ECAI 2020
G.D. Giacomo et al. (Eds.)

doi:10.3233/FAIA200193

Learning to Continually Learn

Shawn Beaulieu[1] and **Lapo Frati**[1] and **Thomas Miconi**[2] and **Joel Lehman**[2] and **Kenneth O. Stanley**[2] and **Jeff Clune***[2,3] and **Nick Cheney***[1]

Abstract. Continual lifelong learning requires an agent or model to learn many sequentially ordered tasks, building on previous knowledge without catastrophically forgetting it. Much work has gone towards preventing the default tendency of machine learning models to catastrophically forget, yet virtually all such work involves manually-designed solutions to the problem. We instead advocate meta-learning a solution to catastrophic forgetting, allowing AI to learn to continually learn. Inspired by neuromodulatory processes in the brain, we propose A Neuromodulated Meta-Learning Algorithm (ANML). It differentiates through a sequential learning process to meta-learn an activation-gating function that enables context-dependent selective activation within a deep neural network. Specifically, a neuromodulatory (NM) neural network gates the forward pass of another (otherwise normal) neural network called the prediction learning network (PLN). The NM network also thus indirectly controls selective plasticity (i.e. the backward pass of) the PLN. ANML enables continual learning without catastrophic forgetting at scale: it produces state-of-the-art continual learning performance, sequentially learning as many as 600 classes (over 9,000 SGD updates).

1 Introduction

Intelligent animals are typically able to solve new problems without corrupting hard-won knowledge of how to solve previously encountered problems. On the other hand, artificial neural networks have long been known to suffer from catastrophic forgetting (CF) [12], in which learning to solve new tasks rapidly degrades previously acquired capabilities. This problem is particularly acute for continual learning systems that encounter tasks sequentially. For that reason, deep learning's success stories have come in the non-sequential setting, with data sampled independently and identically distributed (i.i.d.) from the full dataset (aka interleaved, or shuffled, training). A number of solutions to catastrophic forgetting have been developed, but they are typically *manually designed*.

One such solution is to use replay methods [37], which alleviate forgetting by saving prior experiences and mixing them with newly encountered data to approximate interleaved (i.e. not sequential) training. However, the ability to interleave memories of all previously seen tasks between training examples on new tasks is expensive in both storage and computation, and does not scale to many tasks.

A second class of solutions attempt to limit the extent to which parameters can be modified in response to new data, which we refer to as *selective plasticity*. But such approaches typically involve manually designed heuristics that control such plasticity. In the extreme, all previously learned weights can be frozen and additional capacity can be added to the network to solve new tasks [36]. Similarly, hand-designed modules within an overparameterized network can be frozen when they are used to solve a task [8]. This approach also scales poorly as the number of tasks become large, and, by design, old pathways cannot improve based on information from other tasks.

Rather than completely freezing its previously learned value, *elastic weight consolidation* (EWC) [19]) alters the plasticity of a given parameter (indirectly, via regularization penalties) in proportion to the manually-selected criteria of Fisher information, which approximates how important each parameter was for solving prior tasks. Those parameters deemed unimportant are made comparatively more susceptible to change, thus allowing the network to adapt to new problems while reducing corruption of existing knowledge. Other approaches similarly seek to modulate learning via task-specific synaptic importance [45], via the top-down attention mechanism of contrastive excitation backpropagation [20, 46], by interleaving pseudo examples generated from the current weights to limit plasticity [25], by employing L2 regularization on the weight changes between tasks [24], or by combining regularization and the Copy Weight with Reinit strategy [27, 30].

Another approach to mitigate CF is to directly incentivize the creation of maximally sparse or disjoint representations [11, 26], with the goal of minimizing interference between activations. Such sparse representations indirectly affect which parameters get updated during backpropagation, thereby allowing the network to avoid forgetting. However, a perspective we advocate is that, when possible, we should not optimize for one thing (e.g. sparse representations) and hope doing so leads to another thing (in this case, reduced catastrophic forgetting): Instead, we should optimize directly for what we want (here, learning without forgetting).

Rather than trying to manually implement a solution to CF or adding auxiliary losses we believe will alleviate CF, in this work we directly optimize to learn without forgetting. In other words, we harness meta-learning to allow the network to learn to continually learn. However, instead of vanilla meta-learning (e.g. MAML [9] with traditional neural networks), a contribution of this paper is introducing a new network architecture that improves the ability to learn to continually learn. Specifically, we have one network, conditioned on the input, gate the activation of another network (i.e. context-dependent gating), resulting in *selective activation* of that second network. Gating functions for such conditional computation [3] have been learned via the REINFORCE algorithm [2] and at scale via a standard backpropagation in a sparsely-gated mixture-of-experts [38], but these works focus on computational capacity and efficiency rather than catastrophic forgetting.

[1] University of Vermont, USA, email: {shawn.beaulieu, lapo.frati, ncheney} @uvm.edu
[2] Uber AI Labs, USA, email: {tmiconi, joel.lehman, kstanley} @uber.com
[3] OpenAI, USA, email: jeffclune@OpenAI.com. Current affiliation (work done at Uber AI Labs).
*Co-senior authors

One prior work [32] showed the promise of gating mechanisms for mitigating catastrophic forgetting, including high performance for up to 500 classes, although in an easier problem setting in which the network is explicitly told which task it is being trained or tested on, instead of having to learn that too. In realistic applications, such as a robot learning in the world, there will not be clear boundaries between tasks nor an oracle to provide task labels. Additionally, in this prior work, which neurons were gated was randomly chosen. Finally, the random gating was dependent on the task, not the data itself. In this work we remove the need to tell the network which task it is being trained or tested on, and we meta-learn the activation gatings, which provides the opportunity to substantially improve performance over random gating and enables the gatings to be dependent on the data itself. Finally, our learned masks are continuous and not binary (as in the prior work), further increasing the potential power of our method over binary, random gating.

While all of the above approaches are interesting and mitigate CF to some extent, they nevertheless fall into the category of manual approaches to reduce CF. That is, to solve a challenging problem they try to identify the required building blocks and how to combine them, and then let machine learning tune parameters within this complex assembly [5]. However, a clear trend in machine learning is that manually designed solutions give way to entirely learned solutions, which ultimately perform much better, once sufficient compute and data are available. That has been true, for example, for learning features (e.g. in computer vision or speech recognition) [21], neural architectures [47, 35], and hyperparameters [29]. An alternative to the manual path to AI that is in line with this trend of replacing hand-designed pipelines with learned solutions is to develop *AI-generating algorithms* [5], which try to learn as much of the solution as possible, including learning the learning algorithms themselves via meta-learning.

The meta-learning approach directly sets the ultimate goal (here: the ability to continually learn without forgetting) as a meta-loss and employs the power of machine learning to produce an AI algorithm well suited to it. An example is *Model-Agnostic Meta-Learning* (MAML) [9], which searches for an initial set of weights for a neural network that, when subjected to SGD, rapidly learns a new task. It does so by differentiating through many steps of inner-loop SGD learning to calculate the outer-loop gradient of how to improve the weight initialization.

We are aware of only one prior work that optimizes to solve catastrophic forgetting via meta-learning. We were inspired by it, build off of it, compare to it, and adopt its experimental protocol. *Online aware Meta-Learning* (OML) [17] employs a MAML-style meta-learning algorithm to produce a representation (set of neural network layers) that, when frozen and used by additional, downstream neural network layers, minimizes catastrophic forgetting in those downstream layers. Excitingly, while sparsity in this representation layer was not explicitly encouraged, OML produced not only sparsity, but a better version of it. Explicitly encouraging sparsity yields many dead neurons (those that never fire across the dataset), whereas OML optimized representations had no dead neurons [17]. This ability to search for what *is* empirically effective, rather than what *is believed* to be effective, distinguishes meta-learning approaches from manual-path approaches. OML provided a substantial advance over the prior state of the art, enabling continual learning over 200 sequential classes. Because OML is in line with our vision of meta-learning solutions to hard machine learning problems, and because it performs so well, we seek to further improve upon it.

