Parameter-efficient Continual Learning Framework in Industrial Real-time Text Classification System

Tao Zhu, Zhe Zhao [*], Weijie Liu, Jiachi Liu, Yiren Chen, Weiquan Mao, Haoyan Liu,
Kunbo Ding, Yudong Li, and Xuefeng Yang
Tencent Research, Beijing, China
{mardozhu, nlpzhezhao, jagerliu, jiachiliu, yirenchen, weiquanmao, haoyanliu, karlding, yudongli, ryanxfyang}@tencent.com

Abstract

Catastrophic forgetting is a challenge for model deployment in industrial real-time systems, which requires the model to quickly master a new task without forgetting the old one. Continual learning aims to solve this problem; however, it usually updates all the model parameters, resulting in extensive training times and the inability to deploy quickly. To address this challenge, we propose a parameter-efficient continual learning framework, in which efficient parameters are selected through an offline parameter selection strategy and then trained using an online regularization method. In our framework, only a few parameters need to be updated, which not only alleviates catastrophic forgetting, but also allows the model to be saved with the changed parameters instead of all parameters. Extensive experiments are conducted to examine the effectiveness of our proposal. We believe this paper will provide useful insights and experiences on developing deep learning-based online real-time systems.

1 Introduction

In industry, many text-related applications have enjoyed a superior performance boost from the emerging of pre-trained language models, such as word2vec (Mikolov et al., 2013a,b; Zhao et al., 2017) , ELMo (Peters et al., 2018), GPT (Radford et al., 2018, 2019), and BERT (Devlin et al., 2019; Liu et al., 2019; Sun et al., 2019). However, when a fine-tuned model needs to be updated to master a new task swiftly, it usually loses the ability to handle previous tasks. This phenomenon is known as catastrophic forgetting (French, 1999), and it poses a significant issue in industrial settings.

Continual learning aims to incrementally expand acquired knowledge for future learning (Chen and Liu, 2018), and mitigate the impact of catastrophic forgetting in the meantime. Existing continual learning methods usually use data replay (Rebuffi et al., 2017b), parameter isolation (Rusu et al., 2016; Fernando et al., 2017), and regularization (Kirkpatrick et al., 2017; Li and Hoiem, 2017) to make models adapt to new tasks without catastrophic forgetting. However, these approaches lack research on implementing continual learning in industrial scenarios, where endowing models with continual learning capabilities meets numerous practical constraints. For time constraints, when new data or tasks arrive, the model should be launched in minutes or even seconds, which is common in time-sensitive scenarios, e.g., blocking certain rumors content. For space constraints, the strong demand for tracing tasks makes it necessary to save every model once it is changed. So for the current large-scale pre-trained models, storage becomes an industrial challenge with the increase of new tasks.

To solve these industrial challenges, we propose a parameter-efficient continual learning framework based on an offline parameter selection strategy. The framework consists of two parts, i.e., offline calculation and online training. In the offline calculation part, all the parameters that are important to the old task are selected to be fixed, while the remaining parameters are employed to learn the new task. Since in a real industrial scenario, the arrival of a new task will have an interval of hours or even days, we can make full use of this interval to advance the selection of parameters. During the online training phase, the model is parameter-efficiently trained on a new task within a small set of parameters and further combines multiple regularization-based methods (Kirkpatrick et al., 2017; Li and Hoiem, 2017) to overcome catastrophic forgetting. To alleviate storage costs, we only save the modified parameters for each snapshot. Extensive experiments demonstrate that our framework can maintain the old task performance while learning a new task quickly. Our implementa-

[*]Corresponding author.

tion is based on UER-py pre-training toolkit[1] (Zhao et al., 2019).

The main contributions of this paper can be summarized as follows:

- We are the first to explore continual learning with only a few model parameters, and show that updating 0.1% parameters of BERT can achieve competitive performance.
- We propose a parameter-efficient continual learning framework that solves issues in real-world industrial settings by utilizing parameter-efficient-based offline parameter selection strategies and regularization-based online training methods.
- Extensive experiments on a real-world domain incremental text classification task verify the effectiveness of our proposed framework.

