An Optimization View of Elastic Weight Consolidation

Yang Yiliu

Dec 2022

Abstract

This serves as a further study of the ESTR3112 project to introduce another way to understand Elastic Weight Consolidation (EWC)[1] from an optimization perspective. This is a little different from the project report with some minor changes.

1 Introduction

Online learning is a learning problem where training data comes in sequential order. A trivial solution is to retrain the whole model when new data comes. This method usually costs many resources. Another solution is introducing L_2 regularization, which means introducing another penalty term $(\Delta\theta)^2$ and directly training on the new data. This method usually performs poorly.

Elastic Weight Consolidation (EWC)[1] is a regularization method for online learning introduced by DeepMind in 2016. Its basic idea is to follow L_2 regularization but weigh different parameters separately.

In detail, under the situation of online learning, we have already finished the optimization problem on task A by solving $\theta_A^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_A(\theta)$. Here comes another task B and we want to finish another optimization problem on both task A and B by solving $\theta^* = \underset{\theta}{\operatorname{argmin}}(\lambda\mathcal{L}_A(\theta) + \mathcal{L}_B(\theta))$, where λ is a hyper-parameter indicating the importance of task A. The idea of EWC is that we can use another computational cheaper loss function to replace the original one, i.e.

$$\min_{\theta}\ \lambda\mathcal{L}_A(\theta) + \mathcal{L}_B(\theta)\ \equiv\ \min_{\theta}\ \mathcal{L}_B(\theta) + \frac{\lambda}{2}\sum_i F_i(\theta_i - \theta_{A,i}^*)^2, \tag{1}$$

where $\mathcal{L}_A$ and $\mathcal{L}_B$ are twice-differentiable convex loss functions for task A and task B, $\theta_A^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_A(\theta)$ and $F_i = (\frac{\partial \mathcal{L}(\theta_A^*)}{\partial \theta_{A,i}^*})^2$ under some assumptions.

Equality (1) is based on the 2-task scenario, but it can be easily converted into the multi-task scenario[1]. Since DeepMind introduced EWC, many online learning models focusing on different fields have appeared thanks to the power and generalization ability of EWC.

[1] proves equality (1) using statistical methods. Now we will try to use optimization methods to deal with this equality. However, this equality is not easy to prove from an optimization perspective. Here we will give an equivalent proof to a simpler version, which can be proved equivalent to (1) using statistical method. That is

$$\min_{\theta} \ \lambda \mathcal{L}_A(\theta) + \mathcal{L}_B(\theta) \ \equiv \ \min_{\theta} \ \mathcal{L}_B(\theta) + \frac{\lambda}{2} \sum_i F_i'(\theta_i - \theta_{A,i}^*)^2, \tag{2}$$

where $F_i' = \frac{\partial^2 \mathcal{L}(\theta_A^*)}{\partial (\theta_{A,i}^*)^2}$.

2 Proof

This section focuses on proving equality (1) and (2). Before starting the proof, we need to declare some assumptions to make the proof holds. According to the experiments conducted on 3, these assumptions hold for certain kinds of dataset.

Assumption 1 *New trained θ_{new} is close to θ_A^*, which indicates $||\theta_{new} - \theta_A^*|| \leq \epsilon$ for some small ϵ.*

Then the Second-order Taylor Series Expansion of $\mathcal{L}_A$ around θ_A^* will be

$$\mathcal{L}_A(\theta) \ = \ \mathcal{L}_A(\theta_A^*) \ + \ (\nabla \mathcal{L}_A)^T(\theta - \theta_A^*) \ + \ \frac{1}{2}(\theta - \theta_A^*)^T H_{\mathcal{L}_A}(\theta - \theta_A^*) \tag{3}$$

Assumption 2 *$\theta s'$ are independent from each other.*

Formally, $\frac{\partial^2 \mathcal{L}_A(\theta)}{\partial \theta_{i,j}^2} = 0$ if $i \neq j$. It indicates the Hessian matrix of $\mathcal{L}(\theta)$ is a diagonal matrix, i.e.

