to minimizing costs and allowing tractable second-order optimization, minimizing the number of inner

Algorithm 1 Meta Pre-Trained Online Continual Learning Prior meta-learners (*orange*) compared to SiM4C (*green*)

```
input: $\eta$, learning rate; S, pre-training steps; $k$, inner steps
META (PRE-)TRAINING PHASE
Initialize $\theta \leftarrow \theta_r \cup \theta_p$
for $s \leftarrow 1, 2, \dots S$ do
    Sample remember set $D^{\text{REM}}$ from all tasks in $\mathcal{T}^{\text{train}}$
    Sample task $D^{\text{train}} \in \mathcal{T}^{\text{train}}$
    Sample $(x_{1:k}, y_{1:k}) \subset D^{\text{train}}$
    $\theta^0 \leftarrow \theta$
    for $i \leftarrow 1, 2, \dots k$ do
        $\theta_r^i \leftarrow \theta_r^{i-1} - \eta \frac{\partial L^{ce}(f_{\theta^{i-1}}(x_i), y_i)}{\partial \theta_p^{i-1}}$
        $\theta^i \leftarrow \theta_r \cup \theta_r^i$
    $D^{\text{OUT}} \leftarrow (x_{1:k}, y_{1:k}) \cup D^{\text{REM}}$
    $\theta \leftarrow \theta - \eta \sum_{(x,y) \in D^{\text{OUT}}} \frac{\partial L^{ce}(f_{\theta^k}(x), y)}{\partial \theta^k}$    ▷ first-order approx.
    Sample $(x, y) \in D^{\text{train}}$
    $\theta'_r \leftarrow \theta - \eta \times \frac{\partial L^{ce}(f_\theta(x), y)}{\partial \theta_p}$
    $\theta' \leftarrow \theta_r \cup \theta'_r$
    $D^{\text{OUT}} \leftarrow D^{\text{train}} \cup D^{\text{REM}}$
    $\theta \leftarrow \theta - \eta \sum_{(x,y) \in D^{\text{OUT}}} \frac{\partial L^{ce}(f_{\theta'}(x), y)}{\partial \theta}$    ▷ exact gradient
META-TESTING TRAINING PHASE
for $t \leftarrow 1, 2, \dots T$ do
    Retrieve $D_t^{\text{train}} \in \mathcal{T}^{\text{test}}$
    for $i \leftarrow 1, 2, \dots n$ do
        Retrieve $(x_i, y_i) \in D_t^{\text{train}}$
        $\theta_p \leftarrow \theta_p - \eta \frac{\partial L^{ce}(f_\theta(x_i), y_i)}{\partial \theta_p}$
output: $f_\theta$    ▷ return final model
```

optimization steps will maximize the generalization challenge of the outer meta-loss with respect to the remaining unseen data. We hypothesize this harder objective might force the model to also effectively re-use old task information when optimizing the new task, promoting cross-task *forward transfer*. Moreover, it might help meta-learn models that generalize to test-time tasks that could also provide lower information content, counteracting overfitting to the meta-pre-training tasks. Minimizing k also allows to utilize all the larger set of remaining unseen data in D^{train} to increase diversity and reduce variance in the meta-objective, which might further assist with training stability.

Based on our observations about the limitations of prior work and the role of k, we propose a new meta learning optimization strategy with three key properties: **(i) Fixing the number of inner steps to $\mathbf{k = 1}$**, the minimum possible value. **(ii) Using exact second-order meta-gradients**, which can now be tractably computed due to the shallower computation graph. **(iii) Having numerous unseen task samples in the meta-loss**, significantly incentivizing forward transfer, together with plasticity and backward transfer. We remark that these changes minimize the amount of available unseen data per task and should make the overall meta-optimization process very fast and stable to execute, even when computing the exact second-order gradients. We note that placing emphasis on forward transfer brings our objective significantly closer to the original few-shot generalization objective that motivated the design decisions of MAML. We provide a summary highlighting the differences of our **si**mple **m**eta-learner for **c**ontinual learning (SiM4C) in Algorithm 1. In Section 4, we empirically evaluate our algorithm's performance and theorized properties on the Omniglot benchmark against the aforementioned meta pre-trained online continual learning methods. Then, in Section 5, we propose to utilize the SiM4C meta-loss with memory buffer data as an auxiliary objective for continual learning without any meta pre-training, evaluate on a wide range of benchmarks that include Class-IL, Task-IL, and Domain-IL problems. In both settings, SiM4C attains near-universal improvements and state-of-the-art performance without any ad hoc hyper-parameter tuning.