2 Related Work

2.1 Continual Learning

The major challenge of continual learning is catastrophic forgetting (McCloskey and Cohen, 1989; French, 1999), which occurs when optimizing for a new task causes performance degradation on a task learned previously. Methods designed to mitigate catastrophic forgetting mainly fall into three categories: replay methods, parameter isolation methods, and regularization-based methods (Delange et al., 2021).

Replay methods explicitly retrain on a subset of stored old task samples while training on new tasks. Instead of selecting samples at random, Rebuffi et al. (2017b) incorporated the Herding technique (Welling, 2009) to choose samples that best approximate the mean feature vector of a class, and it is widely used in Castro et al. (2018), Wu et al. (2019), Hou et al. (2019), Zhao et al. (2020), Mi et al. (2020a,b). Ramalho and Garnelo (2019) proposed to store samples that the model is least confident. However, replay methods exploit samples from old tasks, which will slow down the online training. To meet the time constraint, they are not used in our framework.

Parameter isolation methods dedicate different model parameters to each task, which are divided in two directions. One is growing a new branch network for a new task, while freezing previous task parameters (Rusu et al., 2016; Xu and Zhu, 2018). The other one is masking out parameters of previous task during new task training, which is imposed either at parameters level (Fernando et al., 2017; Mallya and Lazebnik, 2018), or unit level (Serra et al., 2018). Parameter isolation is unsuitable for usage in industrial scenario. It is difficult to keep track of the model's scale if the number of used parameters is continually accumulated as the number of tasks increases.

Regularization-based methods add an additional regularization term in the loss function, which will consolidate previous knowledge when learning on new data (Delange et al., 2021). Elastic weight consolidation (EWC) (Kirkpatrick et al., 2017) is a well-known regularization-based method, which introduces network parameter uncertainty in the Bayesian framework. LwF (Li and Hoiem, 2017) is another regularization method, using the previous model to infer current data and taking the outputs as soft labels to mitigate forgetting and transfer knowledge.

2.2 Parameter-efficient Training

Training a model with a few parameters is useful in many applications. Not only does the model have the potential to achieve better performance, but also disk space can be saved by only saving the updated parameters for each task. Recent work has shown that it is possible to update only a small subset of the model's parameters during training. This kind of work could heavily alleviate storage and deployment communication requirements . For example, Adapters (Houlsby et al., 2019; Rebuffi et al., 2017a; Bapna et al., 2019) introduce additional trainable parameters into a pre-trained model in the form of small task-specific modules while the rest of the model's parameters are kept fixed. Many works like Diff Pruning (Guo et al., 2020) and BitFit (Ben Zaken et al., 2021) have shown that it is possible to update only a small subset of the model's parameters during training, which can alleviate storage and communication requirements. Xu et al. (2021) and Sung et al. (2021) even show a acceptable performance on random selection of parameters. Therefore, we choose to perform parameter-efficient training on continual learning to quickly master a new task while avoid catastrophic forgetting.

3 Methodology

We introduce a parameter-efficient continual learning framework, as shown in figure 1. Our frame-

[1]https://github.com/dbiir/UER-py/

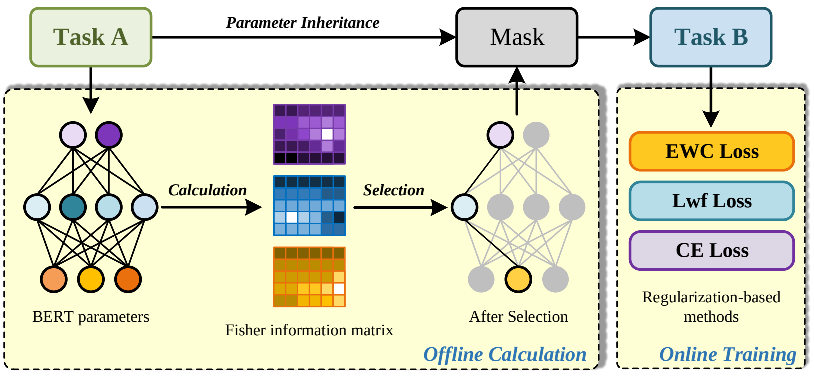

Figure 1: The overall architecture of the parameter-efficient continual learning framework.

work is divided into two components. The first is offline computation, which makes use of the interval between tasks to evaluate the data and select the parameters that are crucial to old tasks. These parameters are kept fixed in the new task. The other part is online training. In this stage, we utilize parameter-efficient training to perform new task training on the parameters that are not fixed. In addition, we introduce some well-known regularization-based methods in our framework for further improvement.

