Can EWC overcome task saturation?

Bright Elyon
brelng14@gmail.com

Charles Lagacé
thecharleslagace@gmail.com

Abstract

Elastic Weight Consolidation (EWC) showed promising results to mitigate catastrophic forgetting in neural networks. However, the memory constraint becomes ill-defined as the number of tasks increases, and experiments in the original paper only train up to 10 tasks. This is important because we expect that a trade-off between task saturation and catastrophic forgetting will compromise model performance once the task number is sufficiently large. In this work, we define a novel task sampling procedure on the CIFAR100 dataset, which allows to obtain a very large number of tasks. Using this paradigm, we trained EWC and some baseline models on a total of 100 tasks. Our experiments show EWC enters a task saturation regime during the second half of training.

1 Introduction

When trained on a sequence of task, neural networks typically suffer from catastrophic forgetting [1, 2]. This is because important weights from previous tasks are updated to meet the objectives of the new task. However, the mammalian brain can avoid catastrophic forgetting thanks to synaptic consolidation, a process which reduces the plasticity of important synapses [3, 4].

Analogous to synaptic consolidation, [5] proposes the Elastic Weight Consolidation (EWC) algorithm, which slows down learning on weights that were important for previous tasks. This work became a pioneering approach in the continual learning literature. The key idea of EWC [5] is to remember old tasks by selectively penalizing gradients on the weights that were important for those tasks. More formally, given a previous task A with optimal parameters θ_A^*, and a new task B with optimal parameters θ_B^*, EWC protects the weights learned on task A by constraining the parameters θ_B^* to be similar to θ_A^*. Importantly, this penalty should be greater for parameters that were more important for task A. This is implemented using the diagonal of the Fischer information matrix, F:

$$L(\theta) = L_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta_{A,i}^*)^2 \tag{1}$$

Where $L_B(\theta)$ is simply the loss on the new task, and the penalty term $\sum_i \frac{\lambda}{2} F_i (\theta_i - \theta_{A,i}^*)^2$ can be thought of as a prior-based constraint, which defines which directions of parameter variation are expected to cause severe performance degradation on the previous task. However, this penalty becomes ill-defined as the number of tasks increases. This is because we would need a separate penalty to embed the sub-components of the network which are deemed important for every single task. For example, suppose that we don't have 2 tasks, but 100 tasks, $T \in \{T_1, T_2, ..., T_{100}\}$. The combined loss function when training on the 100th task in a continual learning setting would then

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

become:

$$L(\theta) = L^{(T_{100})}(\theta) + \sum_i \frac{\lambda}{2} F_i^{(T_1)} (\theta_i - \theta^*_{T_1,i})^2 + \sum_i \frac{\lambda}{2} F_i^{(T_2)} (\theta_i - \theta^*_{T_2,i})^2 + ... + \sum_i \frac{\lambda}{2} F_i^{(T_{99})} (\theta_i - \theta^*_{T_{99},i})^2 \tag{2}$$

We hypothesize that, if one were required to train a model on a large number of tasks, this penalty formulation would be prone to task saturation. This is because the additive nature of single-task penalties, which is required to memorize each previous task, will overshadow the loss of the new task if the regularization coefficient λ is sufficient big. Thus, the optimization path will favor small gradients in most directions, and, assuming a constraint of fixed parameter capacity, the model will be unable to learn a suitable representation of the new task. On the other hand, if λ is too small, the model might degrade to a catastrophic forgetting regime. It is unfortunately impossible to observe any evidence or refutation of this task saturation hypothesis in the original EWC paper, since they stopped training after 10 tasks of the permuted MNIST benchmark.

In this work, we investigated this trade-off between task saturation and forgetting. In order to do so, we designed a task sampling procedure on the CIFAR100 dataset which allows to train on several thousands of tasks. This method allows to really test EWC to its limits, and observe how the performance across tasks might change as the algorithm progresses through this extensive training. Our main objective is to conduct a robustness experiment on EWC, with the trade-off between task saturation and catastrophic forgetting in mind.

