Continual NLU: Enabling language models to learn downstream tasks in an online fashion

Mohith Damarapati, Govind Mittal, Chandra Prakash Konkimalla
{md4289, gm2724, cpk290}@nyu.edu

Abstract

Qualified general intelligence has the ability of learning new tasks without immediately becoming worse at an older one. With this statement at the core of this paper, we explore various techniques to prevent catastrophic forgetting (CF) in a state-of-the-art language model. The paper explores two neuroscience inspired regularization techniques to mitigate CF, *viz.*, Elastic Weight Consolidation and Memory Aware Synapses on pre-trained BERT model. Backward transfer is the metric used to evaluate these techniques. The paper aims to reduce negative backward transfer on the previously trained tasks. As per our knowledge, this is the first work attempting to enable language models to learn downstream tasks in an online fashion and evaluating them based on the metric mentioned above.

1 Introduction

Natural cognitive systems gradually forget previously learned information unlike the CF observed in current connectionist approaches (French, 1999). This problem is significant in the domain of natural language understanding (NLU) as well.

Humans are remarkable at using their previously acquired knowledge when learning a new language task. For instance, children first learn a language by mostly observing and listening to their parents. They bolster their language skills through curricula by learning more formal language rules like grammar and syntax. Using their linguistic knowledge, they can comprehend intricate texts and can answer questions based on them. They can even infer meanings of natural language sentences used in complex contexts. In all stages of learning, humans are exceptionally good at retaining important parts of their previous knowledge.

In order to design real-world self-learning systems, two requirements are essential – adaptability to learn new knowledge and stability while evolving and integrating it. Therefore, a the ratio of *how plastic* and *how static* a system should be decides how adaptable and how stable it is, respectively. A balance needs to be maintained to alleviate the problem of CF of consolidated knowledge, which gives rise to the *stability-plasticity dilemma*.

State-of-the-art deep learning based language models like BERT (Devlin et al., 2018) perform exceptionally well on the NLU tasks. However, these models struggle to learn new tasks by retaining previously acquired knowledge (Yogatama et al., 2019). Whenever new data becomes available, the model has to be re-trained again on all the previous data. Further, it expects our data to be shuffled to perform well. Re-training is a costly process and shuffling is not possible in case of scenarios where data comes in a continuous stream.

Current literature evaluates language models based on metrics like accuracy and F1 score on a single task for which it is fine-tuned for. We call this as vanilla fine-tuning. Vanilla fine-tuning does not measure adaptability of our model to learn new tasks without forgetting the previously acquired knowledge. In this paper, we propose a continual learning framework to fine-tune our language models with a capability to adapt to learning new tasks. Apart from accuracy and F1 score, our framework also outputs backward transfer Lopez-Paz and Ranzato which is a metric to measure the retention capacity. We further argue that retaining and reusing the linguistic knowledge and avoiding CF is an important characteristic of general linguistic intelligence.

Our work in this paper mainly focuses on comparing vanilla fine-tuning with continual fine-tuning on BERT-base. We incorporate neuroscience inspired techniques like Elastic Weight Consolidation (EWC, Kirkpatrick et al. 2016) and Memory Aware Synapses (MAS, Aljundi et al. 2018) into our framework to mitigate CF.

2 Related Work

There has been multiple ways proposed to mitigate CF in connectionist networks. The paper segregates and reviews some of them into the following categories.

2.1 Memory based methods

Memory methods involve *rehearsing* on old samples along with seeing new distribution data alternately (Robins, 1995; Gepperth and Karaoguz, 2016). These methods work best in preventing CF, but the biggest drawback is that they involve storing past events explicitly, which makes it difficult to scale up with time as the neural networks becomes very large and neural resources are limited. Also, this approach does not consider the fact that a network needs to learn unknown number of future tasks, which makes choosing the *defining set of examples* per task for storing in a fixed resourced system complicated.

2.2 Regularization based methods

These methods alleviates CF by enforcing a cost-based constraint during the update step of the network weights. These approaches aim to change a subset or all of the current parameters to an optimal value that works well for both the previous tasks and the current new task.

Kirkpatrick et al. (2016) proposed **Elastic Weight Consolidation (EWC)** which constrains each parameter based on how important it is to the previous task, which results in model that is flexible for adapting to new tasks while maintaining the performance on previous task. Per-parameter significance is calculated using the diagonal of the Fischer information matrix, which is an approximation of the second derivative of loss with respect to each parameter. This approximation is linear to compute in the number of parameters and training examples. The final parameters are calculated as follows:

$$\theta_N^* \leftarrow \text{argmin}_\theta L_N(\theta) + \lambda \sum_i \frac{1}{2}\left(\frac{dL_{Pi}(\theta_{Pi})}{d\theta_{Pi}}\right)^2 (\theta - \theta_{Pi})^2 \quad (1)$$

where θ_N^* are the optimal parameters after training on new task. $L_N(\theta)$ is the loss metric for new task, $dL_{Pi}(\theta_{Pi})$ is the loss metric for previous (i^{th}) task, θ_{Pi} are the optimal parameters until i^{th} task. λ is a hyper-parameter for measuring the importance of remembering old task parameters.