Rather than meta-learning representations as in OML, we meta-learn a context-dependent gating function (a neural network, called the neuromodulatory network) that enables continual learning in another neural network (called the prediction network) (Fig. 1). The neuromodulatory network has the flexibility to explicitly turn on and off activations in subsets of the prediction network conditioned on the input. We call this mechanism selective activation. Selective activation in turn enables selective plasticity because the strength of backward gradients is a function of how active neurons were during the (modulated) forward pass, indirectly controlling which subset of the network will learn for each type of input. By leveraging meta-learning to optimize when and where to gate activations to maximize continual learning, our approach explores the potential for solutions beyond hand-designed selective plasticity strategies.

Our harnessing of selective activation via context-dependent gating and selective plasticity is inspired by neuromodulatory processes in the brain. These include inhibitory mechanisms that become active in response to specific environmental stimuli [32] and the suppression of synaptic plasticity [13] or activation [1] in the presence of neuromodulatory signals. There is prior work in machine learning that harnesses neuromodulation techniques [7, 16, 39, 40, 42], but in such work neuromodulation directly modulates learning rates, instead of the approach taken in this work of directly modulating activations and thus indirectly controlling learning. Modulating learning can ensure that information about one task can be localized to only the parts of the network relevant to that task. However, an insight in this work is that modulating learning alone is not enough, because there can still be interference between the tasks during the forward pass. For example, even if an agent's chess playing and bike riding networks are physically separated such that learning in one does not corrupt information in the other, neither task will be performed well if both are actively producing muscle outputs when performing either task. The insight behind why the approach in this work of directly modulating activations (and indirectly modulating learning) is superior is because it allows optimization to reduce interference in *both* the forward and backward passes of the network.

2 The Problem Formulation

One goal of continual learning is to solve catastrophic forgetting, meaning learning new tasks without forgetting of already learned skills. More precisely, we want to be able to learn a large number of tasks $\mathcal{T}_{1..n}$ *sequentially* from a common domain $\mathcal{T}$, but in such a way that after sequential learning average performance on all $\mathcal{T}_{1..n}$ tasks is high. Following [17], the experimental domain $\mathcal{T}$ is the Omniglot few-shot learning dataset. Each task $\mathcal{T}_i$ is a class of characters (each with k and v unique training and validation instances, respectively).

Before continuing, it is helpful to establish terminology for meta-learning, as it is complex. Because we are learning to learn, learning happens both on the outer loop and the inner loop. The goal is to have the outer loop take steps to make each round of inner-loop learning better. The phase in which the outer loop takes optimization steps to improve the learning ability of the inner loop is called *meta-training*. Once meta-training has concluded, we then want to test the quality of the inner-loop learner, a phase called *meta-testing*. During meta-training, within each inner loop, something must be learned, which is called *meta-training training*. For example, the inner-loop agent might be a normal neural network learning via SGD to classify MNIST examples. During meta-training training, it is shown multiple MNIST samples (and given the correct label) and SGD steps are applied to improve accuracy. After each inner-loop iteration of meta-training training, the trained agent must be evaluated, which

we call *meta-training testing*. This meta-train testing error is the *meta-training loss* to be minimized. After meta-training completes, we switch to meta-testing, wherein we need to evaluate how well the meta-learned learner performs. Again, we need a training and testing phase, which are called *meta-test training* and *meta-test testing*, respectively. We have found this MTT (for meta-train/test training/testing) terminology essential for clear communication and advocate the meta-learning community adopt it.

A naive application of meta-learning to this problem would involve performing the entire learning process at each iteration of the inner loop: i.e. learn a whole sequence of $n \times k$ characters, evaluate the network's validation performance on $n \times v$ validation instances (the meta-training loss), backpropagate the gradient of this validation error all the way through the learning process to the initial weights, perform one step of gradient descent on these initial weights, and repeat. However, when n is high (e.g. 600), this approach is unfeasible due to limitations with modern algorithms (e.g. unstable gradients) and hardware (e.g. memory).

The OML framework [17] introduced an elegant solution to this problem by setting the meta-loss to be an approximation of this true, desired meta-loss. The idea is to measure after each new class whether (1) it was learned and (2) whether old knowledge was lost. After each newly encountered class (k instances), the meta-loss is calculated as the error on the newly learned class and the error on a random sample of characters from all meta-train training classes (called the *remember set*). Ideally this remember set would only be previously seen meta-train training set classes, but we follow the choice of [17] in sampling from all meta-test training classes). The remember set is used *in meta-training only*.

As a result, the network is meta-trained to learn a potentially large number of different classes without catastrophic forgetting, even though each inner-loop only involves learning k instances of one character class, which makes the process computationally tractable. We use the same meta-learning procedure for training our novel architecture.

The experiments employ the Omniglot few-shot learning dataset, which has 1,623 character classes [22]. 660 classes are held out for meta-testing. The remaining 963 are for meta-training. For meta-train training, $k = 20$, i.e. 20 labeled instances of each character are provided. The remember set randomly samples from these same 20 instances per class. In meta-testing (described below), 15 instances of each character are used for meta-test training, while the remaining 5 instances are held out for meta-test testing.

3 A Neuromodulated Meta-Learning Algorithm (ANML)

While meta-learning provides the opportunity to learn the solution to a problem instead of manually designing it, we still must decide on the search space, including which *materials* the neural networks are made of [5]. To attempt to solve catastrophic forgetting, here we propose learning a neuromodulatory network, which is a context-dependent function that gates the forward pass of another (otherwise normal) neural network (Fig. 1). This setup allows different subnetworks within the normal network to be used for, and learn from, different types of tasks. It also allows the model to interpolate between tasks it has not previously encountered. By meta-learning, we automate, rather than presuppose, the contexts and locations within the network that should be active at any given time, and to what degree. We call this approach A Neuromodulated Meta-Learning algorithm (ANML).

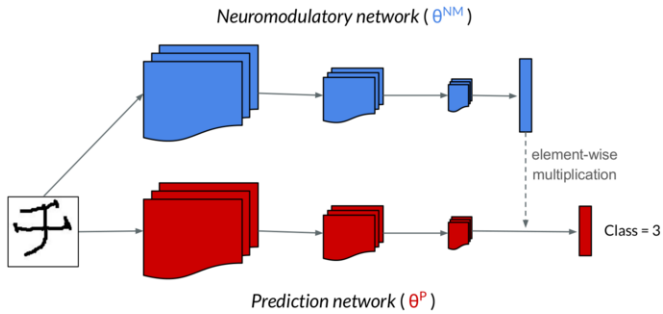

Figure 1: The architecture for *A Neuromodulated Meta-Learning algorithm* (ANML). The prediction network (red) is a normal neural network updated in the inner loop via SGD (or similar). The neuromodulatory network (blue) produces an element-wise gating of the prediction network's forward-pass activations, enabling selective activation (i.e. conditional computation) and indirectly enabling selective plasticity by affecting the gradient updates of the prediction network. The initial weights (at the start of each inner loop) of both the neuromodulatory and prediction network are meta-learned in the outer-loop of optimization. The weights of the neuromodulatory network are not updated in the inner loop, but those of the prediction network are. The example image is from the Omniglot dataset, which is the experimental domain for the experiments in this paper.

3.1 ANML Architecture

OML [17] divides a single deep neural network into two distinct components. The first 6 layers, which are convolutional, have meta-learned parameters that are frozen (not changed) during the inner loop. This subnetwork is called the *representation learning network* (RLN). The final 2 layers, which are fully connected, have parameters that meta-learn their initial weights in the outer-loop, and then are updated during each inner-loop via SGD. See Javed and White [17] for details. This subnetwork is called the *prediction learning network* (PLN).