3.1 Offline calculation

The goal of offline calculation is to select the subset of parameters that are (in some sense) the most important to all of the old tasks, and fix them. Therefore, we make full use of the interval between tasks to review the previous training data, and calculate the parameters that are important to the previous tasks in the latest model (snapshot). These parameters will be fixed in the new task, and the remaining parameters will participate in the training.

As for the method of measuring the importance of parameters, we consider the indicator of how much changing the parameter will impact the model's output. The Fisher information is particularly well suited to identifying the highly relevant subset of parameters for previous tasks. It serves as an useful tool for estimating how much information a random variable contains about a parameter of the distribution (Tu et al., 2016). The Fisher information assumes that the more important the parameter towards the target task, the higher value it conveys. Formally, the Fisher information for the parameter θ_i is as follows:

$$F(\theta_i) = \frac{1}{|D|}\sum_{j=1}^{|D|}\left(\frac{a\partial \log p\left(y_j|x_j;\theta\right)}{\partial \theta_i}\right)^2 \quad (1)$$

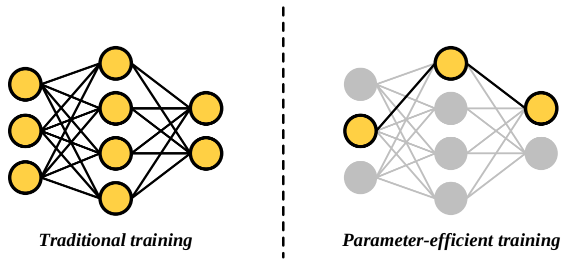

Figure 2: Illustration of parameter-efficient training.

where D denotes the task-specific training data, x and y denote the input and the output respectively.

3.2 Parameter-efficient Online training

Given the important parameters selected in offline stage, we use the remaining parameters for online training. Updating subset of the parameters can avoid catastrophic forgetting to some extent and largely decrease the storage space required by the snapshot. In addition, we combine the parameter-efficient training with two typical regularization-based continual learning methods. The combination of multiple orthogonal techniques can further improve the performance of our system. The overview optimization objective is as follows:

$$L(\theta^*) = L_{CE}(\theta^*) + L_{regul-based}(\theta^*), \theta^* \in S_\theta \quad (2)$$

where S_θ are parameters selected from the offline calculation stage, which have small Fisher values. L_{CE} denotes the cross-entropy loss, and $L_{regul-based}$ denotes regularization-based method loss.

3.2.1 Parameter-efficient learning

As shown in figure 2, in traditional training setting (left), all of the model's parameters are updated. But in our online training (right), we only train a few parameters, and most of the parameters are fixed according to the result of offline calculation to avoid forgetting old tasks. In general, the model will be easy to forget old tasks while learning new tasks, if more parameters are updated. Therefore, in our framework, we choose to perform parameter-efficient training on parameters that are not important to previous tasks, which are not fixed. Refer to previous experience (Ben Zaken et al., 2021; Xu et al., 2021), we chose a layer-wise strategy to select important parameters. We fixed the parameters from large to small (Fisher information) in a certain proportion at each layer.

On the other hand, updating a small number of parameters is beneficial for storage purpose and rapid deployment. Sometimes we need to deploy our model on thousands of servers. So the model size needs to be as small as possible (around 1.2 GB for BERT-Large 400 MB for BERT-Base). Our framework only needs to store the values and indices denoting the position of the updated parameters. Our experimental results demonstrate that only 0.1% trainable parameters of the original model can achieve competitive performance.