Aljundi et al. (2018) proposed **Memory Aware Synapses** inspired by Hebbian learning in natural systems. The core idea is to learn which parts of the model parameters are important relative to others, using unlabelled data itself. They calculate importance weights Ω_{ij} for all parameters θ_{ij}, where i and j are indices of pairs of neurons n_i and n_j residing in two consecutive layers, respectively. The importance weight represent the *sensitivity* of the function output of a learned function, $F : X_n \longrightarrow Y_m$, to changes in the corresponding parameter. Therefore,

$$\Omega_{ij} = \frac{1}{N}\sum_{k=1}^{N} \| \, g_{ij}(x_k) \, \| \quad (2)$$

where, N is the number of data points at a given stage and

$$g_{ij}(x_k) = \frac{\partial(F(x_k;\theta))}{\partial\theta_{ij}} \quad (3)$$

for a given data point x_k. Therefore, whichever parameters have lower importance can be changed without degrading the accuracy on previous tasks, while decreasing the loss for subsequent tasks. Thus, if we wish to learn a new task, the total loss function $L(\theta)$ becomes,

$$L(\theta) = L_n(\theta) + \lambda \sum_{i,j} \Omega_{ij}(\theta_{ij} - \theta_{ij}^*)^2 \quad (4)$$

where, $L_n(\theta)$ is the new task loss, θ_{ij}^* are the old parameters and λ is a regularizer hyperparameter. Zenke et al. (2017) **Synaptic Intelligence** loss is similar to MAS:

$$L(\theta) = L_n(\theta) + \lambda \sum_k \Omega_k^\mu (\theta - \theta_k)^2,$$

$$\Omega_k^\mu = \sum_{v<\mu} \frac{\omega_k^v}{(\Delta_k^v)^2 + \epsilon}$$

$$\omega_k^v = \int_{t^{\mu-1}}^{t^\mu} g_k(t)\theta_k^{'}(t)$$

Here k signifies task number, $g_k(t)$ is the gradient of loss w.r.t parameter k, $\theta_k^{'}(t)$ is the parameter change at time t. In this case, the importance of parameter is an accumulation of product of gradient and parameter change. It depends on two quantities: 1) ω_k^v is a drop in loss over the trajectory of training. 2) How far the loss has changed $\Delta_k^v = \theta_k(t^v) - \theta_k(t^{v-1})$.

2.3 Dynamic Architectures

Dynamic approaches the problem by augmenting a network as new tasks are encountered and more neural resources are needed (Rusu et al., 2016; Cortes et al., 2017). Old network parameters are fully preserved to prevent CF and additional neurons are added and trained separately. Although this approach works well for reinforcement learning tasks but architecture complexity increases with number of learned tasks.

3 Proposed Method

3.1 Evaluation Metrics

Our continual learning framework is evaluated by the following two metrics. They are:

1. *Backward transfer (BT)* is the influence of learning a new task on the previous tasks.

2. *Accuracy* is the test classification accuracy on the current new task.

We define R as the accuracy matrix and Tr as the transfer matrix, where $R_{i,j}$ is the test classification accuracy of the model on task t_j after the previous task t_i and $Tr_{i,j}$ is the backward transfer when $i > j$, respectively.

3.2 Continual Framework [1]

Algorithm 1: Proposed Framework

```
$T \leftarrow$ Tasks
$N \leftarrow$ Number of Tasks
$\theta_{B_0} \leftarrow$ Bert Pre-trained Parameters
$R \leftarrow 0_{N\times N}$ ;Test Accuracy Matrix
$Tr \leftarrow 0_{N\times N}$ ;Transfer Matrix
for $i = 1 : N$ do
    $D_i \leftarrow data(T_{i,train})$
    $\theta_{B_i}, \theta'_{T_i}$
      $\leftarrow \text{argmin}_{\theta_B,\theta_T} L(D_i, \theta_{B_{i-1}}, \theta_{T_i})$
    for $j = 1 : N$ do
        $D_{test} \leftarrow data(T_{j,test})$
        $R_{i,j} \leftarrow acc(\theta_{Bi}, \theta'_{T_j}, D_{test})$
        $Tr_{i,j} \leftarrow R_{i,j} - R_{i,i}$
return $R, Tr$
```

The paper proposes a continual framework for learning multiple downstream tasks on BERT and illustrates in Algorithm 1. Let T be the tasks to be learned, N be the number of tasks, θ_{B_0} be BERT pre-trained parameters obtained from huggingface[2], R be the accuracy matrix and Tr be the transfer matrix, both initialized to zero.