ANML takes a different approach, with two parallel neural networks: a neuromodulatory (NM) network and prediction network (Fig. 1). The weights for both are meta-learned in the outer loop. The weights of the neuromodulatory network are not updated in the inner loop, but those of some, but not all, of the prediction network are (which prediction network weights are updated differs for meta-training and meta-testing, as described below). To keep overall architecture and parameter sizes similar to [17], each of these networks has 3 convolutional layers (each followed by a batchnorm layer [15]) and one fully connected layer. The final layer of the neuromodulatory network is of the same size as the input to the final layer of the prediction network (i.e. the flattened latent representation output by the final convolutional layer in the prediction network). The neuromodulatory output is used to gate the latent representation of the prediction network (via element-wise multiplication) during a forward pass. All activation functions are ReLUs except the gating multiplier is restricted to the range $[0, 1]$ via a sigmoid, meaning that in this work it can only suppress activations of the prediction network (instead of negate or amplify them).

3.2 Meta-Training Learning Procedure

The ANML algorithm (Algorithm 1)[4] consists of an inner loop nested inside an outer loop of optimization. Within each inner loop, a

[4] Code available at `github.com/uvm-neurobotics-lab/ANML`

copy θ_0^P of the prediction network weight initializations θ^P is trained for 20 SGD iterations – producing $\theta_1^P ... \theta_{20}^P$ – on the training set of a single Omniglot meta-training class $\mathcal{T}_n$.

During each of these 20 forward passes, the inputs to the prediction network's final layer are gated by neuromodulatory weights θ^{NM}, enabling *selective activation* by modifying the outputs of the prediction network during the forward pass.

During each backwards pass, the gating of the prediction network naturally has the effect of reducing the gradients flowing back towards a subset of its weights, thereby modifying the SGD update and resulting in *selective plasticity*. The neuromodulatory weights θ^{NM} are not updated within the inner-loop.

Following the 20 sequential updates on this single Omniglot meta-training class, a meta-loss is calculated by making predictions using the final post-inner-loop-training weights (θ_{20}^P) on all 20 images from the single Omniglot class just trained on plus a random sample of 64 character instances from the set of all meta-training classes (i.e. from the remember set). This meta-loss function is referred to as the Online aware Meta-Learning (OML) objective [17], and incentivizes the meta-learning of networks that can learn a new tasks over many steps of SGD without forgetting how to solve previously learned tasks.

We then backpropagate this meta-loss back through the 20 steps of SGD updates (following the style of MAML [9]) to calculate the gradients of the initial prediction network weights θ^P and the neuromodulatory network weights θ^{NM}, and then take an outer-loop gradient update step on each of them. This outer-loop step is taken via Adam [18], while gradient updates for all inner-loop updates are via SGD with a fixed learning rate. This step constitutes the completion of the first outer-loop iteration. The next outer-loop iteration then begins by copying the prediction net weights and conducting inner-loop training starting with them on 20 instances from the next Omniglot meta-training class. For the experiments in this paper, this process continues for 20,000 outer-loop meta-training iterations.

Following OML [17], when a copy of the prediction network is made at the start of each meta-iteration, the weights in the final layer that lead into the output node for the single Omniglot class in the upcoming meta-train training trajectory are initialized randomly in that inner-loop copy (rather than being initialized with their meta-learned weight initialization). This ensures that the inner-loop learner has not already converged to a good solution to classifying images in the upcoming trajectory through its initial-weight-meta-learning alone, and will result in larger gradient steps during backpropagation – both of which more closely match the situation that will be encountered during meta-testing, when each class encountered is new. Note that reinitializing the entire final layer would not be ideal, as it would not enable the network to make intelligent predictions about the remember set (which contains classes not trained on during this inner-loop). This weight reinitialization procedure is not employed during meta-test-training.

3.3 Performance Evaluation in Meta-Testing

Following the completion of meta-training, in the meta-test phase (Algorithm 2) the resulting prediction and neuromodulatory networks are evaluated on their ability to learn many tasks while minimizing catastrophic forgetting.

Starting from the meta-learned prediction network weight initialization θ^P and neuromodulatory network θ^{NM}, the weights of the fully connected layer in the prediction network (and only those weights) are fine-tuned on meta-test classes. This idea of freezing the weights of all but the last few layers of the prediction network follows [17]. These weights are fine-tuned for q (here, 15) instances of each meta-test class (here, the 600 Omniglot classes that were not used during meta-training). Unlike meta-training, in meta-test training, a new copy of the prediction network is not made for each new class. Thus, in our experiments, the final weights from meta-training are fine-tuned during meta-test training with 9,000 iterations of SGD. The model therefore undergoes 8,985 fine-tuning updates since it last saw an instance from the first Omniglot meta-test training class. Using these final weights ($\theta_{9000}^P, \theta^{\text{NM}}$), all 9,000 meta-test training instances are reevaluated to assess the meta-test training performance of the model (i.e. how well it can learn the training set without forgetting). Furthermore, the final model ($\theta_{9000}^P, \theta^{\text{NM}}$) is evaluated on five held-out instances from each of the 600 meta-test classes (the meta-test test set), which measures the model's ability to generalize what it has learned to novel instances of each class. In short, the meta-test training performance shows the ability to memorize without forgetting, and the meta-test test performance reveals the ability to conduct continual learning in a way that allows generalization, which is what we ultimately care most about.

Following OML's evaluation protocol [17], this procedure is repeated for various sequence lengths of meta-test classes (trajectories of 10, 50, 75, 100, 150, 200, 300, 400, 500, and 600 Omniglot classes) to show how the models scale to longer sequential task trajectories. A hyperparameter search is performed for each sequence length to set the learning-rate β for its inner-loop updates.

Algorithm 1 A Neuromodulated Meta-Learning algorithm (ANML)

Require: $\mathcal{T} \leftarrow$ trajectory of N sequential meta-training tasks
Require: $\theta^{\text{NM}} \leftarrow$ weights of the neuromodulatory network
Require: $\theta^P \leftarrow$ weights of the prediction network
Require: $\alpha, \beta \leftarrow$ learning-rate hyperparameters
initialize θ^{NM}, θ^P
for $n = 1, 2, \ldots$ **do** ▷ meta-learning outer-loop
  $S_{traj} = \mathcal{T}_n$ ▷ trajectory for inner-loop training
  $S_{rem} \sim \mathcal{T}$ ▷ sample instances from all tasks to remember
  $\theta_0^P = \theta^P$ ▷ create inner-loop copy of prediction net
  for $i = 1, 2, \ldots, k$ **do** ▷ task-learning inner-loop
    $\theta_i^P \leftarrow \theta_{i\text{-}1}^P - \beta \nabla_{\theta_{i\text{-}1}^P} \mathcal{L}(\theta^{\text{NM}}, \theta_{i\text{-}1}^P, S_{traj})$ ▷ SGD on $\theta_{i\text{-}1}^P$
  end for
  $\theta^{\text{NM, P}} \leftarrow \theta^{\text{NM, P}} - \alpha \nabla_{\theta^{\text{NM, P}}} \mathcal{L}(\theta^{\text{NM}}, \theta_k^P, S_{traj}, S_{rem})$
    ▷ meta-update on θ^{NM}, θ^{P} w.r.t. final inner-loop θ_k^P
end for