3.2.2 Regularization-based method

EWC and LwF are two representative approaches for preventing catastrophic forgetting in neural networks. They respectively add restrictions on model parameters and output activation. The two methods are orthogonal and we combine them as follows:

$$L_{regul-based}(\theta) = \lambda_1 L_{EWC}(\theta) + \lambda_2 L_{LwF}(\theta) \tag{3}$$

Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2017) introduces network parameter uncertainty in the Bayesian framework. Intuitively, this approach consists of a $L2$ penalty on the difference between the parameters for the old θ_i^* (i denotes the indexes of the parameters) and the new θ_i. It uses the diagonal of the Fisher information matrix F_i (2) to weight different parameters. The EWC loss (4) slows down the learning process of task-relevant parameters, which contains knowledge learned previously.

$$L_{EWC}(\theta) = \sum_i F_i(\theta_i - \theta_i^*)^2 \tag{4}$$

From formula (4), we can see that if most of the parameters are fixed, it is equivalent to reducing the EWC loss, which is beneficial to preventing catastrophic forgetting.

Learning without forgetting (LwF) (Li and Hoiem, 2017) is another method for continual learning. Before training the new task, network outputs for the new task data are recorded, which is denoted by y'_o. It will be subsequently used during training to distill prior task knowledge. LwF employs a variant of knowledge distillation. In our framework, we use $L2$ (5) loss to regulate the outputs:

$$L_{LwF}(\theta) = \sum_k (y_o - y'_o)^2 \tag{5}$$

4 Experiment

In this section, we empirically verify the effectiveness and efficiency of our framework under the setting of incremental text classification tasks in the industrial scenarios.

4.1 Dataset & Implementation Details

In the experiments, we utilize the Amazon Reviews dataset (He and McAuley, 2016) to examine our method, which is widely used in text classification tasks. The original dataset contains 142.8 million product reviews collecting from 29 different domains. To be consistent with our industrial scenario, we firstly create a reduced dataset by randomly selecting 10000 pieces of data from 12 domains as the base task, 6000 of which are used as training set, 2000 as validation set, 2000 as test set. In the incremental learning session, we construct 17 subsequent tasks with 300 examples for training, 100 for validation and 100 for test from the rest domains.

We adopt the BERT-Base model (Devlin et al., 2019) and the BERT default uncased vocabulary. All runs use the AdamW optimizer[2] (Kingma and Ba, 2014; Devlin et al., 2019) with 5 epochs, 32 batch size and 0.1 dropout rate. For the base task, we set learning rate as 2e-5. For the incremental tasks, we set learning rate as 2e-3, λ_1 as 0.35 and λ_2 as 1. Based on offline calculation results, we only pick up 0.1% parameters to participate in each training process. Our experiments are conducted in Intel(R) Xeon(R) CPU E5-2699 v4 at 2.20GHz, 2 Nvidia Tesla P40 GPU with 24 GB of RAM.

[2]https://www.fast.ai/2018/07/02/adam-weight-decay/

Order ↓ ~ Method →	EWC	LwF	EWC&LwF	PE-rand	PE-*	Lower Bound	Upper Bound
I	47.39	47.6	45.76	**50.6**	50.46	45.14	51.59
II	48.54	47.95	48.3	51.35	**51.63**	45.81	53.18
III	40.44	45.23	42.11	45.42	**49.51**	38.29	51.1
IV	43.27	45.16	41.04	42.07	**46.8**	41.27	54.78
Average Acc	44.91	46.49	43.30	47.36	**49.6**	42.63	52.66

Table 1: Main result of text classification (above) averaged accuracy score respectively (see Appendix A for the dataset orderings).

4.2 Models

We compare our proposed models with the a series of baseline methods in our experiments:

- lower-bound: a standard classification model is fine-tuned on the individual task without any continual learning strategy, which can be considered as the lower-bound method.
- upper-bound: a model is trained on all tasks simultaneously, which can be considered as the upper-bound method since it has access to the whole dataset.
- EWC & LwF: Two classical regularization-based methods for continual learning.
- PE-rand: Our proposal model is trained by randomly choosing some parameters and keeping them unchanged during online training stage, instead of using the offline calculation strategy.
- PE-*: Our continual learning framework, including offline calculation and parameter-efficient online training.

