Testing ANML in the wild

Marc-Andre Piche
Mila, University of Montreal
marc-andre.piche@mila.quebec

Abstract

Continual learning (CL) requires an agent or model to learn many tasks sequentially without forgetting previous task. With the recent success of Neuromodulated Meta-Learning Algorithm (ANML) [1] on learning classes sequentially without catastrophic forgetting, its potential begged to be explored. Mainly, as it was the original purpose behind this work; can ANML be adapted to a task-agnostic setting? However, by reviewing its experimental formulation, our work revealed weaknesses which might invalidate some of the paper's findings. Nevertheless, the ANML model formulation achieved impressive state of the arts results on a difficult benchmark, meaning we should explore further what discovery was made by [1].

1 Introduction

Building upon Online aware Meta-Learning (OML) [2] that employs a MAML[3]-style meta-learning to avoid manually designed solutions, but rather use optimization to reach the desired model, ANML instead set the outer-loop network (frozen during meta-testing) and inner-loop network in parallel rather than sequentially. Inspired by neuromodulatory [4] processes in the brain, ANML differentiates through a sequential learning process to meta-learn an activation-gating function that enables context dependent activation within a neuromodulatory (NM) neural network. Specifically, it gates the forward pass of another neural network, but also the backpropagation and thus indirectly controls selective plasticity. ANML's authors claim that rather than meta-learn representations, ANML meta-learns a context-dependent gating function. We will see that might not necessarily the case. Both OML[2] and ANML encourage sparse representations rather than enforce it, which surely a key to their success.

1.1 Neuromodulation

With over 7 million parameters, the ANML[1] network is over-complete. But we've known for decades that doesn't change catastrophic forgetting, it happens anyway to almost the same degree. How and where information is learned is just as important as sparsity. We can even imagine that with their success at learning 600 classes, ANML[1] is not even close to testing the limits of its capacity. This why selectively gating how information is learned in an intelligent is so important. This is the key idea of neuromodulation[4] as it changes the learning rate of other neurons and thus enables task-specific learning. To achieve this, the output neurons of the NM network are mapped one-to-one with the output of the prediction network with pointwise multiplication. The purpose is to control the flow of the gradient back propagated through the prediction network, and therefore controls which parts of the network gets updated with selective plasticity (Figure 1).

*Work done in the context of the IFT6760B class on continual learning
.

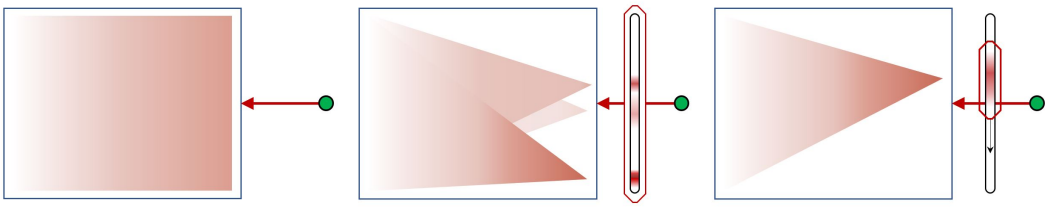

Figure 1: Selective plasticity: normal back propagation (left), soft gating (center), sliding gate (right)

2 Continual Learning Settings

2.1 Task incremental vs Class incremental

OML[2] and ANML[1] set their experiment by learning a random sequence of Omniglot[5] characters with 15 examples of the same character seen at a once, and only one pass through the dataset. While still impressive that their network can learn in this fashion, this experimental setup invalidates some of their claim about continual learning. We were initially in agreement with AMNL's author that in a continual learning setting where we simply learn a great number of classes should generalize to all continual learning settings, since we could always put those classes in a bin. But, in the ANML[1] & OML[2] experimental settings, a task is a class and vice versa. Meaning the network is only required to tag the correct bit out of a large vector. Not a trivial task in itself, but since we have the labels, we therefore have the tasks and so automatically the distinction between each task boundary, which is a continual learning challenge of importance that is set aside and ignored. Why is this important? Because the bottlenecks of the network force it to confuse representations, and that first bottleneck is the output vector. So, in theory it could be that the smaller the output vector, the more it is susceptible to catastrophic forgetting.

2.1.1 Randomized sequence of Tasks

The most important flaw in OML & ANML's experimental setup, is that by ignoring task boundaries, they also ignore the internal distribution of a task. In this setting a randomized "task order" has an impact. Each alphabet has its own internal distribution where characters resemble one another compared to another alphabet, but also some alphabets greatly resemble each other. So even if they learn characters one at a time with 20 examples each, these characters differ greatly. Meaning that by randomizing the sequence, not only is the dataset in an almost i.i.d. setting but also the network can see an example more than once. This setting is more akin to one-shot learning than continual learning. Our experiments reveal how this can impact performance.