4. Meta-pretrained continual evaluation

We evaluate SiM4C on the meta pre-trained online continual learning setting using the Omniglot dataset [28]. As described in Section 3.1, we meta pre-train on the first 963 classes and meta-test on sampled sequences of 600 tasks. Each class corresponds to a separate task where the model has access to each datapoint only once, following the main experimental setup from the considered baselines. We compare the performance of SiM4C against OML [23] and ANML [4]. For OML, we consider two versions utilizing the authors' shared best meta pre-trained models, where the second *improved* model was uploaded long after their publication, employing additional optimization tricks and a similar network architecture to ANML. We keep SiM4C's main network architecture and hyper-parameters consistent with ANML to ensure a fair comparison. However, ANML still uses an auxiliary neuromodulation network, effectively doubling its total weights. The Omniglot benchmark greatly emphasizes the ability of meta pre-training to recover effective models that defy forgetting, as simple finetuning approaches do not exceed random accuracy when learning from even just 5 consecutive tasks. In the reported performance curves we average over five distinct models and five evaluation runs, each obtained with different random seeds. Shaded regions and error bars correspond to standard deviations. We refer to Appendix B for all further details regarding our baselines, training procedure, and models.

Performance analysis. As shown in Figure 2A, SiM4C greatly outperforms all other considered algorithms, setting new state-of-the-art results on the Omniglot benchmark. After observing 600 meta-testing tasks, SiM4C achieves a final mean accuracy on unseen samples exceeding 71%. In contrast, the best collected mean final accuracies of ANML (62%) and OML (56%) closely match (and even surpass) the ones reported in the respective paper, lagging behind SiM4C by a considerable margin. Remarkably, SiM4C achieves these results with a much simpler and more efficient implementation, removing unnecessary complexity, computation, and hyper-parameters. These advantages are reflected in Figure 3A, showing the recorded time taken and GPU memory allocation from running the respective meta-optimization loops. In particular, we find that SiM4C trains around four times faster than ANML (five times for OML), while having also the lowest GPU memory requirements.

Understanding the performance gap. We analyze the key factors making SiM4C's simpler and lighter meta-optimization considerably more effective than prior work. By comparing Figure 2A and Figure 2B, we observe that SiM4C displays a much stronger generalization ability as compared to ANML, attaining a lower gap between its test and training accuracy throughout meta-testing. This improved generalization appears to validate the effectiveness of SiM4C's meta pre-training objective, emphasizing *forward* transfer to unseen task data. In comparison, ANML's meta-objective solely focuses on plasticity (on the k inner examples) and backward transfer (on the remember set), reflecting its high knowledge retention ability.

However, we note that despite its objective's focus on maximizing performance retention on seen data, ANML's training accuracy ultimately still lags behind SiM4C's. To understand the reasons causing this, we record and analyze the final meta-testing performances of SiM4C and ANML from frequently saving and evaluating all meta pre-trained models stored every 5000 meta pre-training steps.