Continual fine-tuning procedure involves common BERT parameters and multiple heads. First, T_1 is trained with BERT parameters(θ_{B_0}) and a randomly initialized task-specific head (θ_{T_1}) with corresponding loss ($L(D_1, \theta_{B_0}, \theta_{T_1})$). Next task T_2 is trained with previous BERT parameters(θ_{B_1}) and another randomly initialized task-specific head(θ_{T_2}) with loss($L(D_2, \theta_{B_1}, \theta_{T_2})$). This procedure is repeated for all the remaining tasks. Accuracy($R_{i,j}$) is calculated by using θ_{B_i} and head θ'_{T_i} with test data ($data(T_{j,test})$), where θ'_{T_i} are the final task-specific head parameters.

In case of vanilla fine-tuning, all task-specific heads are initialized randomly and BERT-specific parameters (All BERT parameters except the task specific ones) are re-initialized to θ_{B_0}. But, in this framework, BERT-specific parameters are *not re-initialized*, when training new tasks.

In EWC, we compute mean and Fisher consolidate after training each task. Fisher consolidate is computed as a running average instead of accumulating all the gradients by approximating (ref. eq. 1) as given in Rao 1992. This approximation minimizes the memory complexity of our gradient updates which is important for a feasible implementation of a model, like BERT consisting of ~100M parameters.

In MAS, we use a modified Adam optimizer to access the gradients of each batch (ref. eq. 3) and update the corresponding importance weights (ref. eq. 2) as running average of the absolute values of per-parameter gradients for a given batch. The gradient of each parameter for the current task is added with $2\lambda\Omega(\theta - \theta^*)$ (ref. eq. 4).

After computing importance weights of each parameter for the current task and finishing training, we consolidate the previously trained weights θ_{B_i}. These consolidated previous parameters helps us to retain knowledge obtained from previously learned tasks and are used while learning a new one.

While evaluating the model, test accuracy on task T_i is reported after re-installing its task-specific head. After training a task T_i, backward transfer for all previous tasks ($T_j, \forall j < i$) is evaluated. Backward transfers are obtained from R and stored in the lower triangle of an $N \times N$ transfer matrix Tr. These consolidated previous parameters helps us to retain knowledge obtained from previously learned tasks and are used while learning a new one.

3.3 Data and Base Model

All our experiments are performed on GLUE tasks (Wang et al., 2019). We selected two low resource (MRPC and RTE) and two high resource (QNLI and SST-2) tasks. Results are reported on multiple

[1]Code available at - `https://github.com/mittalgovind/continual-learning-nlu`.

[2]https://github.com/huggingface/transformers

combinations of two and three tasks.

All our experiments are performed on BERT [3], which can also be extended to more recent language models like RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019), etc.

4 Results and Analysis

Table 1: Low Resource Tasks. Representation: 0.87 (-0.48)→0.64 meant the accuracy was 0.87 after the model is trained on 1st task. It reduced by 0.48 after training on 2nd task and the 2nd task accuracy is 0.64.

Task	MRPC→RTE	RTE→MRPC
Base	0.87 (-0.48)→0.64	0.65 (-0.14)→0.83
EWC	0.86 (-0.00)→0.65	0.65 (-0.09)→0.86
MAS	0.87 (-0.09)→0.71	0.71 (-0.16)→0.87

Table 2: High Resource Tasks: SST-2 and QNLI

Task	SST-2→QNLI	QNLI→SST-2
Base	0.91 (-0.39)→0.77	0.86 (-0.38)→0.90
EWC	0.91 (-0.08)→0.79	0.87 (-0.01)→0.88
MAS	0.90 (-0.41)→0.50	0.87 (-0.30)→0.90

Table 3: Three Tasks: MRPC→RTE→SST-2. *Future transfer (FT) is not considered.

Task	MRPC	RTE	SST-2
Baseline	0.87	*FT	*FT
	0.46 (-0.41)	0.71	*FT
	0.43 (-0.44)	0.5 (-0.21)	0.91
EWC	0.88	*FT	*FT
	0.83 (-0.05)	0.68	*FT
	0.85 (-0.03)	0.6 (-0.08)	0.91
MAS	0.87	*FT	*FT
	0.64 (-0.33)	0.65	*FT
	0.67 (-0.30)	0.52 (-0.23)	0.92

We performed hyperparameter optimization on all results for EWC by running for each task with different lambdas - {5e5, 1e6, 1.5e5, 2e6} and for MAS the lambdas are {150, 200, 250}. The reported results for each task are the best among their lambda and seed settings.