$$(H_{\mathcal{L}_A})_{i,j} = 0 \quad if \ i \neq j. \tag{4}$$

Let's simulate the procedure of online learning. Suppose we have a model with parameter θ_A^* which has already been optimized at the original training set (task A). So $\theta_A^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_A(\theta)$ is a known

[1] See 4.2 for detail.

parameter and also a local (global) optimal solution of $\mathcal{L}_A(\theta)$. According to First Order Necessary Condition,

$$\nabla \mathcal{L}_A(\theta_A^*) = 0. \tag{5}$$

If we have an additional training set (task B) and we want our model to be optimized on both task A and task B, then what we need to get is the newly optimized θ^* such that $\theta^* = \underset{\theta}{\operatorname{argmin}}(\lambda\mathcal{L}_A(\theta)+\mathcal{L}_B(\theta))$. The corresponding optimization problem is

$$\min_{\theta} \ \lambda\mathcal{L}_A(\theta) + \mathcal{L}_B(\theta). \tag{6}$$

Since we already have the optimized parameter θ_A^* on task A and we believe the newly trained parameter will be close to θ_A^* (Assumption 1), expand $\mathcal{L}_A$ around θ_A^* according to (3), then we will get (6) is equivalent to

$$\min_{\theta} \ \mathcal{L}_B(\theta) + \lambda(\mathcal{L}_A(\theta_A^*) \ + \ (\nabla\mathcal{L}_A)^T(\theta-\theta_A^*) \ + \ \frac{1}{2}(\theta-\theta_A^*)^T H_{\mathcal{L}_A}(\theta-\theta_A^*)). \tag{7}$$

Because $\mathcal{L}_A(\theta_A^*)$ is a constant, this term does not affect the optimization problem. Besides, we have (5). Hence, (7) is equivalent to

$$\min_{\theta} \ \mathcal{L}_B(\theta) + \frac{\lambda}{2}(\theta-\theta_A^*)^T H_{\mathcal{L}_A}(\theta-\theta_A^*). \tag{8}$$

Due to Assumption 2, only diagonal entries of $H_{\mathcal{L}_A}$ are non-zero. Furthermore, we have $(H_{\mathcal{L}_A})_{i,i} = \frac{\partial^2\mathcal{L}(\theta_A^*)}{\partial(\theta_{A,i}^*)^2}$. Finally, (8) is equivalent to

$$\min_{\theta} \ \mathcal{L}_B(\theta) + \frac{\lambda}{2}\sum_i \frac{\partial^2\mathcal{L}_A(\theta_A^*)}{\partial(\theta_{A,i}^*)^2}(\theta_i-\theta_{A,i}^*)^2, \tag{9}$$

which shows (2) holds.

If loss functions $\mathcal{L}$ is negative log-likelihood, using some statistical methods[2], we can get

$$E\left[\left(\frac{\partial\mathcal{L}(\theta)}{\partial\theta}\right)^2\middle|\theta\right] \ = \ E\left[\frac{\partial^2\mathcal{L}(\theta)}{\partial\theta^2}\middle|\theta\right]. \tag{10}$$

[2]See Lehmann & Casella, eq. (2.5.16), Lemma 5.3, p.116. Or https://en.wikipedia.org/wiki/Fisher_information#Definition. Minus sign is eliminated because here we consider its negative form.

However, (10) can not directly show $(\frac{\partial \mathcal{L}(\theta)}{\partial \theta})^2 = \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta^2}$. To continue the proof, we have to take it as an assumption.