2 Related work

In the continual learning literature, EWC is often classified as a prior-based method. Other methods of this same family include SI [6], MAS [7], and, Riemannian Walk [8]. A common theme with prior-based methods is that, since they require a memory trace of every previously encountered task, it becomes increasingly hard with a larger number of tasks to make this memory trace sufficiently constraining in order to preserve critical information from each of these tasks, but not as constraining as to impede learning new tasks. Thus, other prior-based methods are likely to suffer from a similar trade-off than EWC, between task saturation and forgetting.

Riemannian Walk [8] is interesting because it also provides an algorithmic definition of intransigence, or the inability of a model to update its knowledge, which is closely related to task saturation. For this reason, [8] might be more robust to task saturation, but it is unclear whether a proper balance between intransigence and forgetting can be achieved in practice, especially when the number of tasks is sufficiently large.

In a similar line of thought than our EWC critique, [9] identified some frequent shortcomings of continual learning papers, in terms of the robustness of performance evaluation. They pointed out that popular benchmarks such as Permuted MNIST [10] and Split MNIST [6] tend to make continual learning easier for prior-based methods, since the distribution shift between tasks is fairly small. Thus, memory traces from previous tasks are more likely to be already well-aligned with an efficient representation of future tasks, even though the model has never seen any sample from the future tasks beforehand. Evaluating EWC on tasks derived from CIFAR100 instead of Permuted MNIST will also allow to test the algorithm in a more natural training environment.

3 Methods

In order to have a very large number of tasks, we define a task sampling procedure on the CIFAR100 dataset, which can be summarized as follows:

1. Define the list T of all possible binary classification tasks.
2. Randomly select without replacement 50 class pairs, without replacement of the individual CIFAR100 classes. Remove these class pairs from T.

3. Repeat until exhaustion of all the tasks in T.

We note that approach allows a total of 4950 tasks. Nevertheless, we decided to use only 100 tasks for our experiments, because we observed that the training behaviour of our models was already stable at this number of task. Also, we decided to use CIFAR100 as opposed to a larger dataset that also contains many classes, such as ImageNet, because we wanted to evaluate the robustness of EWC within a reasonable computational budget.

In addition to EWC, we implemented two baseline models to compare performance: the naive model and the foolish model. When it reaches a new task, the naive model keeps training on the parameters from the previous task, without any form of regularization. On the other hand, the foolish approach re-initializes the parameters randomly after each task. Because of this, the foolish model has no memory of previous tasks. We note that the foolish also doesn't have any form of regularization.

Finally, in terms of the model architecture, we used a convolutional neural network (CNN) with the following signature:

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 10, kernel_size=5)
        self.conv2 = nn.Conv2d(10, 20, kernel_size=5)
        self.conv2_drop = nn.Dropout2d()
        self.fc1 = nn.Linear(500, 50)
        self.fc2 = nn.Linear(50, 10)