Algorithm 2 Meta-Testing Evaluation Protocol

Require: $\mathcal{T} \leftarrow$ trajectory of N unseen sequential meta-testing tasks
Require: $\theta^{\text{NM}} \leftarrow$ meta-learned weights of the neuromodulatory net
Require: $\theta^P \leftarrow$ meta-learned weights of the prediction network
Require: $\beta \leftarrow$ learning-rate hyperparameter
$S_{train} = [\,]$
for $n = 1, 2, \ldots, N$ **do**
  $S_{traj} \sim \mathcal{T}_n$ ▷ get next task training trajectory
  $S_{train} = S_{train} + S_{traj}$ ▷ add to meta-test train set
  for $i = 1, 2, \ldots, k$ **do**
    $\theta^P \leftarrow \theta^P - \beta \nabla_{\theta^P} \mathcal{L}(\theta^{\text{NM}}, \theta^P, S_{traj})$ ▷ SGD on θ^P
  end for
end for
record $\mathcal{L}(\theta^{\text{NM}}, \theta^P, S_{train})$ ▷ eval final θ^P on meta-test train set
$S_{test} = \mathcal{T} - S_{train}$ ▷ held-out meta-test test set
record $\mathcal{L}(\theta^{\text{NM}}, \theta^P, S_{test})$ ▷ eval final θ^P on meta-test test set

3.4 Baseline Controls

To help analyze the effectiveness of ANML, we compare it to a series of controls. All architectures in the control treatments share approximately the same numbers of total parameters as ANML (∼6M). They do not include a neuromodulatory network, but instead have the OML architecture (described in Section 3.1). Empirically, we found that the batchnorm layers included in ANML were detrimental to the performance of OML, so they were omitted from these controls (consistently with [17]).

Training from Scratch: Consistently with traditional machine learning, this control involves no meta-training and simply randomly initializes a prediction network at the start of meta-test training time. In this treatment the entire network can learn, rather than fine-tuning only the fully-connected layers, as the network does not include any learned features at the start of meta-test training.

Pretraining and Transfer: The previous control is perhaps unfair because ANML gets to learn from the meta-training set before being evaluated on the meta-test set. The Pretraining and Transfer control tries to address this inequity. Consistently with traditional transfer learning for deep neural networks [44], this control pretrains i.i.d. on the meta-training training image set. The number of images seen during pretraining is set to be equivalent to the numbers of instances seen during meta-train training and meta-train testing (1.68M image evals).

The weights learned during this pretraining phase are transferred to the meta-test phase, where the fully connected layers are fine-tuned on the meta-test training trajectory.

Online aware Meta-Learning (OML): OML [17] represents the current state-of-the-art on this problem domain. ANML has the same meta-learning procedures as it, with the exception that OML does not include a neuromodulatory network (θ^{NM}) for selective activation, and instead meta-learns the weights of the convolutional layers only, which they call the Representation Learning Network (RLN) [17]). The RLN weights in OML are frozen at meta-test time and within each inner-loop of meta-train training: they are modified only via the outer-loop updates during meta-training. Similarly, ANML freezes the convolutional weights and batch norm weights of the prediction network during meta-testing, only updating the weights of the fully connected layer of the prediction network during this meta-test phase. However, in contrast to OML, in ANML no weights in the prediction network are frozen during meta-train training. Additionally, as with the parameters in OML that are only meta-learned (the RLN), the weights of the neuromodulatory network in ANML are frozen during both meta-testing and meta-train training (they are only updated via outer loop steps during meta-training).

The meta-test procedure for the original OML algorithm includes fine-tuning both fully-connected layers in the prediction network at meta-test time, while freezing the 6 convolutional layers. However, fine-tuning two layers (vs. one) leads to approximately 50% more fine-tuned parameters (∼3M) compared to the ANML architecture (which fine-tunes ∼2M parameters). To counterbalance this, we also examine a treatment that fine-tunes only the final layer of OML, which results in about 50% fewer fine-tuned parameters (∼1M) than in ANML. We call this control treatment *OML with One-Layer Fine-Tuned* (OML-OLFT). The OML-OLFT meta-testing procedure often (but not always) results in improved performance compared to the original OML, but never results in OML-OLFT outperforming ANML. Because it is the published previous method, we refer to the original OML formulation when making statistical comparisons to OML in the text (unless otherwise noted), but we show both in plots.

Interleaved-Training Oracles: To understand the limits of this problem domain, the "Oracle" version of any of the above algorithms keep their same meta-training procedure, but at meta-test time are presented with an i.i.d. sampling of all the classes from the meta-test training set. These treatments thus examine the ability of an algorithm to learn without the challenge of catastrophic forgetting from sequential/continual learning, and provide an upper bound on the performance on a model with respect to avoiding catastrophic forgetting.

4 Results

4.1 Continual Learning at Scale

To our knowledge, the longest continual learning trajectories reported to date that exhibit robustness to catastrophic forgetting involve learning 200 tasks sequentially and come from OML [17].[5] Here we push that further, testing whether 600 sequential tasks can be learned – seeking sequential learning across up to 9,000 SGD updates without catastrophic forgetting.

For each treatment, we meta-train 10 independent models on the same set of 963 meta-training classes. At the *end* of the meta-test *training* trajectory, we re-evaluate the trained models on all of the meta-test *training* data seen over the trajectory to see how well they resist catastrophic forgetting. ANML significantly outperforms OML for meta-test training trajectories at all lengths tested (all $p \leq 1.26 \times 10^{-8}$; all p-values computed using the Mann-Whitney U test [31]).

Consistent with the expectation that standard neural networks suffer from catastrophic forgetting, both pretraining-then-transferring learned representations, and training randomly initialized networks from scratch have significantly worse meta-test training accuracy than either OML and ANML for all class sequence lengths tested (all $p \leq 6.023 \times 10^{-20}$), resulting in very low classification accuracy (Fig. 2; $< 3\%$ on trajectories of ≥ 50 sequential classes for *Scratch*, and $< 3\%$ on trajectories of ≥ 400 sequential classes for *Pretrain*)

Interestingly, the OML-OLFT treatment, which fine-tunes fewer parameters than standard OML, has higher meta-test training accuracy than OML on trajectories of 300 or more classes, but performs significantly worse on this metric for trajectories of up to and including 200 classes. All $p \leq 1.87 \times 10^{-13}$ up to 200 classes for OML relative to OML-OLFT; and all $p \leq 1.33 \times 10^{-29}$ for trajectories equal to 400 or more for OML-OLFT relative to OML.

We now examine meta-test *testing*, which involves the more difficult challenge of learning (without forgetting) in a way that *generalizes* to never-before-seen instances of classes learned during meta-test training. For each of the 10 meta-training trials of each algorithm, we evaluate the meta-trained model on 10 independent meta-test training trajectories, each with a random set and order of tasks drawn from the 660 meta-test classes that were held out from the meta-training dataset. For trajectories of any length, ANML has significantly better meta-test *test* accuracy than any other treatment including OML (Fig. 3, $p \leq 2.58 \times 10^{-12}$). After meta-test training on 600 classes, ANML is able to correctly classify a mean of 63.8% of held-out meta-test test instances, compared to the 18.2% for OML and 44.2% for OML-OLFT. Furthermore, ANML significantly outperforms chance ($p < 7.96 \times 10^{-6}$) on 99.3% of classes over the 600-task sequence, including many classes that it has not seen for

[5] Sequences up to 500 have been tried, but only when providing a knowledge of which unique task is being solved directly to the network instead of requiring it to learn that too [32].