The results for the baseline (BERT-base), EWC and MAS for the low resource strengthened our hypothesis that the models suffer from very high negative backward transfer. In Table 1, EWC achieves zero backward transfer for the MRPC→RTE experiment. MAS has higher accuracy for RTE but has a lower negative backward transfer compared to EWC. In case of data-intensive tasks (QNLI/SST-2) we can see that the baseline backward transfer is low but the EWC performs significantly better. MAS does not perform well on data-intensive tasks across all tested seeds and lambdas.

[3]We argue that CF is a problem not specific to language models but to neural networks as a whole. In this work we wish to prove that using regularization techniques can alleviate CF. Therefore, it justifies using BERT, as the negative backward transfer might remain proportional and serves as a proof-of-concept.

Table 4: Three Tasks: MRPC→SST-2→RTE.

Task	MRPC	SST-2	RTE
Baseline	0.87	*FT	*FT
	0.41 (-0.46)	0.91	*FT
	0.64 (-0.23)	0.74 (-0.17)	0.52
EWC	0.85	*FT	*FT
	0.82 (-0.03)	0.91	*FT
	0.85 (-0.00)	0.9 (-0.01)	0.63
MAS	0.86	*FT	*FT
	0.4 (-0.46)	0.91	*FT
	0.75 (-0.11)	0.84 (-0.07)	0.61

In case of three consecutive tasks (MRPC→RTE→SST-2), the baseline reduces to random choice accuracy for MRPC and RTE tasks. Therefore, the baseline model has catastrophically forgotten the previous tasks and making a random choice. For MRPC→SST-2→RTE the baseline has better accuracy. In both of the three task cases, EWC keeps the negative backward transfer in check.

5 Conclusion

We present a proof-of-concept that neural networks, like BERT, suffer from catastrophic forgetting. In order to amend CF, current models expect shuffled training data and re-training procedures for all the previous tasks while learning a new task. Re-training is very costly while shuffling is not possible in the cases when data comes continuously. Further, models suffering from it cannot be safely deployed in real-time systems to learn new tasks without the intervention of a deep learning practitioner. We proposed a continual learning framework which could replace vanilla fine-tuning to enable language models to learn new tasks in an online fashion. While EWC greatly mitigated CF in all cases, but MAS fails to achieve the same benchmark, specially for high resource tasks.

6 Future Work

Backward transfer was the metric we used to evaluate our models, but in order to achieve general linguistic intelligence high future transfer is also expected. The solution we proposed can be combined by using few-shot learning techniques like Matching Networks, Model Agnostic Machine Learning (MAML), Prototypical Networks to build more robust language models.

We also worked (and implemented) on Synaptic Intelligence (SI, Zenke et al. 2017) but it suffers from gradient explosion problem and therefore has been pushed to future work.

Availability

The code and LaTeX source files are available at `https://github.com/mittalgovind/continual-learning-nlu`.

Collaboration Statement

The contributions of the authors, in the same order, are as follows:

Mohith Damarapati: Implemented EWC (Elastic Weight Consolidation) from scratch. Designed a framework to evaluate models in a continual way. Fixed some crucial bugs that hindered our project progress. Ran all the experiments for Baseline.

Govind Mittal: Implemented Memory Aware Synapses (MAS) and Synaptic Intelligence (SI) for BERT from scratch. Wrote the complete related work (Sec. 2, part of introduction (Sec. 1) and paper reviewing.

Chandra Prakash Konkimalla: Implemented the baseline framework for fine-tuning the tasks one after another. Fixed bugs in EWC (Elastic Weight Consolidation) training procedure. Integrated Baseline and EWC into one code so that the results can be consistent. Ran all the experiments for EWC and MAS.

References

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

Corinna Cortes, Xavi Gonzalvo, Vitaly Kuznetsov, Mehryar Mohri, and Scott Yang. 2017. Adanet: Adaptive structural learning of artificial neural networks. *ArXiv*, abs/1607.01097.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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

Alexander Gepperth and Cem Karaoguz. 2016. A bio-inspired incremental learning architecture for applied perceptual problems. *Cognitive Computation*, 8:924–934.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2016. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continuum learning. *CoRR*, abs/1706.08840.

C. Radhakrishna Rao. 1992. *Information and the Accuracy Attainable in the Estimation of Statistical Parameters*, pages 235–247. Springer New York, New York, NY.

Anthony V. Robins. 1995. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connect. Sci.*, 7:123–146.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. 2016. Progressive neural networks. *ArXiv*, abs/1606.04671.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. 2019. Learning and evaluating general linguistic intelligence. *CoRR*, abs/1901.11373.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Improved multitask learning through synaptic intelligence. *CoRR*, abs/1703.04200.