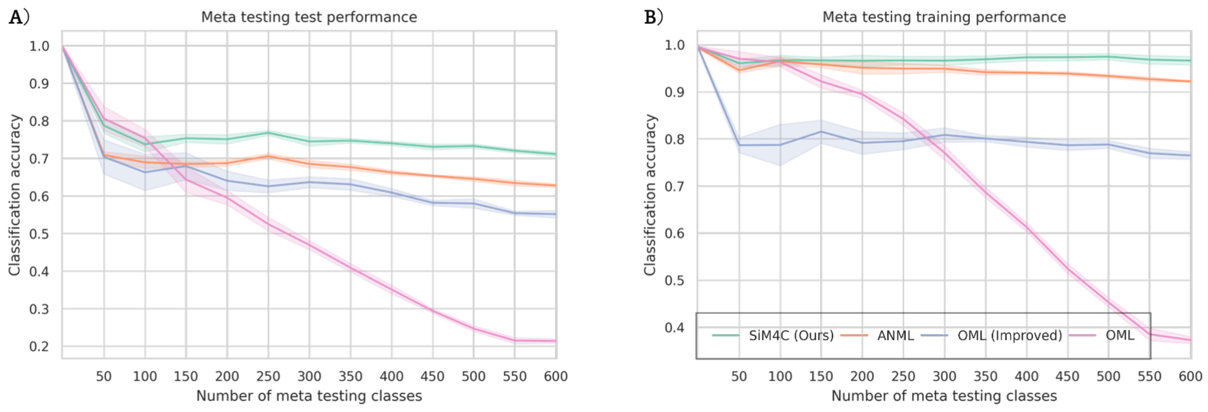

Figure 2. Performance on the online meta continual Omniglot benchmark *after meta pre-training*. We compare classification accuracy on **(A) unseen test data** and **(B) seen training data** of all experienced tasks from $\mathcal{T}^{\text{test}}$ as we perform meta-testing.

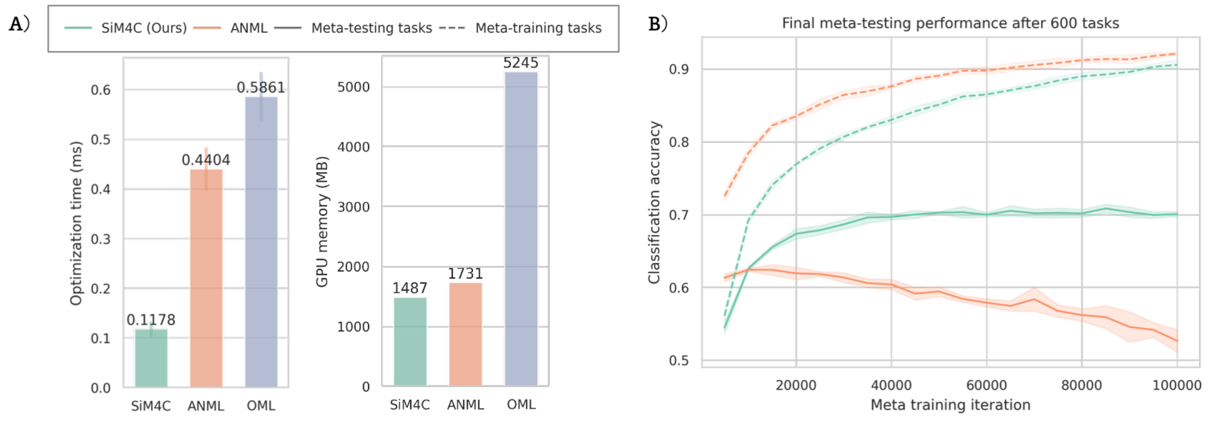

Figure 3. Comparison of the properties between SiM4C and prior methods: **(A) Average time taken and GPU memory required to perform meta-optimization**. **(B) final meta-testing performance as a function of the number of meta pre-training steps.**

As shown in Figure 3B, with pre-training progressing we observe a stark contrast between how the meta-testing performance evolves. In particular, SiM4C's final accuracy rises and stably settles close to its apex while ANML's performance starts visibly deteriorating after the first 20K pre-training steps. We compare these results with the final accuracies collected by emulating meta-testing using tasks *already seen during pre-training*, displayed by the dotted curves. In this alternative scenario, ANML's final accuracy grows monotonically as we increase the length of the meta pre-training phase, even surpassing SiM4C's. These findings are a clear indication of *meta-overfitting*, which suggests that ANML's meta pre-training procedure struggles to defy forgetting beyond the precise distribution of meta-training tasks. In contrast, they highlight how utilizing SiM4C's purposefully regularized objective that minimizes the ratio between seen and unseen task samples, and computing the exact second-order gradients allows to meta-learn a more *general* strategy to continually learn. They also show that SiM4C is more robust to the meta pre-training procedure, as evidenced by its lower performance dependence on appropriate early stopping. We refer to Appendix D for ablation studies and additional results analyzing the contributions of SiM4C's individual components.