2.1.2 Known number of classes in advance

Finally, in the experimental model (in their code), OML[2] ANML[1] output their answer in a vector of size 1000 to be over their needs of 963 classes. Which means the network knows how many classes there are ahead of time. Which we cannot say is a true continual learning setting as this is not really dealing with an unknown number of classes. But more importantly, it is unknown what clues it might receive from this gradient. We cannot say if this an advantage to the network, or if it is in fact making the task much harder as it falls in the formulation of extreme classification. After all, getting the correct output after a SoftMax on a such a long output vector is quite impressive to achieve at a 64% rate.

2.2 Our experimental Setup

2.2.1 Task boundaries Hard, Soft, Softer, Blurred

In order to test ANML[1] in a correct continual learning setting, we ran all our experiments with the same sequential ordered pass over the data, one alphabet at a time. To test the formulation in a multitask setting we divided the Omniglot[5] alphabets modulo k. A task being defined as learning

k characters from the same alphabet. We tested $k \in 2, 4, 5, 8, 13$ as these divisions conserved the most data. We keep modulo 8 as our main candidate for extensive experiments to compare with the original paper, as training and testing these models demands extensive amounts of computation. We defined and tested on two task boundary settings in this work, hard task boundary where all examples of a task are seen only during the training of that task, and soft task boundary where examples from previous and future tasks can be encountered with a certain probability. In the soft task boundary, examples are distributed around a certain μ_t for task t. All time steps μ where evenly spaced, but we could imagine a more general setting where this might not be the case. Varying σ_t also has a huge impact on training and testing, as a higher variance brings the distribution closer to i.i.d., which helps in training if we know the task labels, but makes task labelling much harder at test time. This soft task definition should cover most continual learning challenges depending on the parameters. For example, if we have tasks that are very close in definition to one another, with a high σ_t, we are then in the distribution drift setting which is the most difficult continual learning setting. The less the distinction between contexts is clear, the more a CL model is vulnerable against forgetting. We tested our network for both known and unknown task labels at training time.

2.2.2 Local vs Global Accuracy

It is confusing to say a network has many tasks. Usually there is only one task and it is to produce the correct output of fixed length. So why does multi-output tasks matter so much compared to just learning representations? Mainly it is because the environment is part of the classification. Depending on the environment, the correct output could vary greatly. Looking at the original experimental setup of OML[2] & ANML[1] of correctly classifying 600 characters is much more difficult than getting the correct output for tasks of 8 characters. However, they are equivalent when the network must also correctly label the task to get rewarded for the correct answer. The question here is: If I you do something right for the wrong reasons, should you get rewarded? What should you learn from such a choice? In the context of an ANN, global accuracy where both task and character labels must be correct should be used to measure the network performance. But only the local accuracy loss and gradient should be back propagated through the correct head. Otherwise mistakes would pollute our network. If task boundary is unknown at training time, we obviously cannot do this, so only local accuracy should be considered overall. This adds yet more in getting high task labelling accuracy. In summary, we use global accuracy the performance result, and use local accuracy for learning.

3 Implementation

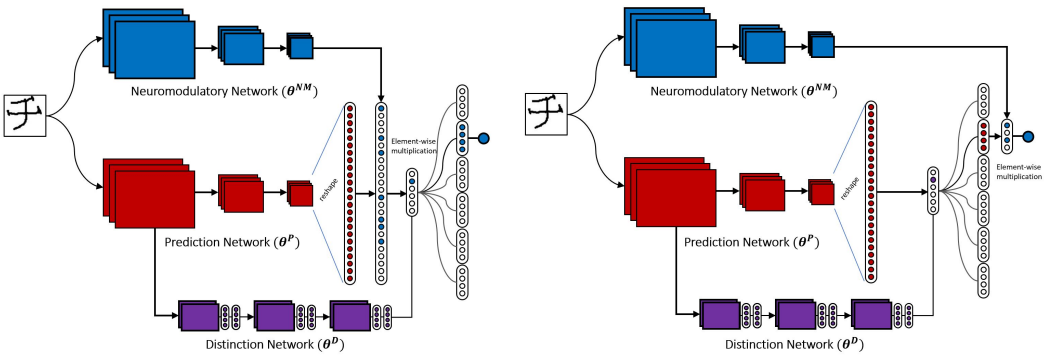

Figure 2: Wide and Narrow neuromodulation.
Gating the representations (left); Gating the heads (right)

3.1 Cellular Division Model

Taking inspiration from the Expert Gate Model[6] we devised a distinction network to correctly identify the correct task. In order to keep the number of parameters low as possible in this dynamically expanding setting, the distinction network identify the correct branch in a tree of tasks. The distinction network creates a new head for each leaf added in the tree. So that the decision of this

network is as discriminative as possible, the task labels are assigned dynamically into the branches of the tree by branching the task with the initial task it was confused with. To be clear; when a new task is assigned or guessed in the case where task boundaries are unknown, it is placed in the tree where it must be distinguished with the most similar task, simply chosen by making the mistake. For each layer in the tree, a small new network is created that classify all tasks into two bins similar to a SSVM. The binary outputs of these networks are the task label.

In an ideal model, we wouldn't want to just select the correct head, we would want to use this information at the input of the prediction network. Even better, the last layer of each node in the distinction network is rich in information. These could be concatenated into a context vector. We have searched extensively for a model formulation that would allow this, but with the context vector's progressively changing and ever-increasing cause particular problems for both continual and meta-learning.