4.3 Results

The models are trained on the current training set and evaluated on the union of all the test sets. To ensure the robustness of the task ordering, we evaluate our methods on the four different orderings (chosen randomly), which are shown in Appendix A.

Table 1 provides a summary of our main results. We report the micro-averaged accuracy for the classification task. The lower bound is trained in the current task without using any continual learning strategy to overcome catastrophic forgetting, while the upper bound is trained on all data after the new task comes, which can be considered multi-task method. There is a significant gap between the lower bound and the upper bound, which illustrates the need for continual learning. As the classical CL methods, EWC and LwF outperform the standard model without any specific continual learning, but still suffer from catastrophic forgetting in the order IV. It can be seen that our proposed PE-* achieves a better performance than EWC, LwF and their combination. This is because most of the parameters in BERT are fixed, which is equivalent to posing a strict regularization to the parameters to prevent catastrophic forgetting while using the remaining parameters to learn new tasks. Compared to PE-rand, PE-* has a better average accuracy, which verifies the importance of parameter selections. Although the random selection method outperforms PE-* in order I, it is difficult to obtain a suitable set of parameters in most cases for models to learn new tasks while maintaining previous knowledge.

Moreover, according to the principles of EWC and LwF, the former records the initial model parameters, and the latter records the data features of new tasks. As the training progresses, their regular loss terms especially $L_{EWC}(\theta)$ will get bigger and bigger in model like BERT-Base with 110M parameters. What's more, in real industrial scenarios, each new task may have different suitable hyper parameters. We do not have time to do grid search of the best hyper parameters, so λ_{EWC} and λ_{LwF} may not be optimal solutions. This results in an unbalanced ratio of $L(\theta)$ to $L_{EWC}(\theta)$ and $L_{LwF}(\theta)$, where $L(\theta)$ may much smaller than $L_{EWC}(\theta)$ and $L_{LwF}(\theta)$. Therefore, as the number of tasks increases, the training of new tasks will become more and more difficult with the same set of hyper-parameters. However, the previous tasks have not been fully learned. This problem accumulates gradually in regularization-based method and leads to results that are not as good as our method (PE-*) which just handles a very small amount parameters.

Figure 3 shows the accuracy of model on the first task test set as the model are trained on more tasks. The figure illustrates how well each model retains its previously acquired knowledge as it learns new knowledge. We can see that our framework is con-

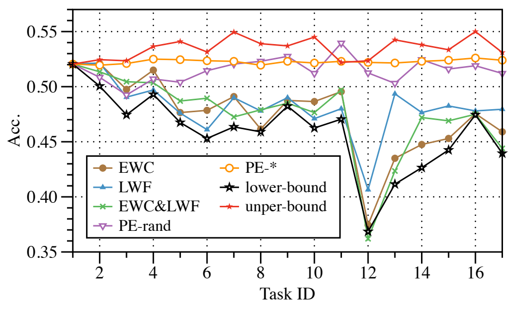

Figure 3: Performance on the first task test examples of order I during training as training progresses.

Reserved Percent	Order1	Order2	Order3	Order4
0.50%	**50.46**	**51.63**	**49.51**	46.8
1.00%	50.04	51.63	49.05	**47.34**
5.00%	45.11	48.61	46.44	44.43
10.00%	39.08	38.36	34.38	37.61

Table 2: The effect of layer-wise parameter reserved percentage on accuracy.

sistently better and more stable compared to other methods.

4.4 Parameter-efficient Strategy

Figure 4 shows the weight map of Fisher information. It can be seen that in the BERT model, the parameters of the embedding-layer have little effect on our classification task. Most of the important parameters are concentrated in the transformer block layer, and the importance of the attention layer is higher than that of the feed forward layer. In addition, according to our statistics, we found that 13%(about 14M) of the parameters' Fisher information is 0, and most of them are in embedding-layer.