5. SiM4C without meta pre-training

Integration with memory-based algorithms. We port SiM4C to more canonical continual learning settings, lifting the assumptions of having access to a meta pre-training phase. Here, we propose to employ SiM4C as an auxiliary objective during each continual learning step, integrating it with existing memory-based algorithms. Our implementation tries to make minimal changes from its base approach, such that it can be included as a simple plug-in addition and extended to future continual learning algorithms. As detailed in Section 2.1, memory-based continual learning approaches follow a general structure of optimizing a weighted sum between a main cross-entropy loss L^{ce} using a batch of data from the *current* task $D^{\text{curr}} \subset D_t^{\text{train}}$ and an auxiliary loss L^{aux} using a batch of *past* data from the

Algorithm 2 Continual Learning with SiM4C (*green*)

```
input: $\eta$, learning rate; S, task training steps; $\alpha$, memory coefficient
Initialize $\theta$
for $t \leftarrow 1, 2, \dots T$ do
    Retrieve $D_t^{\text{train}} \in \mathcal{T}$
    for $i \leftarrow 1, 2, \dots S$ do
        Sample batches $D^{\text{curr}}$, $D^{\text{fut}} \sim D_t^{\text{train}}$
        Sample batch $D^{\text{past}} \sim \mathcal{M}$
        $l_\theta^{ce} \leftarrow \sum_{(x,y)\in D^{\text{curr}}} L^{ce}(f_\theta(x), y)$    ▷ memory-based losses
        $l_\theta^{aux} \leftarrow \alpha \times \sum_{(x,y)\in D^{\text{past}}} L^{aux}(f_\theta(x), y)$
        $\theta' \leftarrow \theta - \eta \frac{\partial l_\theta^{ce}}{\partial \theta}$    ▷ SiM4C inner step
        $l_{\theta'}^{ce} \leftarrow \sum_{(x,y)\in D^{\text{fut}}} L^{ce}(f_{\theta'}(x), y)$    ▷ meta losses
        $l_{\theta'}^{aux} \leftarrow \alpha \times \sum_{(x,y)\in D^{\text{past}}} L^{aux}(f_{\theta'}(x), y)$
        $\theta \leftarrow \theta - \eta \frac{\partial (l_\theta^{ce} + l_\theta^{aux} + l_{\theta'}^{ce} + l_{\theta'}^{aux})}{\partial \theta}$    ▷ update model
        $\mathcal{M} \leftarrow reservoir(\mathcal{M}, D^{\text{curr}})$    ▷ update memory
output: $f_\theta$    ▷ return final model
```

memory buffer $D^{\text{past}} \subset \mathcal{M}$. Together with the current and past data batches (D^{curr}, D^{past}), the SiM4C auxiliary loss relies on what we refer to as a future batch D^{fut} and involves two simple steps based on SiM4C's optimization procedure:

1. Performing a single inner optimization step on D^{curr}.
2. Calculating the learner's new loss on D^{fut} and D^{past}.

We find several different choices for obtaining D^{fut} work well in practice. For instance, a natural choice is to temporarily store each sampled batch data of the current task for two optimization steps rather than one. Hence, we can use the earlier collected batch as D^{curr} and the later collected batch as D^{fut}. However, we note this change introduces minimal constant memory costs from storing the extra batch that could make evaluation unfair with the other memory-based algorithms. Therefore, we resort to an even simpler solution that we find works almost as well. Given that modern implementation of all the considered continual learning baselines entail some form of data augmentation that tries to capture within-task variations, we augment the sampled batch from the current task an additional time to act as D^{fut}. We refer to our Appendix E for additional discussion and empirical comparisons of different viable approaches to obtain D^{fut}. We add the SiM4C auxiliary loss without any tuned scaling coefficient to avoid introducing any additional hyper-parameter. Due to its simplicity, integrating SiM4C requires very few additional lines of code as exemplified in Algorithm 2, where we highlight the changes from a general memory-based continual learning implementation.