    def forward(self, x):
        x = F.relu(F.max_pool2d(self.conv1(x), 2))
        x = self.conv2_drop(self.conv2(x))
        x = F.relu(F.max_pool2d(x, 2))
        x = x.view(-1, 320)
        x = F.relu(self.fc1(x))
        x = F.dropout(x, training=self.training)
        x = self.fc2(x)
        return x
```

The models were trained using a stochastic gradient with a learning rate of 0.01 and a momentum coefficient of 0.9. In all cases, we used the accuracy metric to measure model performance.

4 Results

First, we measured the average accuracy of each model across all tasks, after training for some number of tasks (Figure 1). Contrary to our expectations, we do not observe any memory overhead with EWC. Indeed, the average accuracy already degrades to random performance after 10 tasks. We think that this could be due to the relatively small sample size in each of our tasks. Indeed, the size of the training partition of CIFAR100 is 50,000 samples, of which 500 samples belong to each class. Thus, each of our tasks, which exactly contains the samples from 2 of the CIFAR100 classes, only has 1000 samples. Given this small sample size, we think that the Fischer Information Matrix estimate from EWC might not be precise enough to effectively enforce a memory constraint on the previous tasks.

Another reason which might cause this rapid performance degradation of EWC is the distribution shift between different tasks. Indeed, in the original paper, EWC was evaluated on the Permuted MNIST benchmark, which always has small distribution shift as mentioned in [9]. However, CIFAR100 does not provide EWC with such a controlled environment, and the classification problem might vary from discrimating cars and airplanes to discriminating cats and dogs. It is possible that EWC just isn't robust to large distribution shifts between tasks.

We then measured the accuracy of the last task only, in order to investigate model saturation (Figure 2). We observed that, even though there can be huge systematic performance variations between different tasks, the foolish model gets the best accuracy most of the time. This makes a lot of sense

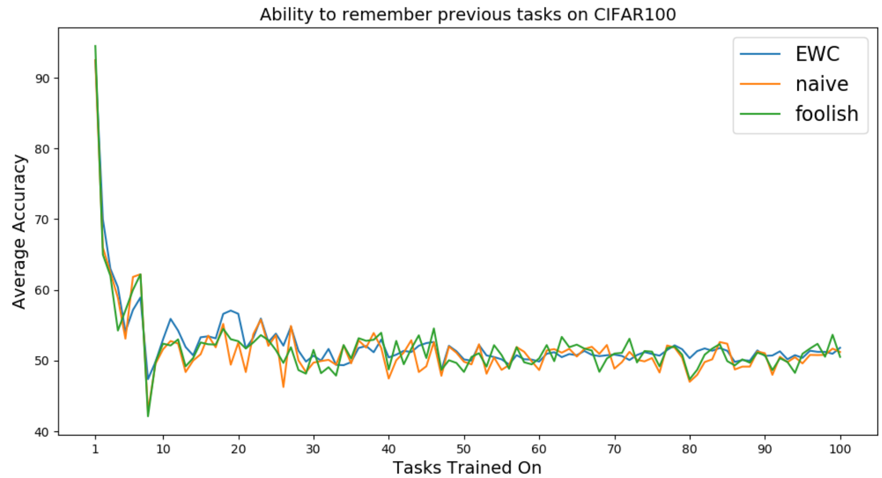

Figure 1: Ability to remember previous tasks on the CIFAR100 dataset. The average accuracy across all encountered tasks is measured against the number of tasks that the models have been traines on so far. EWC: Elastic Weight Consolidation.

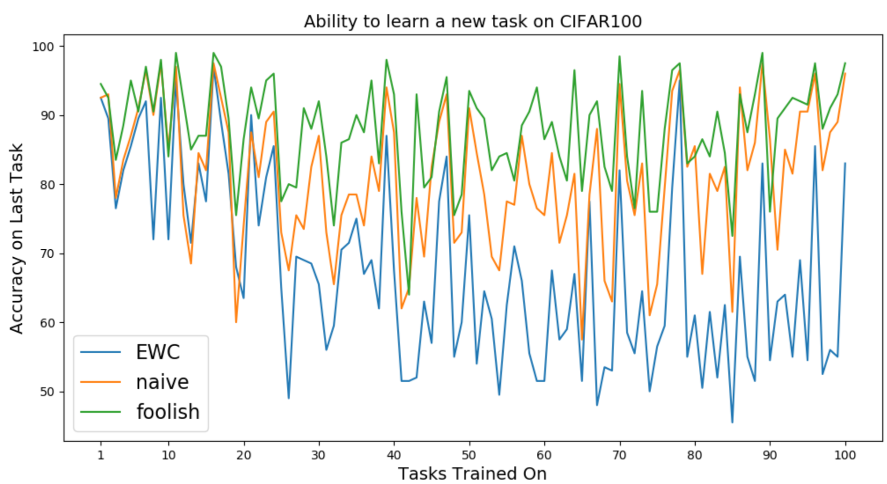

Figure 2: Ability to learn a new task on the CIFAR100 dataset. The accuracy on the last task the models were trained on is plotted against the total number of training tasks so far. EWC: Elastic Weight Consolidation.

because, since the foolish doesn't have any memory of the previous tasks, it should not exhibit any task saturation behaviour. We note that the naive model tends to have slightly worse accuracy than the foolish model. Since the naive model also doesn't have any mechanism that could introduce task saturation, this might just mean that, on average, random initialization is a better prior for model learning than the optimal weights of a previous tasks.

In the case of EWC, we find that during the first 30 tasks of training, the accuracy on the last task seems quite similar to the naive model. However, once we get beyond that point, we start to observe much lower accuracy, and the accuracy is often not much better than random for EWC during the second half of training. This suggests that task saturation is indeed happening, but several dozens of tasks are required before it becomes a major issue. It would be interesting to investigate how the dataset size or the distribution shift between tasks might affect the number of tasks required before EWC enters a task saturation regime.