In our experiments, we found that the parameters to be fixed cannot be determined simply in order of magnitude. According to (Jawahar et al., 2019), BERT encodes rich linguistic information in different transformer blocks. Therefore, refer to previous experience (Ben Zaken et al., 2021; Xu et al., 2021), we chose a layer-wise strategy to select important parameters. We fixed the parameters from large to small in a certain proportion at each layer of the model. To this end, each layer has parameters for new tasks to learn.

4.5 Model Size

We set layer-wise parameter reserved for new task to different values, 0.5%, 1%, 5% and 10%, and the percent of parameters fixed for old task are 99.5%, 99%, 95% and 90%. The advantage of our parameter-efficient continual learning becomes more pronounced at extreme sparsity rates. In Table 2, we report the accuracy across different task orders and reserved rates. We can observe that the more parameters fixed, the better the effect on alleviating catastrophic forgetting.

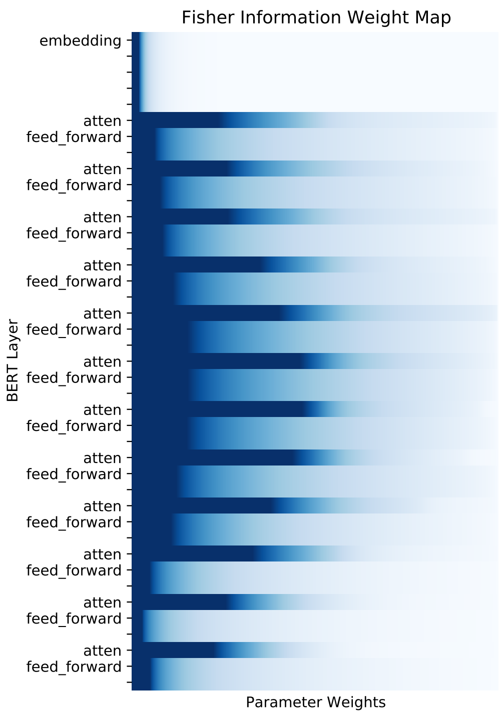

Figure 4: Calculating the model Fisher information by review old tasks, where dark colors are weighted more than light colors. The layers of BERT are ordered from bottom to top (i.e. the embedding layer is shown at the top).

Method	Acc	Saved Parameters	Saved model size
EWC	47.49	110B	421MB
LwF	47.95	110B	421MB
EWC&LwF	48.65	110B	421MB
PE-*	**51.63**	0.1B	1.2MB

Table 3: Comparison of storage costs.

In the above experiments, we only trained 0.1% of the parameters in the BERT model. We use the sparse-matrix method to store the model, and only store the index and value each time, occupying about 1.2 Mb of space, which is 0.3% of the entire model, as shown in Table 3. Parameter-efficient strategy greatly saves network bandwidth and storage requirements.

5 Conclusion

This paper introduces a parameter-efficient continual learning framework, which is designed for real-time incremental learning system. In offline stage, the framework identifies the parameters that are less important to the old tasks. By updating these parameters in online training stage, the model is able to learn new tasks in short time without forgetting the old ones. Furthermore, we surprisingly find that decent results can be achieved by only training a small subset of parameters (e.g. 0.1%). This observation enables us to largely decrease the storage of the snapshot, which is important for the system requiring frequent update.

References

Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. *arXiv preprint arXiv:1909.08478*.

Elad Ben Zaken, Shauli Ravfogel, and Yoav Goldberg. 2021. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. *arXiv e-prints*, pages arXiv–2106.

Francisco M Castro, Manuel J Marín-Jiménez, Nicolás Guil, Cordelia Schmid, and Karteek Alahari. 2018. End-to-end incremental learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 233–248.

Zhiyuan Chen and Bing Liu. 2018. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207.

Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv preprint arXiv:1701.08734*.

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

Demi Guo, Alexander M Rush, and Yoon Kim. 2020. Parameter-efficient transfer learning with diff pruning. *arXiv preprint arXiv:2012.07463*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. 2019. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 831–839.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3651–3657, Florence, Italy. Association for Computational Linguistics.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

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

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

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*.

Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7765–7773.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Fei Mi, Lingjing Kong, Tao Lin, Kaicheng Yu, and Boi Faltings. 2020a. Generalized class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 240–241.

Fei Mi, Xiaoyu Lin, and Boi Faltings. 2020b. Ader: Adaptively distilled exemplar replay towards continual learning for session-based recommendation. In

Fourteenth ACM Conference on Recommender Systems, pages 408–413.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Tiago Ramalho and Marta Garnelo. 2019. Adaptive posterior learning: few-shot learning with a surprise-based memory module. *arXiv preprint arXiv:1902.02527*.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017a. Learning multiple visual domains with residual adapters. *arXiv preprint arXiv:1705.08045*.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017b. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

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

Joan Serra, Didac Suris, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming catastrophic forgetting with hard attention to the task. In *International Conference on Machine Learning*, pages 4548–4557. PMLR.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Yi-Lin Sung, Varun Nair, and Colin A Raffel. 2021. Training neural networks with fixed sparse masks. *Advances in Neural Information Processing Systems*, 34.

Ming Tu, Visar Berisha, Martin Woolf, Jae-sun Seo, and Yu Cao. 2016. Ranking the parameters of deep neural networks using the fisher information. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2647–2651. IEEE.

Max Welling. 2009. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1121–1128.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. 2019. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 374–382.

Ju Xu and Zhanxing Zhu. 2018. Reinforced continual learning. *arXiv preprint arXiv:1805.12369*.

Runxin Xu, Fuli Luo, Zhiyuan Zhang, Chuanqi Tan, Baobao Chang, Songfang Huang, and Fei Huang. 2021. Raise a child in large language model: Towards effective and generalizable fine-tuning. *arXiv preprint arXiv:2109.05687*.

Bowen Zhao, Xi Xiao, Guojun Gan, Bin Zhang, and Shu-Tao Xia. 2020. Maintaining discrimination and fairness in class incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13208–13217.

Zhe Zhao, Hui Chen, Jinbin Zhang, Wayne Xin Zhao, Tao Liu, Wei Lu, Xi Chen, Haotang Deng, Qi Ju, and Xiaoyong Du. 2019. Uer: An open-source toolkit for pre-training models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 241–246.

Zhe Zhao, Tao Liu, Shen Li, Bofang Li, and Xiaoyong Du. 2017. Ngram2vec: Learning improved word representations from ngram co-occurrence statistics. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 244–253.

A Task order

We use the following task orders (chosen randomly) for text classification:

- Kindle_Store → Arts_Crafts_and_Sewing → Electronics → Magazine_Subscriptions → Pet_Supplies → Sports_and_Outdoors → Prime_Pantry → Office_Products

→ Movies_and_TV → Automotive → CDs_and_Vinyl → Gift_Cards → Digital_Music → Clothing_Shoes_and_Jewelry → Home_and_Kitchen → Software → Grocery_and_Gourmet_Food

- Pet_Supplies → Gift_Cards → Electronics → Prime_Pantry → Office_Products → Digital_Music → Magazine_Subscriptions → Home_and_Kitchen → CDs_and_Vinyl → Grocery_and_Gourmet_Food

- Digital_Music → Arts_Crafts_and_Sewing → Office_Products → Magazine_Subscriptions → Kindle_Store → Software → Automotive → Prime_Pantry → Grocery_and_Gourmet_Food → Movies_and_TV → Electronics → Home_and_Kitchen → Pet_Supplies → CDs_and_Vinyl → Clothing_Shoes_and_Jewelry → Gift_Cards

- Magazine_Subscriptions → Sports_and_Outdoors → Digital_Music → Electronics → Prime_Pantry → CDs_and_Vinyl → Grocery_and_Gourmet_Food → Home_and_Kitchen → Software → Arts_Crafts_and_Sewing → Clothing_Shoes_and_Jewelry → Pet_Supplies → Office_Products → Kindle_Store → Gift_Cards