Class-, Task-, and Domain-IL settings. We integrate SiM4C with the state-of-the-art DER and DER++ algorithms [6] introduced in Section 2.1, together with a more traditional memory-based baseline from Riemer et al. [41] (ER) using a simple cross-entropy loss as its auxiliary loss for the memory data. We compare final test accuracy also with additional memory-based continual learning algorithms [2, 5, 8, 9, 35, 40]. We consider several established continual learning benchmarks for image classification. For Class-IL and Task-IL, **S-CIFAR-10** and **S-Tiny-ImageNet** are constructed by splitting the classes of CIFAR-10 [26] and of Tiny-ImageNet [29] in 5 and 10 mutually-exclusive classification tasks, following [6, 11, 55]. On these benchmarks, all methods employ a ResNet18 architecture [19] and train for 50 and 100 epochs, respectively. For Domain-IL, **Permuted MNIST** [25] and **Rotated MNIST** [35] involve 20 total tasks, each applying a different random permutation or rotation in $[0, \pi)$ to the images from MNIST [30]. On these benchmarks, all methods employ a simple 2-layer fully-connected network with 100 hidden units and train for a single epoch. We provide the baseline performances reported by Buzzega et al. [6], obtained after tuning hyper-parameters for each different continual dataset and memory buffer size. However, we do not re-tune any hyper-parameter of the SiM4C integrations to evaluate its general applicability. We refer to Appendix B for all further details.

Performance analysis. As shown in Table 1, integrating SiM4C leads to near-universal benefits and *state-of-the-art* performance, with recorded final accuracy gains in 35/36 experiments. Remarkably, using SiM4C appears to make even a simple memory-based baseline (ER) competitive with the much more complex DER++ algorithm, outperforming it in multiple settings. SiM4C's benefits are particularly evident for the more challenging problem settings that preclude task information (Class-IL) or involve an increased number of complex samples (S-Tiny-ImageNet). Intuitively, for these continual learning problems, interference is more likely to occur, validating the impact of SiM4C's explicit meta-optimization objective. For the comparatively easier Domain-IL settings on the low-dimensional MNIST dataset, we still recorded performance benefits in 11/12 experiments, however, their magnitude was more limited. We believe this contrast is also due to the fact that continual learning performance on these benchmarks is already quite close to joint training, leaving less overall room for improvement. In Appendix C, we also show that SiM4C adds only minimal computational overheads and we provide additional detailed results recording additional metrics and considering additional buffer sizes and baselines.

Considerations. Since, most of the prior memory-based continual learning methods make use of some fixed heuristic procedure to approximate the true *i.i.d.* data loss using the memory buffer data, we believe they inevitably require some level of hyper-parameter tuning to account for problem-specific levels of non-stationarity and other factors affecting their training dynamics. In principle, by adding SiM4C as an auxiliary step, we synergistically make the relative continual learning algorithm explicitly aware of the

Buffer	Method	S-CIFAR-10		S-Tiny-ImageNet		P-MNIST	R-MNIST
		Class-IL	*Task-IL*	*Class-IL*	*Task-IL*	*Domain-IL*	*Domain-IL*
✗	JOINT	92.20	98.31	59.99	82.04	94.33	95.76
	SGD	19.62	61.02	7.92	18.31	40.70	67.66
	ER [41]	44.79	91.19	8.49	38.17	72.37	85.01
(Ours)	**ER with SiM4C**	54.77 +22.3%	**92.41** +1.3%	8.88 +4.5%	41.76 +9.4%	72.56 +0.3%	85.95 +1.1%
	GEM [35]	25.54	90.44	-	-	66.93	80.80
	A-GEM [8]	20.04	83.88	8.07	22.77	66.42	81.91
	iCaRL [40]	49.02	88.99	7.53	28.19	-	-
	FDR [5]	30.91	91.01	8.70	40.36	74.77	85.22
200	GSS [2]	39.07	88.80	-	-	63.72	79.50
	HAL [9]	32.36	82.51	-	-	74.15	84.02
	DER [6]	61.93	91.40	11.87	40.22	81.74	90.04
(Ours)	**DER with SiM4C**	63.81 +3.0%	91.86 +0.5%	13.27 +11.8%	42.53 +5.8%	81.83 +0.1%	90.64 +0.7%
	DER++ [6]	64.88	91.92	10.96	40.87	**83.58**	90.43
(Ours)	**DER++ with SiM4C**	**65.93** +1.6%	92.22 +0.3%	**13.31** +21.5%	**43.43** +6.3%	82.96 −0.7%	**91.57** +1.3%
	ER [41]	57.74	93.61	9.99	48.64	80.60	88.91
(Ours)	**ER with SiM4C**	66.05 +14.4%	94.37 +0.8%	10.83 +8.4%	52.10 +7.1%	80.65 +0.1%	89.39 +0.5%
	GEM [35]	26.20	92.16	-	-	76.88	81.15
	A-GEM [8]	22.67	89.48	8.06	25.33	67.56	80.31
	iCaRL [40]	47.55	88.22	9.38	31.55	-	-
	FDR [5]	28.71	93.29	10.54	49.88	83.18	89.67
500	GSS [2]	49.73	91.02	-	-	76.00	81.58
	HAL [9]	41.79	84.54	-	-	80.13	85.00
	DER [6]	70.51	93.40	17.75	51.78	87.29	92.24
(Ours)	**DER with SiM4C**	72.62 +3.0%	93.94 +0.6%	18.57 +4.6%	**53.68** +3.7%	87.41 +0.1%	92.63 +0.4%
	DER++ [6]	72.70	93.88	19.38	51.91	88.21	92.77
(Ours)	**DER++ with SiM4C**	**74.77** +2.8%	**94.43** +0.6%	**20.61** +6.3%	52.37 +0.9%	**88.23** +0.0%	**92.99** +0.2%