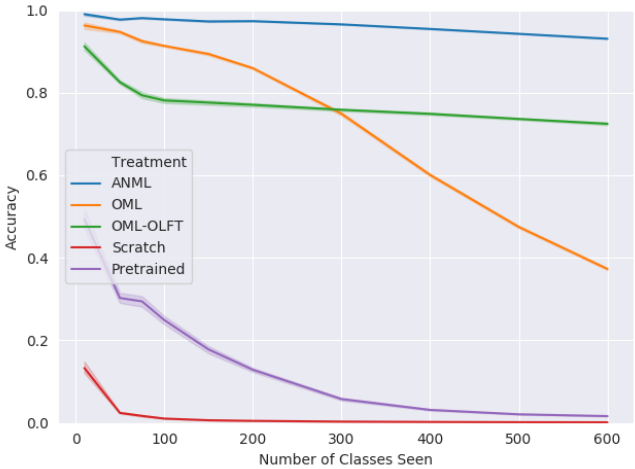

Figure 2: Meta-test *training* classification accuracy. The x-axis shows the number of sequential tasks/classes in the meta-test training trajectory. Accuracy is calculated with the final prediction network parameters after training sequentially on that full meta-test training trajectory, and on all instances in the that trajectory (i.e. the meta-test training set).

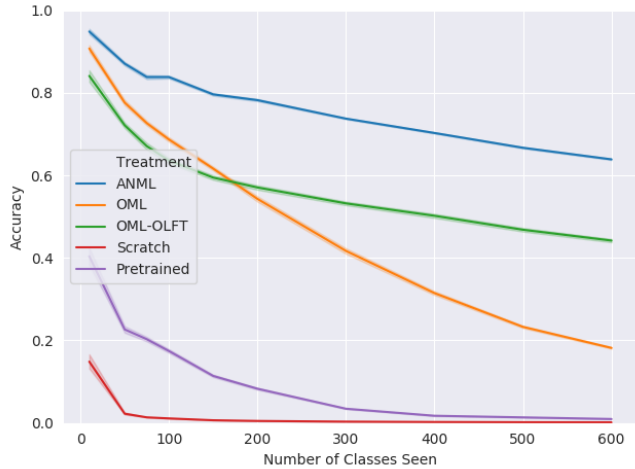

Figure 3: Meta-test *testing* classification accuracy. The x-axis shows the number of sequential tasks/classes in the meta-test training trajectory. Accuracy is calculated with the final prediction network parameters after training sequentially on that full meta-test training trajectory, and the evaluation is on held-out (i.e. test) instances of the meta-test classes. Thus, these meta-test test instances were not seen during meta-training or meta-test training. For all trajectory lengths tested, ANML significantly outperforms OML, the pretrained-and-transfer networks, and models trained from scratch.

hundreds of tasks (thousands of SGD updates), demonstrating learning without forgetting catastrophically[6].

At meta-test time, the OML-OLFT treatment is significantly inferior to OML for trajectories of up to 150 classes ($p \leq 6.23 \times 10^{-8}$), but is better able to scale to longer trajectories, outperforming OML when sequentially learning 300 or more classes ($p \leq 4.69 \times 10^{-12}$ for ≥ 300 classes). While OML-OLFT fine-tunes fewer parameters during meta-test training, additional experiments explore fine-tuning of a greater number of parameters for both the OML and ANML networks during meta-test training find that ANML maintains a greater proportion of its performance when some or all additional layers are also fine-tuned[7].

Given that ANML significantly reduces catastrophic forgetting, a natural question to ask is what the upper-bound on performance is when catastrophic forgetting isn't a problem – that is, when data is presented non-sequentially. The ideal version of any model, or "oracle", can be approximated by training it in an interleaved fashion; replacing the sequential trajectory of correlated images with i.i.d. samples from the set of all images in this sequence. As expected, the i.i.d. oracle version of each algorithm significantly outperforms its sequentially-trained counterpart (Fig. 4, $p \leq 1.27 \times 10^{-34}$ for all treatment vs. oracle pairs for sequences of length 600). However, the sequentially-trained version of ANML significantly outperforms even the oracle versions of OML and the other algorithms at 600 classes (all $p \leq 1.93 \times 10^{-23}$). The only treatment that outperforms ANML is the ANML-Oracle (for 600 classes, ANML-Oracle: 71%, ANML 63.8%, $p = 1.27 \times 10^{-34}$). The finding that ANML produces models that perform better than traditional approaches even when catastrophic forgetting is not an issue hints at the promise of the ANML approach to also provide performance improvements to traditional i.i.d. training tasks in addition to the continual learning setting, which is an interesting area for future work.

Another useful metric is the relative *drop* in performance from the i.i.d. oracle to the sequentially-trained version of each algorithm.

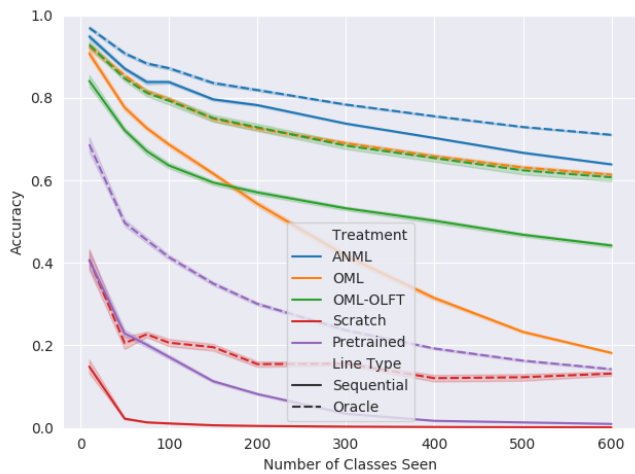

Figure 4: Meta-test *testing* classification accuracy vs. oracles. The x-axis shows the number of sequential tasks/classes in the meta-test training trajectory. Accuracy is calculated with the final prediction network parameters after training on that full meta-test training set i.i.d. for all *oracle* treatments (and after sequential training for the ANML treatment). Accuracy is reported for held-out instances of the meta-test classes not seen during meta-test training or meta-training. ANML outperforms the i.i.d.-trained oracle versions of all other treatments.

Presuming that the primary difference between these treatments is the presence of catastrophic forgetting, the smaller the relative performance drop between them, the more the sequentially trained version of the algorithm has avoided catastrophic forgetting. Of the relative drops in performance on trajectories of 600 classes for the sequentially learned versions of training-from-scratch (99% relative drop in performance from i.i.d. to sequential learning), pretraining-

[6] Figure available in supplementary material of arXiv manuscript
[7] Figure available in supplementary material of arXiv manuscript

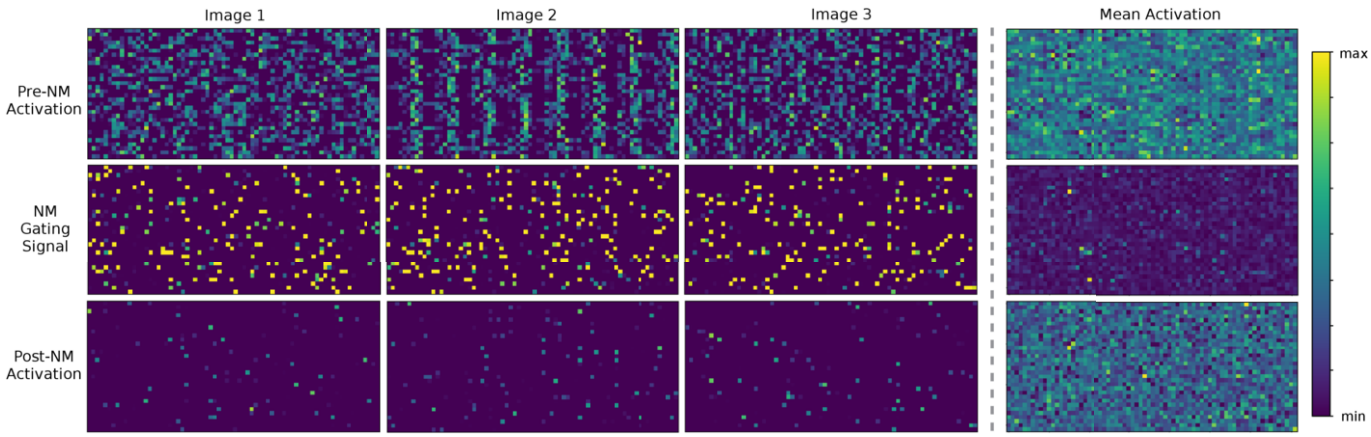

Figure 5: The sparsity of activations before (top row) and after (bottom row) the neuromodulatory gating signal (middle row) has been applied, shown for three random inputs from the meta-test test set, and the mean across all images in the meta-test test set. Colorbars for subfigures are individually normalized to better show min (blue) and max (yellow) activations. Note that post-NM activations are sparse for each individual image, but near-uniformly distributed on average, revealing that NM helps to create sparse, orthogonal representations, and efficiently uses all of its compute resources instead of creating wasteful "dead neurons."