Assumption 3 *The second order derivative of $\mathcal{L}(\theta)$ is roughly equal to the square of the first order derivative of $\mathcal{L}(\theta)$.*

$$(\frac{\partial \mathcal{L}(\theta)}{\partial \theta})^2 = \frac{\partial^2 \mathcal{L}(\theta)}{\partial \theta^2} \tag{11}$$

Then (9) is equivalent to

$$\min_{\theta} \ \mathcal{L}_B(\theta) + \frac{\lambda}{2} \sum_i (\frac{\partial \mathcal{L}_A(\theta^*_A)}{\partial \theta^*_{A,i}})^2 (\theta_i - \theta^*_{A,i})^2. \tag{12}$$

Overall, we have (6) is equivalent to (12), which shows (1) holds. □

3 Experiment

This section focuses on the fitness of data to assumptions.

3.1 Assumption 1

We use a linear regression model to fit randomly sampled data[3]. Different tasks are defined as using different data features to predict values. Moreover, in different tasks, we also change the θ_{Actual} in data generation to simulate the distribution shift.

What we find is that no matter how data changes, the error caused by Taylor expansion will always be dominated by the error caused by Assumption 2.

3.2 Assumption 2

We use a linear regression model to fit randomly sampled data[4]. According to Figure 1, the observation is that

$$\frac{\sum_i |(H_{\mathcal{L}})_{i,i}|}{\sum_i \sum_j |(H_{\mathcal{L}})_{i,j}|} \propto \log(\frac{|M|}{Dim_M}),$$

where $|M|$ is the size of sample data and Dim_M is the dimension of sample data[5]. The left-hand side ratio can be treated as approximate accuracy. That means if this ratio is high, our approximation in Assumption 2 will be more accurate.

[3]Notebook is available at https://github.com/Yasgant/pytorch-ewc/blob/master/taylor.ipynb
[4]Notebook is available at https://github.com/Yasgant/pytorch-ewc/blob/master/hessian.ipynb
[5]Another way to explain it is that Dim_M is the number of parameters

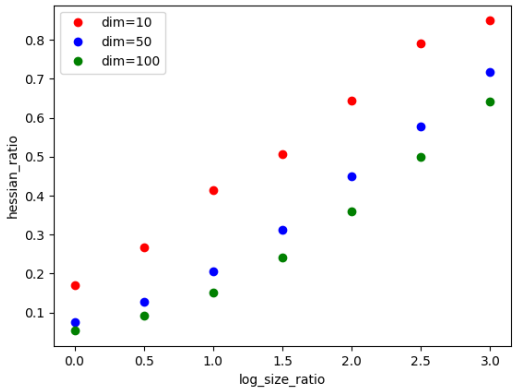

Figure 1: Hessian ratio

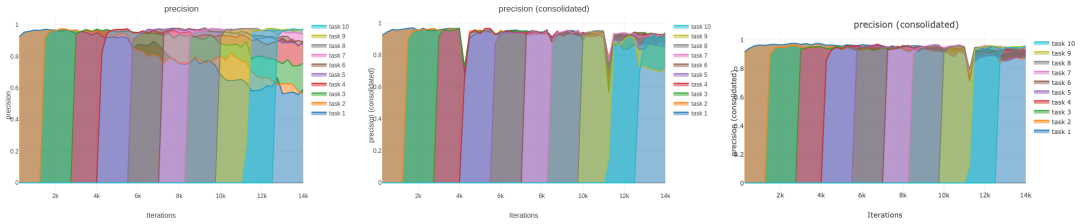

Figure 2: Result on MNIST dataset. No EWC(Left), Original EWC(Middle), New EWC(Right)

3.3 Assumption 3

Theoretically, Assumption 3 is always wrong because θ_A^* is a local optimal solution of $\mathcal{L}_A$ and $\frac{\partial \mathcal{L}_A(\theta_A^*)}{\partial \theta_A^*} = 0$ while $\frac{\partial^2 \mathcal{L}_A(\theta_A^*)}{\partial (\theta_A^*)^2} \neq 0$. But in practice, it is hard to get the optimal θ_A^*. $\frac{\partial \mathcal{L}_A(\theta_A^*)}{\partial \theta_A^*} \neq 0$ sometimes hold due to the limitation of practical optimization methods.