Table 1. Mean final classification accuracy and relative improvements (*green*) from using SiM4C with different memory-based algorithms, as evaluated on popular continual learning benchmarks. We provide the baseline results reported in prior work. Empty entries indicate either incompatibility with the relative problem setting (iCaRL in Domain-IL) or intractable training times [6].

problem-specific training dynamics (within SiM4C's inner loop), enabling meta-learning to affect them (in SiM4C outer optimization). This intuitive property is reflected by how SiM4C with a single set of hyper-parameters. appears to consistently outperform all DER-based methods using task-specific tuned hyper-parameters. Overall, we believe our experiments strongly validate SiM4C's effectiveness beyond the meta pre-trained setting, which we hope will have implications for the future design and adoption of lighter and more general meta-learning methods for continual learning.

6. Related work

The canonical objective of continual learning has been to mitigate catastrophic forgetting [12, 14, 15, 31]. Currently, the more general and best-performing methods are replay-based, in line with recent results showing that it is hard to avoid forgetting in class-incremental settings without a memory buffer [32, 49]. However, consistently with our motivation, several other properties have been recognized as desirable for a continual learning system, such as forward and backward transfer, fast adaptation, and computational efficiency [12]. Furthermore, while several recent methods have been proposed for online continual learning [1, 6, 10], scaling to large numbers of tasks in this setting currently appears infeasible without some form of pre-training [23].

Most meta-learning methods for continual learning optimize for fast remembering [20] or slow forgetting [23]. Variations of these approaches include methods that explicitly incentivize *sparsity* to minimize interference. For instance, ANML [4] meta-learns a separate Neuromodulatory network, gating the forward and backward passes, while sparse-MAML [51] focuses on meta-learning *where to learn*, inducing sparse learning updates. MER [41] proposed an algorithm to specifically incentivize gradient alignment based on Reptile [37] and is compatible with continual meta-learning without pre-training by using an experience replay, like SiM4C. Gupta et al. [17] extended this work by partially addressing MER's high compute costs and extending the meta-learned parameters to include a set of per-parameter learning rates [17]. In contrast to SiM4C, all this work entailed optimizing small variations on OML's objective, mostly focusing on plasticity and knowledge retention rather than forward transfer. Finally, another related class of meta-learning methods is instead based on dataset distillation [52, 54], with the goal of optimizing the memory buffer content to increase the number of samples stored while keeping a low memory footprint [42, 44, 53, 56].

7. Conclusion

This work introduced SiM4C: a novel, simple, and generally applicable meta-learning algorithm to tackle continual learning. Unlike prior related approaches, SiM4C purposefully minimizes the number of inner optimization updates to allow for practically specifying challenging forward and backward transfer objectives in its outer meta-loss by making use of larger amounts of unseen task data. Even without first-order gradient approximations, we show our design leads to better stability, computational efficiency, and robustness to meta-overfitting. Furthermore, SiM4C can both make use of a meta pre-training phase or be integrated out-of-the-box with any memory-based continual learning method in a few lines of code. Empirically, its application produces near-universal performance gains and achieves state-of-the-art results across five different continual image classification benchmarks. In the spirit of our work, we hope that future meta-learning methods will also be designed for applicability and efficiency, to have concrete implications for *general* continual learning challenges.