and-transferring (99%), OML (70.32%), OML-OLFT (27.2%), and ANML (10%), ANML shows the smallest relative drop in performance between the i.i.d. and sequentially-trained version (all $p \leq 3.11 \times 10^{-24}$ relative to ANML), further supporting the benefits of ANML over the other algorithms for continual learning tasks. It is remarkable to see such a small performance drop (only 10%) for ANML between the i.i.d. and sequential version of the task, meaning that ANML is solving most of the catastrophic forgetting problem on this challenging test across 600 sequential tasks.

Traditionally, continual learning in neural networks involves seeing examples in a single continuous stream – therefore, the oracle treatments described above see each meta-test training image *exactly once*. That is one reason why the performance is so low for standard deep learning techniques, such as training from Scratch and Pretraining & Transferring, even with i.i.d. sampling (Fig. 4). With additional passes through the data, performance increases across the board. After 20 epochs of i.i.d. training on the 600-class meta-test training dataset, training from Scratch gets 61.8% accuracy on the meta-test test set, while the Pretrain & Transfer control results in 48.66% accuracy. While these accuracy levels may seem low, recall that the Omniglot meta-test training set here contains 600 classes, each with *just 15 training instances*. By comparison, the relatively high training accuracies (e.g. 99.77% at 500 epochs of training [4]) on handwritten digits in the MNIST dataset [23] result from a dataset with just 10 classes, each with 6,000 training instances. Because the number of training instances per class is low, the Scratch and Pretrain controls overfit. Evidence of overfitting is that, while their test accuracy is low, their performance on the meta-test *training* set is near-perfect (for Scratch accuracy is 99.3%; for Pretrain & Transfer it is 98.9%).

In this context, it is remarkable how well ANML performs. With just one pass through the data, and with the data ordered sequentially instead of i.i.d., the accuracy of ANML (63.8%) is still higher than the Scratch and Pretrain & Transfer 20-epoch, i.i.d. controls. The performance of ANML further increases when it has the benefit of training *i.i.d* for one epoch (Fig. 4; 71%). With 20 epochs, the accuracy increases further, to 75.37% for ANML. ANML continues to significantly outperform all other treatments with more epochs of training[8] ($p = 3.38 \times 10^{-12}$). It is an open, interesting, important research question to figure out exactly why ANML outperforms the controls even in the multi-epoch, i.i.d. setting.

4.2 The Meta-Learned Representations

Prior work has shown that encouraging sparse representations (e.g. via an auxiliary loss) alleviates catastrophic forgetting [11, 12, 26]. One of the key findings of OML [17] was that, despite not explicitly incentivizing representations to be sparse, when meta-learning directly for the reduction of catastrophic forgetting, the system learned on its own to produce sparse representations. Here we investigate whether sparse representations also arise in ANML without an explicit incentive for sparsity.

To examine the output of the neuromodulatory network, and how its gating affects the sparsity of the prediction network's representation, we analyze the activations of both networks in response to the images in the meta-test test set. The prediction network's representation layer *prior* to neuromodulation has a mean of 52.77% of neurons active (defined as activations > 0.01). *After* neuromodulation, the proportion of neurons in this layer that are active drops significantly to 5.9% on average ($p < 10^{-6}$). By comparison, the sparsity of the representation in OML to these same images is 3.89%. The average number of neurons active in the last layer of the NM network (which does the gating) is 56.9%. Finally, both OML and ANML have 0% "dead neurons" at the representation layer during meta-test training time. This means that every neuron is active for at least one class in the meta-test training set. In contrast, directly encouraging the production of sparse representations creates dead neurons, which are wasteful, at much higher frequencies [17], providing more evidence that it is better to optimize directly for what we want (learning without forgetting), rather than optimizing for one thing (sparsity) and hoping we get an optimal version of something else.

For a randomly chosen meta-training trial and for three different example input images (randomly chosen from the meta-test test set), the activations of the neuromodulatory network and the pre- and post-neuromodulatory-gating activations of the prediction network

[8] Figure available in supplementary material of arXiv manuscript

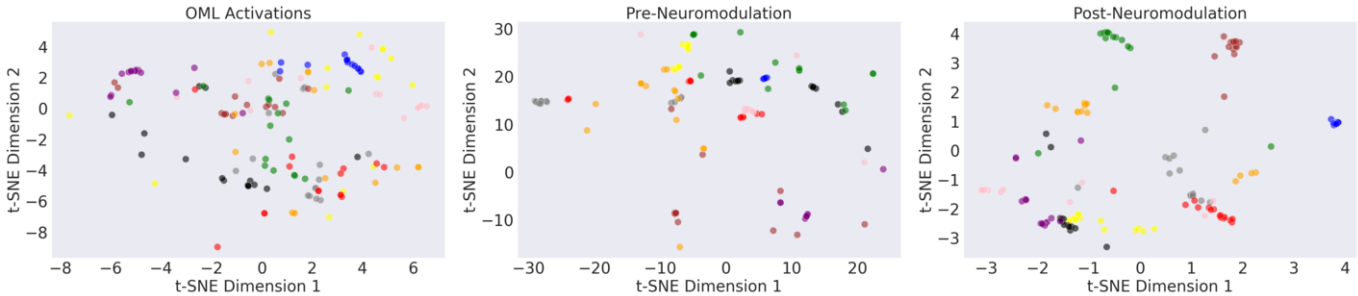

Figure 6: 2D t-SNE projections of the latent representations for 10 randomly selected meta-test training classes when the network saw them for the first time (before any labels are provided). *Left:* OML representations. *Middle:* PLN representations before being gated by the neuromodulatory network. *Right:* PLN representations after being gated by the neuromodulatory network. Qualitatively, the post-neuromodulatory activations provide more well-separated clusters. KNN classification accuracy quantitatively shows the improved accuracy that results from such separation (see text).

are shown in Fig. 5. This visualization, in combination with the average activation rate reported above, show that ANML has indeed learned to produce sparse representations simply through a incentive to directly maximize continual learning, as OML did [17]. Furthermore, that ANML is more successful at avoiding catastrophic forgetting than OML, yet does so with a less sparse representation, suggests that having sparse representations alone are insufficient. ANML is able to meta-learn a combination of sparse representations and selective plasticity that together are especially important for continual learning.

We hypothesize that the neuromodulation networks learns to recognize different types of images, and creates different selective activation masks for them. That would, in turn, enable *different parts of the prediction learning network* to store information about *different types* of images, reducing catastrophic forgetting (recall that selective activation leads to selective plasticity). A prediction of this hypothesis is that the outputs of the neuromodulatory network should be different for different types of images (i.e. that it should be possible to cluster semantically similar images by their neuromodulatory outputs).

To test this prediction, we use the K-Nearest-Neighbor (KNN) algorithm [10] to predict the class label of images from the meta-test test set, using the Euclidean distance in the space of NM activations (labels come from the meta-test training set). The following results are with $K = 5$, but the results are qualitatively unchanged across $1 \leq K \leq 20$).