Figure 3: Ability to remember recent tasks on the CIFAR100 dataset. The accuracies up to a lag of 9 tasks are shown for each multiple of 10 tasks trained on. The most recent tasks are shown with opaque lines, and the first encountered tasks are shown with the more transparent lines. EWC: Elastic Weight Consolidation.

Finally, because the accuracy on previous tasks observed in Figure 1 was extremely low, even after only a few tasks, we decided to investigate the short-term memory of EWC. The accuracies on the previous 10 tasks after training for some number of tasks are shown in Figure 3. What we see is that, even when looking only 2 or 3 tasks in the past, the performance is already almost random. This reinforces our belief that, with the current experimental setting, the Fischer Information Matrix estimate might not be precise enough to effectively enforce a memory constraint.

5 Conclusion

To conclude, we performed a robustness experiment on EWC using a novel task sampling procedure on the CIFAR100 dataset. By randomly selecting pairs of CIFAR100 classes, we were able to train the model on 100 distinct tasks. We observed that, after about 30 tasks of training, EWC starts to exhibit task saturation, and it becomes very severe in the second half of training. This suggests that a proper balance between task saturation and forgetting fails to be achieved when the number of tasks is too large. We also observed, to our greatest surprise, that EWC didn't seem to remember previous tasks any better than the naive model or the foolish model, and we identified two possible reasons for this: first, the sample size could be too small to have a good Fischer Information Matrix estimate; second, EWC might simply not robust to large distribution shifts. To investigate this first hypothesis, one could replace CIFAR100 with the ImageNet dataset, and implement a similar task sampling procedure to get a large number of task. However, using such a large dataset as ImageNet would significantly increase the computational resources required to perform these experiments.

Additionally, we mentioned that prior-based methods in general are at risk of suffering from task saturation when the number of tasks is large. An interesting future experiment would be to test some of these other methods using a similar task paradigm. In particular, since Riemannian Walk introduces an algorithmic definition of intransigence, it might fare a little bit better than competing algorithms in that regard. Finally, we propose that our task sampling procedure can be replicated for any continual learning approach, possibly with a larger dataset such as ImageNet, in order to properly evaluate its robustness to task saturation, as well as other issues that may arise from a large number of tasks. In the long-term agenda of continual learning, it will be necessary to develop artificial systems that can not only prevent catastrophic forgetting, but also enable knowledge transfer from a large body of previous tasks to learn new tasks more efficiently.

References

[1] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*, 2019.

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

[3] Stefano Fusi, Patrick J Drew, and Larry F Abbott. Cascade models of synaptically stored memories. *Neuron*, 45(4):599–611, 2005.

[4] Marcus K Benna and Stefano Fusi. Computational principles of biological memory. *arXiv preprint arXiv:1507.07580*, 2015.

[5] 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.

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

[7] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154, 2018.

[8] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018.

[9] Sebastian Farquhar and Yarin Gal. Towards robust evaluations of continual learning. *arXiv preprint arXiv:1805.09733*, 2018.

[10] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.