Acknowledgments

Edoardo Cetin would like to acknowledge the support from the Engineering and Physical Sciences Research Council [EP/R513064/1]. Oya Celiktutan would also like to acknowledge the support from the LISI Project, funded by the Engineering and Physical Sciences Research Council [EP/V010875/1]. Furthermore, we thank Toyota Motor Europe and Toyota Motor Corporation for providing support towards funding some of the utilized computational resources.

References

[1] R. Aljundi, K. Kelchtermans, and T. Tuytelaars. Task-Free Continual Learning. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11246–11255, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.01151. 8

[2] R. Aljundi, M. Lin, B. Goujaud, and Y. Bengio. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, 2019. 7, 8, 17, 21

[3] A. Antoniou, H. Edwards, and A. Storkey. How to train your maml. In *Seventh International Conference on Learning Representations*, 2019. 1, 3, 4, 19

[4] S. Beaulieu, L. Frati, T. Miconi, J. Lehman, K. O. Stanley, J. Clune, and N. Cheney. Learning to continually learn. In *ECAI 2020*, pages 992–1001. IOS Press, 2020. 1, 2, 3, 4, 5, 8, 13, 14, 18

[5] A. S. Benjamin, D. Rolnick, and K. P. Kording. Measuring and regularizing networks in function space. *International Conference on Learning Representations*, 2019. 7, 8, 17

[6] P. Buzzega, M. Boschini, A. Porrello, D. Abati, and S. Calderara. Dark experience for general continual learning: a strong, simple baseline. *Advances in neural information processing systems*, 33:15920–15930, 2020. 2, 3, 7, 8, 15, 16, 17, 20, 21

[7] M. Caccia, P. Rodriguez, O. Ostapenko, F. Normandin, M. Lin, L. Page-Caccia, I. H. Laradji, I. Rish, A. Lacoste, D. Vázquez, and L. Charlin. Online Fast Adaptation and Knowledge Accumulation (OSAKA): A New Approach to Continual Learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 16532–16545. Curran Associates, Inc., 2020. 1, 3

[8] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny. Efficient lifelong learning with a-gem. In *International Conference on Learning Representations*, 2019. 7, 8, 17

[9] A. Chaudhry, A. Gordo, P. K. Dokania, P. Torr, and D. Lopez-Paz. Using hindsight to anchor past knowledge in continual learning. *arXiv preprint arXiv:2002.08165*, 2020. 7, 8, 17

[10] M. De Lange and T. Tuytelaars. Continual Prototype Evolution: Learning Online From Non-Stationary Data Streams. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8250–8259, 2021. 8

[11] M. De Lange, R. Aljundi, M. Masana, S. Parisot, X. Jia, A. Leonardis, G. Slabaugh, and T. Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021. 1, 7, 15

[12] N. Díaz-Rodríguez, V. Lomonaco, D. Filliat, and D. Maltoni. Don't forget, there is more than forgetting: New metrics for Continual Learning. (Nips), 2018. 1, 8

[13] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017. 3, 4, 12, 14

[14] R. M. French. Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks. In *Proceedings of the 13th annual cognitive science society conference*, volume 1, pages 173–178, 1991. 1, 8, 13

[15] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, Apr. 1999. ISSN 1364-6613. doi: 10.1016/S1364-6613(99)01294-2. 1, 8

[16] S. T. Grossberg. *Studies of mind and brain: Neural principles of learning, perception, development, cognition, and motor control*, volume 70. Springer Science & Business Media, 2012. 1

[17] G. Gupta, K. Yadav, and L. Paull. Look-ahead meta learning for continual learning. *Advances in Neural Information*

Processing Systems, 33:11588–11598, 2020. 1, 3, 4, 8, 18

[18] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu. Embracing change: Continual learning in deep neural networks. *Trends in cognitive sciences*, 24(12):1028–1040, 2020. 1

[19] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7, 15, 21

[20] X. He, J. Sygnowski, A. Galashov, A. A. Rusu, Y. W. Teh, and R. Pascanu. Task Agnostic Continual Learning via Meta Learning. *arXiv:1906.05201 [cs, stat]*, June 2019. 8

[21] G. Hinton, O. Vinyals, and J. Dean. Dark knowledge. *Presented as the keynote in BayLearn*, 2(2), 2014. 3