The result show that the NM outputs enable an accuracy of 70.9%, which is significantly higher than chance. The accuracy of a neural network with the same architecture as the NM network, but with random weights is 24.3% ($p = 2.58 \times 10^{-31}$). This result confirms that the NM network has meta-learned the ability to tell apart types of images, including classes it has never seen before, enabling it to create different selective activation and plasticity for different types of images. Interestingly, the separability the neuromodulatory network creates also makes it easier for the prediction network to learn to solve the task (because the classes are better separated).

Another test of the value of the NM network is if it *improves* the representations in the PLN. To test that, we compare KNN accuracy based on the PLN activations before and after they are gated by the NM network. The KNN classification accuracy post-neuromodulation is 81.1%, which significantly outperforms the 57% classification accuracy of pre-neuromodulation activations ($p = 7.87 \times 10^{-12}$).

The improvement to class separability post-neuromodulation can also be seen visually in 2-D t-SNE [28] projections of the representation layer activations for 15 instances each of 10 meta-test classes when the network (resulting from a single randomly chosen meta-training trial) saw these instances *for the first time* during meta-test-training (Fig. 6). The meta-learned representations in the PLN already do a good job of separating the never-seen-before classes, which makes it easier to learn to perform the task well. The meta-learned NM network further increases that separability via selective activation. That, in turn, creates selective plasticity, causing information about each class to be stored in different parts of the network, reducing catastrophic forgetting.

5 DISCUSSION AND FUTURE WORK

The results above demonstrate ANML's impressive ability to continually learn, outperforming the current state-of-the-art approach of OML on this domain [17]. The results show the promise of meta-learning selective activations (and by extension, selective plasticity) to help reduce catastrophic forgetting and enable continual, lifelong learning in deep neural networks. In future work it would be interesting to study the extent to which ANML and meta-learning in general improve other aspects of continual learning, such as forward transfer (being better on future tasks due to prior experience) and backwards transfer (getting better on previously learned tasks when learning new tasks). Sequentially-trained ANML even outperforms the i.i.d. trained OML-Oracle, suggesting that meta-learned neuromodulation and its resulting selective activations may be powerful for problems beyond reducing catastrophic forgetting, and represent a powerful architecture more generally.

ANML's absolute performance of 63.8% accuracy on held-out images from 600 Omniglot classes would not be impressive in an i.i.d., multi-epoch, large-training-dataset context. However, it is impressive considering (1) that the data are presented sequentially, including over nearly 9,000 SGD updates, (2) that each image is seen just one time, and (3) that the number of training instances per class is low (at just 15). ANML's achievement is clearer when comparing to the performance of traditional deep learning methods (the Scratch or Pretrain & Transfer controls) in this difficult task setting.

There are many directions for future work. Initially, while there is only a 10% drop in performance owing to catastrophic forgetting (i.e. between the ANML-Oracle and ANML), that still leaves room for improvement. There is also work to be done in analyzing the implementation choices made during this study, which could further improve performance. For example, the choice to gate selective ac-

tivations in just one layer of the prediction network was a simplifying assumption made in this initial exploration, but the methodological approach can easily be extended to modulate any and all layers within the prediction network. Additionally, the technique could be made more fine-grained. Indeed, during the development of ANML, a version of neuromodulation was created where every synapse in the prediction network was gated by the NM network. This version was competitive with other versions that were tested, but required too many parameters to be a viable alternative given our current computational resources. To avoid this explosion in the parameter count, one could harness indirect encoding, as in HyperNEAT [41], where a small network can create a geometric pattern (here, of gating) that can be applied to a larger network. Interestingly, this approach would also enable the NM network to be able to be applied to prediction networks of different sizes at meta-test time.

So far ANML has been demonstrated in a situation where each class/task is simple (learning one Omniglot character type). New research is needed to see how well ANML and ANML-inspired approaches scale to much more complex tasks. One example is extending the work to reinforcement learning (RL) tasks. If ANML helps in that setting it would be important since RL agents naturally encounter sequential learning environments, and are currently held back by all of the challenges that come with them. Another dimension of difficulty that can be tested is when the meta-test distribution is different from the meta-training distribution. If the meta-training distribution is large enough, it would be interesting to test whether ANML can learn to continuously learn in a way that generalizes to entirely new types of tasks that are not in the meta-training set.

Another potentially profitable direction is to hybridize the insights of ANML with other meta-learning techniques. Two promising ones are combining ANML with the style of meta-learning wherein the entire learning algorithm is meta-learned within a recurrent neural network (either in a supervised [14] or RL context, e.g. RL^2 [6] aka Learning to Reinforcement Learn [43]). Another is combining ANML with recent work into meta-learning with differentiable Hebbian learning [33], including differentiable neuromodulated Hebbian plasticity [34].

Most broadly, the success of OML and ANML underscore the power of meta-learning the solutions to the hardest machine learning problems, such as exploration, safe exploration, generalization, robustness to adversarial examples, and many more. They thus increase our confidence in paradigms like AI-generating algorithms (AI-GAs), which advocate learning as much of the solution as possible [5]. This work focuses on Pillar Two of AI-GAs (meta-learning learning algorithms). The power of the ANML approach should only increase when combined with the other pillars of the AI-GA paradigm, namely architecture search and automatically generating training environments. Architecture search is especially interesting to consider with ANML, because there are so many possible architectures that could be tried and some will likely generate substantial improvements. We could try to investigate that question manually, or we could instead turn to architecture search to discover the answer for us automatically. That will become increasingly attractive as architecture search methods improve.

6 CONCLUSION

This work introduces ANML, a method to improve continual learning by directly optimizing for improved continual learning. Here, we have demonstrated the benefits of ANML for reducing catastrophic forgetting. ANML is motivated by the idea that we should meta-learn the solutions to hard problems, instead of manually engineering machine learning solutions for them. Specifically, ANML meta-learns the parameters of a neuromodulatory network that, conditioned on data, gates the activations of a separate prediction network, creating selective activation and, in turn, selective plasticity. The results presented here demonstrate the effectiveness of this approach to reduce the amount of catastrophic forgetting relative to traditional and current state-of-the-art methods at an unprecedented scale, demonstrating continual learning on trajectories of up to 600 sequentially learned classes (over 9,000 SGD updates). Although work needs to be done to test how well ANML works on harder challenges, this work provides a promising stepping stone towards improved algorithms for continual learning. It also underscores the value of learning as much of the solution as possible, and thus adds momentum to the growing perspective that we should meta-learn solutions to the grand challenges of AI research. That includes pursuing AI-generating algorithms, which attempt to learn as much as possible in the pursuit of our community's grandest ambition: creating artificial general intelligence.

ACKNOWLEDGEMENTS

This work is supported in part by DARPA Lifelong Learning Machines award HR0011-18-2-0018. We thank Hava Siegelmann for her vision in creating that program and for including us in it. We also thank Khurram Javed for providing the OML code and answering clarifying questions about their experiments. Computations were performed on the Vermont Advanced Computing Core (VACC) supported in part by NSF award No. OAC-1827314. We would also like to thank the Vermont Complex Systems Center for assistance, encouragement, and feedback. We are also appreciative of Blake Camp for catching typos in a draft and alerting us to a relevant paper, and to Louis Kirsch for suggesting the addition of a relevant citation.