Our distinction network is quite simple in formulation as its task on the Omniglot[5] dataset is quite easy. But this could be the most sophisticated part of a network as identifying the correct context or environment might the most difficult and important task to solve. For example, when we tested our network on unknown task labels at training time (one of the hardest challenges in continual learning), it expanded unrestrained by assigning a new task for almost all new characters encountered. We used a formulation similar to Neural Dirichlet Process[7] to accumulate k characters in a bin. It only managed to correctly detect new task when σ_t was so narrow that it became a hard task boundary. If we had a differentiable function or network charged with the decision of creating a new head, we could include a hierarchical SSVM loss as a regularizer against expansion.

3.2 Two neuromodulation targets

The first design choice that had to be made for a multi-headed network is where to target the neuromodulation, since in the original paper the gating is after the last activation. We could keep this design by gating before the heads or we can gate on the heads themselves. Meaning in the later that NM network output has much less expressivity but it also better target. Since we cannot know which might be the best case, we created and tested both designs. The head-gating might also lead to a design that could address distribution drift by sliding a window along the output vector (Figure 1).

4 Results

Using the same training procedure as OML[2] & ANML[1] to test performance in meta-testing, we tested our network on a trajectory of f 10, 50, 75, 100, 150, 200, 300, 400, 500, and 590 classes to show how the models scale to longer sequential task trajectories. A hyper-parameter search is performed for each sequence length to set the best learning rate β (3 out of 16) for its inner-loop updates. However, it soon became clear with the results that something was wrong, and so we only tested and compared for a full 590-class trajectory. The resulting performance out our ANML_k8 (8 classes per alphabet) is a surprisingly low 23% with a task detection accuracy of 87%. Considering the odds of getting the right answer with a task detection at 100%, we tested our multiheaded model with an oracle that provides task label at test time. The resulting 30%, of what should be an easy task, exactly the same result in both narrow and wide neuromodulation (gating on representation or on the head), reveals that this way of implementing neuromodulation does not work as intended. By running the original ANML[1] on our continual learning setting resulted in only an 8% performance for one learning rate, and 1 to 2% for all others. Since the main difference in settings is the order of classes in the sequence, by alphabets/tasks or randomized i.i.d., we can conclude that this is what affects learning so much. What this experiment shows is that although ANML[1] is very good at one-shot/20 learning and mitigates catastrophic forgetting, it must learn in a randomized distribution in order to perform. Table 1 compares the method that we ran with full computation time.

4.1 Alternate view of ANML

The true result of this experiment is that although ANML[1] is not a CL network, it can be viewed as something that has its own wonderful potential. By examining the networks from other angles, we can get into some interesting hypothesis explaining all these successes and failures. First, we can

Table 1: Results

	Meta-Training	Meta-Testing
ANML (paper)	–	64%
ANML_k8	98%	30%
ANML_k8 (head gating)	98%	30%
ANML (CL setup)	78%	8%

easily imagine the NM network learning all the classes of the meta-training set on its own. Especially with the 40000 training steps that amounts to 2000 meta-training loss updates that sees 2000 classes with 20 examples each. Which means we could replace the prediction network with a vector of 1s in the meta-training. This might be exactly what it is doing. Instead of learning to gate the learning of the prediction network, so that there is learning to learn at play as intended for meta-learning. The network learns to be robust to interference that is similar to noise or drop-off from another network. In fact, how to tell which network interferes with which? Both are evolving in tandem in a sort of adversarial and collaborative fashion. During meta-test time, we could say these are young and old networks working together to learn as easily as possible. This raises the question of whether the prediction network is needed during training or if they could be trained separately to do away with all that expensive meta-training computing time and maybe even use a powerful pre-trained network for neuromodulation. We might even be able to skip meta-training altogether. Finally, if that is the case, then these networks could be chained for some unknown potential.

5 Conclusion

We are coming out of this work with both important revelations but also intriguing questions that will take much more experimentation to answer. We now know that ANML[1] doesn't really learn to learn as is intended with meta-learning, but rather learns with a preexisting complementary set of representations coming from the same distribution. That in itself is not bad at all as viewing how this method operates correctly opens the door for untapped potential. It certainly shows that we should be suspicious of claims and interpretations without carefully looking at the experimental setting just as much as we should keep an open mind.

All code & slides available on GitHub : maPiche/ANML-Project

References

[1] Shawn Beaulieu, Lapo Frati, Thomas Miconi, Joel Lehman, Kenneth O. Stanley, Jeff Clune, and Nick Cheney. Learning to continually learn, 2020.

[2] Khurram Javed and Martha White. Meta-learning representations for continual learning, 2019.

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

[4] Andrea Soltoggio. Neuromodulation increases decision speed in dynamic environments. 139, 01 2008.

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

[6] Rahaf Aljundi, Punarjay Chakravarty, and Tinne Tuytelaars. Expert gate: Lifelong learning with a network of experts. *CoRR*, abs/1611.06194, 2016.

[7] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning, 2020.