[22] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015. 16

[23] K. Javed and M. White. Meta-learning representations for continual learning. *Advances in neural information processing systems*, 32, 2019. 1, 2, 3, 5, 8, 13, 14, 18

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 14

[25] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. 7

[26] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7

[27] D. Kudithipudi, M. Aguilar-Simon, J. Babb, M. Bazhenov, D. Blackiston, J. Bongard, A. P. Brna, S. Chakravarthi Raja, N. Cheney, J. Clune, et al. Biological underpinnings for lifelong learning machines. *Nature Machine Intelligence*, 4(3): 196–210, 2022. 1

[28] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015. 4, 5

[29] Y. Le and X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 7

[30] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7

[31] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez. Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges. *Information Fusion*, 58:52–68, June 2020. ISSN 1566-2535. doi: 10.1016/j.inffus.2019.12.004. 1, 8

[32] T. Lesort, A. Stoian, and D. Filliat. Regularization Shortcomings for Continual Learning. *arXiv:1912.03049 [cs, stat]*, Feb. 2020. 8

[33] Z. Li and D. Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12), 2017. 16, 17

[34] V. Liu, R. Kumaraswamy, L. Le, and M. White. The utility of sparse representations for control in reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4384–4391, 2019. 13

[35] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems*, 2017. 7, 8, 12, 17

[36] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989. 1

[37] A. Nichol, J. Achiam, and J. Schulman. On First-Order Meta-Learning Algorithms, Oct. 2018. 8

[38] A. Nichol, J. Achiam, and J. Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018. 1, 12

[39] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine. Meta-learning with implicit gradients. *Advances in neural information processing systems*, 32, 2019. 4

[40] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017. 7, 8, 17

[41] M. Riemer, I. Cases, R. Ajemian, M. Liu, I. Rish, Y. Tu, and G. Tesauro. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 7, 8, 12, 17, 18, 20, 21

[42] A. Rosasco, A. Carta, A. Cossu, V. Lomonaco, and D. Bacciu. Distilled Replay: Overcoming Forgetting Through Synthetic Samples. In F. Cuzzolin, K. Cannons, and V. Lomonaco, editors, *Continual Semi-Supervised Learning*, volume 13418, pages 104–117. Springer International Publishing, Cham, 2022. ISBN 978-3-031-17586-2 978-3-031-17587-9. doi: 10.1007/978-3-031-17587-9_8. 9

[43] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell. Progressive neural networks. *arXiv preprint arXiv:1606.04671*, 2016. 16, 17

[44] M. Sangermano, A. Carta, A. Cossu, and D. Bacciu. Sample Condensation in Online Continual Learning. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 01–08, July 2022. doi: 10.1109/IJCNN55064.2022.9892299. 9

[45] J. Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987. 3

[46] J. Schwarz, W. Czarnecki, J. Luketina, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning*, 2018. 16, 17

[47] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 14

[48] G. M. van de Ven and A. S. Tolias. Three scenarios for continual learning. *Arxiv preprint*, Apr. 2019. doi: 10.48550/arXiv.1904.07734. 1, 2

[49] G. M. van de Ven, H. T. Siegelmann, and A. S. Tolias. Brain-inspired replay for continual learning with artificial neural networks. *Nature Communications*, 11(1):4069, Aug. 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-17866-2. 8

[50] J. S. Vitter. Random sampling with a reservoir. *ACM Transactions on Mathematical Software (TOMS)*, 11(1):37–57, 1985. 2

[51] J. von Oswald, D. Zhao, S. Kobayashi, S. Schug, M. Caccia, N. Zucchet, and J. Sacramento. Learning where to learn: Gradient sparsity in meta and continual learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 5250–5263. Curran Associates, Inc., 2021. 8

[52] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros. Dataset Distillation. *arXiv:1811.10959 [cs, stat]*, Feb. 2020. 9

[53] F. Wiewel and B. Yang. Condensed Composite Memory Continual Learning. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2021. doi: 10.1109/IJCNN52387.2021.9533491. 9

[54] R. Yu, S. Liu, and X. Wang. Dataset Distillation: A Comprehensive Review, Jan. 2023. 9

[55] F. Zenke, B. Poole, and S. Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017. 7, 16, 17

[56] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset Condensation with Gradient Matching. In *International Conference on Learning Representations*, Sept. 2020. 9