REFERENCES

[1] Mark F Bear and Wolf Singer, 'Modulation of visual cortical plasticity by acetylcholine and noradrenaline', *Nature*, **320**(6058), 172, (1986).
[2] Emmanuel Bengio, Pierre-Luc Bacon, Joelle Pineau, and Doina Precup, 'Conditional computation in neural networks for faster models', *arXiv preprint arXiv:1511.06297*, (2015).
[3] Yoshua Bengio, 'Deep learning of representations: Looking forward', in *International Conference on Statistical Language and Speech Processing*, pp. 1–37. Springer, (2013).
[4] Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber, 'Multi-column deep neural networks for image classification', in *2012 IEEE conference on computer vision and pattern recognition*, pp. 3642–3649. IEEE, (2012).
[5] Jeff Clune, 'AI-GAs: Ai-generating algorithms, an alternate paradigm for producing general artificial intelligence', *arXiv preprint arXiv:1905.10985*, (2019).
[6] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel, 'RL^2: Fast reinforcement learning via slow reinforcement learning', *arXiv preprint arXiv:1611.02779*, (2016).
[7] Kai Olav Ellefsen, Jean-Baptiste Mouret, and Jeff Clune, 'Neural modularity helps organisms evolve to learn new skills without forgetting old skills', *PLoS computational biology*, **11**(4), e1004128, (2015).
[8] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra, 'Pathnet: Evolution channels gradient descent in super neural networks', *arXiv preprint arXiv:1701.08734*, (2017).
[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine, 'Model-agnostic meta-learning for fast adaptation of deep networks', in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, (2017).
[10] Evelyn Fix and Joseph L Hodges Jr, 'Discriminatory analysis-nonparametric discrimination: consistency properties', Technical report, California Univ Berkeley, (1951).

[11] Robert M French, 'Using semi-distributed representations to overcome catastrophic forgetting in connectionist networks', in *Proceedings of the 13th annual cognitive science society conference*, pp. 173–178, (1991).

[12] Robert M French, 'Catastrophic forgetting in connectionist networks', *Trends in cognitive sciences*, **3**(4), 128–135, (1999).

[13] Michael E Hasselmo and Edi Barkai, 'Cholinergic modulation of activity-dependent synaptic plasticity in the piriform cortex and associative memory function in a network biophysical simulation', *Journal of Neuroscience*, **15**(10), 6592–6604, (1995).

[14] Sepp Hochreiter, A Steven Younger, and Peter R Conwell, 'Learning to learn using gradient descent', in *International Conference on Artificial Neural Networks*, pp. 87–94. Springer, (2001).

[15] Sergey Ioffe and Christian Szegedy, 'Batch normalization: Accelerating deep network training by reducing internal covariate shift', *arXiv preprint arXiv:1502.03167*, (2015).

[16] Akio Ishiguro, Akinobu Fujii, and Peter Eggenberger Hotz, 'Neuromodulated control of bipedal locomotion using a polymorphic cpg circuit', *Adaptive Behavior*, **11**(1), 7–17, (2003).

[17] Khurram Javed and Martha White, 'Meta-learning representations for continual learning', *Neural Information Processing Systems*, (2019).

[18] Diederik P Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, (2014).

[19] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al., 'Overcoming catastrophic forgetting in neural networks', *Proceedings of the national academy of sciences*, **114**(13), 3521–3526, (2017).

[20] Soheil Kolouri, Nicholas Ketz, Xinyun Zou, Jeffrey Krichmar, and Praveen Pilly, 'Attention-based selective plasticity', (2019).

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, 'Imagenet classification with deep convolutional neural networks', in *Advances in neural information processing systems*, pp. 1097–1105, (2012).

[22] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum, 'Human-level concept learning through probabilistic program induction', *Science*, **350**(6266), 1332–1338, (2015).

[23] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al., 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE*, **86**(11), 2278–2324, (1998).

[24] Xuhong Li, Yves Grandvalet, and Franck Davoine, 'Explicit inductive bias for transfer learning with convolutional networks', *arXiv preprint arXiv:1802.01483*, (2018).

[25] Zhizhong Li and Derek Hoiem, 'Learning without forgetting', *IEEE transactions on pattern analysis and machine intelligence*, **40**(12), 2935–2947, (2017).

[26] Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov, 'Rotate your networks: Better weight consolidation and less catastrophic forgetting', in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 2262–2268. IEEE, (2018).

[27] Vincenzo Lomonaco and Davide Maltoni, 'Core50: a new dataset and benchmark for continuous object recognition', *arXiv preprint arXiv:1705.03550*, (2017).

[28] Laurens van der Maaten and Geoffrey Hinton, 'Visualizing data using t-sne', *Journal of machine learning research*, **9**(Nov), 2579–2605, (2008).

[29] Dougal Maclaurin, David Duvenaud, and Ryan Adams, 'Gradient-based hyperparameter optimization through reversible learning', in *International Conference on Machine Learning*, pp. 2113–2122, (2015).

[30] Davide Maltoni and Vincenzo Lomonaco, 'Continuous learning in single-incremental-task scenarios', *Neural Networks*, **116**, 56–73, (2019).

[31] Henry B Mann and Donald R Whitney, 'On a test of whether one of two random variables is stochastically larger than the other', *The annals of mathematical statistics*, 50–60, (1947).

[32] Nicolas Y. Masse, Gregory D. Grant, and David J. Freedman, 'Alleviating catastrophic forgetting using context-dependent gating and synaptic stabilization', *Proceedings of the National Academy of Sciences*, **115**(44), E10467–E10475, (2018).

[33] Thomas Miconi, Jeff Clune, and Kenneth O Stanley, 'Differentiable plasticity: training plastic neural networks with backpropagation', *arXiv preprint arXiv:1804.02464*, (2018).

[34] Thomas Miconi, Aditya Rawal, Jeff Clune, and Kenneth O Stanley, 'Backpropamine: training self-modifying neural networks with differentiable neuromodulated plasticity', (2018).

[35] Hieu Pham, Melody Y Guan, Barret Zoph, Quoc V Le, and Jeff Dean, 'Efficient neural architecture search via parameter sharing', *arXiv preprint arXiv:1802.03268*, (2018).

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

[37] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver, 'Prioritized experience replay', *arXiv preprint arXiv:1511.05952*, (2015).

[38] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean, 'Outrageously large neural networks: The sparsely-gated mixture-of-experts layer', *arXiv preprint arXiv:1701.06538*, (2017).

[39] Andrea Soltoggio, John A Bullinaria, Claudio Mattiussi, Peter Dürr, and Dario Floreano, 'Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios', in *Proceedings of the 11th international conference on artificial life (Alife XI)*, number CONF, pp. 569–576. MIT Press, (2008).

[40] Andrea Soltoggio, Kenneth O Stanley, and Sebastian Risi, 'Born to learn: the inspiration, progress, and future of evolved plastic artificial neural networks', *Neural Networks*, **108**, 48–67, (2018).

[41] Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci, 'A hypercube-based encoding for evolving large-scale neural networks', *Artificial life*, **15**(2), 185–212, (2009).

[42] Roby Velez and Jeff Clune, 'Diffusion-based neuromodulation can eliminate catastrophic forgetting in simple neural networks', *PloS one*, **12**(11), e0187736, (2017).

[43] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick, 'Learning to reinforcement learn', *arXiv preprint arXiv:1611.05763*, (2016).

[44] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson, 'How transferable are features in deep neural networks?', in *Advances in neural information processing systems*, pp. 3320–3328, (2014).

[45] Friedemann Zenke, Ben Poole, and Surya Ganguli, 'Continual learning through synaptic intelligence', in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 3987–3995. JMLR. org, (2017).

[46] Jianming Zhang, Sarah Adel Bargal, Zhe Lin, Jonathan Brandt, Xiaohui Shen, and Stan Sclaroff, 'Top-down neural attention by excitation backprop', *International Journal of Computer Vision*, **126**(10), 1084–1102, (2018).

[47] Barret Zoph and Quoc V Le, 'Neural architecture search with reinforcement learning', *arXiv preprint arXiv:1611.01578*, (2016).