This does not mean what we have done makes no sense. Many experiments have shown that EWC works well in online learning. Our idea is to use some experiments to show F_i' and F_i have the same effect in real-world datasets to prove Assumption 3 indirectly.

We reuse other's EWC implementation on MNIST dataset but change the EWC term from $(\frac{\partial \mathcal{L}(\theta_A^*)}{\partial \theta_{A,i}^*})^2$ to $\frac{\partial^2 \mathcal{L}(\theta_A^*)}{\partial (\theta_{A,i}^*)^2}$.[6] Different tasks are defined as different permutations of image pixels (such as all pixels originally located at $(1,1)$ will be located at $(2,5)$ in a new task). According to the result in Figure 2, we find that $\frac{\partial^2 \mathcal{L}(\theta_A^*)}{\partial (\theta_{A,i}^*)^2}$ (right) performs better than $(\frac{\partial \mathcal{L}(\theta_A^*)}{\partial \theta_{A,i}^*})^2$ (middle). The explanation is that the original F is an estimation of F'. Dropping out this estimation can improve performance.

[6]Code is available at https://github.com/Yasgant/pytorch-ewc

4 Discussion

This section focuses on the capabilities and limitations of EWC on different models and datasets.

4.1 Assumptions

Due to Assumption 1, if there is a huge distribution shift on the new data (task), compared with using EWC, retraining the whole model will be a better choice because new trained θ_{new} will be far from θ^*_A, which cause the approximation accuracy of Taylor expansion become extremely low.

According to the result of 3.2, we know that the approximation accuracy of Assumption 2 depends on sampling data size and parameter number. It means EWC will work well in a low-parameter model with a huge training dataset.

From Figure 2, we conclude that by replacing F with F', the performance of EWC will improve. Although calculating F' takes more time than calculating F, this time is only spent in the preprocessing period. Besides, this additional preprocessing time can be simply omitted if we have large enough training data.

4.2 Online Learning

Recall that in online learning, we have a sequence of tasks. The original EWC can only handle the situation of two tasks. In order to deal with multi-task situations, we can modify EWC to fit multi-task situations.

Suppose we have three tasks A, B, C, and two optimized parameters θ^*_A, θ^*_B on task A, task AB respectively. If we want to train this model on task ABC, the optimization problem will be

$$\min_{\theta} \ \mathcal{L}_C(\theta) + \frac{\lambda_A}{2} \sum_i F_{A,i}(\theta_i - \theta^*_{A,i})^2 + \frac{\lambda_B}{2} \sum_i F_{B,i}(\theta_i - \theta^*_{B,i})^2.$$

In general, suppose we have tasks $T_1, T_2, ..., T_{n+1}$ and optimized parameters $\theta^*_{T_1}, \theta^*_{T_2}, \theta^*_{T_n}$. To train the model on task T_{n+1}, the optimization problem is

$$\min_{\theta} \ \mathcal{L}_{T_{n+1}}(\theta) + \sum_{j=1}^{n} \frac{\lambda_j}{2} \sum_i F_{T_j,i}(\theta_i - \theta^*_{T_j})^2, \tag{13}$$

where $F_{T_j,i} = (\frac{\partial \mathcal{L}_{T_j}(\theta^*_{T_j})}{\partial \theta^*_{T_j,i}})^2$.

However, when the number of tasks is large, we can not afford the cost of computing estimated loss separately. [2] shows that under some restrictions, (13) is equivalent to

$$\min_{\theta} \ \mathcal{L}_{T_{n+1}}(\theta) + \frac{1}{2}\sum_{i} F_i^*(\theta_i - \theta_{T_n}^*)^2, \tag{14}$$

where $F_i^* = \sum_{j=1}^{n}(\lambda_j F_{T_j,i})$, which is independent from θ and can be computed at the beginning of training task T_{n+1}.

References

[1] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[2] Ferenc Huszár. "On quadratic penalties in elastic weight consolidation". In: *arXiv preprint arXiv:1712.03847* (2017).