Rehearsal-free Continual Language Learning via Efficient Parameter Isolation

Zhicheng Wang[1], Yufang Liu[1], Tao Ji[1], Xiaoling Wang[1], Yuanbin Wu[1]
Congcong Jiang[2], Ye Chao[2], Zhencong Han[2], Ling Wang[2], Xu Shao[2], Wenqiu Zeng[2]
[1]School of Computer Science and Technology, East China Normal University
[2]Information Technology Department, Huatai Securities
zcwang@stu.ecnu.edu.cn ybwu@cs.ecnu.edu.cn

Abstract

We study the problem of defying catastrophic forgetting when learning a series of language processing tasks. Compared with previous methods, we emphasize the importance of not caching history tasks' data, which makes the problem more challenging. Our proposed method applies the parameter isolation strategy. For each task, it allocates a small portion of private parameters and learns them with a shared pre-trained model. To load correct parameters at testing time, we introduce a simple yet effective non-parametric method. Experiments on continual language learning benchmarks show that our method is significantly better than all existing no-data-cache methods, and is comparable (or even better) than those using historical data[1].

1 Introduction

Deployment of NLP models could be dynamic in real-world scenarios: models need to evolve continually when coming in new tasks or new domains (updating an event detection model to handle new event types, for example). Recent studies on continual (language) learning (Biesialska et al., 2020; De Lange et al., 2021) show that, compared with sticking with one single unchanging task, incrementally updating a model for a series of tasks is challenging: learning a new task will make our model perform poorly on previously learned tasks. The so-called *catastrophic forgetting* phenomenon is the central research topic in continual learning (Kirkpatrick et al., 2017).

One method for mitigating forgetting is through data replay (*rehearsal*). The model caches some previous tasks' training data along the sequential learning process. When learning a new task, all previous tasks are re-learned with the current task. The vanilla rehearsal-based approach performs well on many continual learning problems (Rebuffi et al., 2017; Rolnick et al., 2019; de Masson d'Autume et al., 2019), but access to previous tasks' data could conflict with the original intention of continual learning. In fact, if previous tasks' data are given, it approaches multi-task learning where all tasks are given at once and forgetting is not a problem. Meanwhile, in some situations, previous datasets are not available due to regulation or privacy considerations. Therefore, it is essential to look into rehearsal-free methods.

In this work, we study parameter isolation strategies for rehearsal-free continual language learning. Basically, each time a new task comes in, such a strategy allocates a new set of model parameters for that task, which aims to prevent potential interference with already learned knowledge, so that forgetting could be alleviated (Rusu et al., 2016; Mallya and Lazebnik, 2018). Three cruxes of making the idea work in practice are,

- for a possibly large number of tasks, we need to control the storage budget for each task.
- given a set of learned models, we need to identify the right model to query for a test sample.
- besides alleviating forgetting, we also would like to facilitate information sharing among tasks.

We give a solution for the above cruxes under the framework of parameter efficient tuning (PET) of pre-trained language models (PLMs). For each task in the continual learning sequence, we build a model for it using two sets of parameters: one is from a frozen PLM, another is an additional small set of new parameters (namely, *delta parameters* (Ding et al., 2022)). PET methods (Li and Liang, 2021; Hu et al., 2022; Liu et al., 2022a) show that by only fine-tuning delta parameters, performances of downstream language learning tasks could be competitive with (or even better than) the full-scale fine-tuning of the PLM. Therefore, we can share

[1] Code is available at https://github.com/Dicer-Zz/EPI

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 10933–10946
July 9-14, 2023 ©2023 Association for Computational Linguistics

the PLM among all tasks in continual learning, while keeping a private delta parameter for each task, which is usually negligible compared with the PLM (less than 0.3% of the PLM's parameters in our experiments).

Second, to determine which model to query in testing time, we propose using a nonparametric task identification method. Specifically, for each task, we record the first and the second moment of its training samples, and approximate the task's input distribution with a Gaussian. Given a sample, we test whether it belongs to a task using its Mahalanobis distance to the task's Gaussian (Lee et al., 2018; Ren et al., 2021). We show that, compared with state-of-the-art parametric task identifiers (Wang et al., 2022b,a), the nonparametric method significantly boosts the accuracy of getting the correct model (and performances of continual learning) despite its simplicity.

Third, to enhance knowledge transfer among continually learned tasks, we investigate information sharing methods among delta parameters of each task. We show that, by simply initializing the current task's delta parameters with those of previous tasks (either by directly copying from them or soft selecting via attention mechanisms), the transfer of knowledge from learned tasks could be improved, especially in few-shot situations.

We conduct extensive evaluations on standard continual language learning benchmarks. The results show that our algorithm not only outperforms all existing rehearsal-free methods by a large margin (40% F1), but also is competitive with (or even better than) state-of-the-art rehearsal-based methods with the standard setting of data cache size.

2 Problem Definition

Continual learning aims to solve problems in streams: a model no longer faces a single unchanging task but a series of tasks arrived sequentially. Following previous works, we focus on task streams containing text classification problems, while the algorithm could be extended to other language learning problems.

Denote $\mathcal{T} = \{T_1, T_2, \cdots, T_{|\mathcal{T}|}\}$ as a sequence of tasks, and the training set of the t-th task T_t is $\{(x_i^t, y_i^t)\}$, where $x_i^t \in \mathcal{X}_t$ is an input text, $y_i^t \in \mathcal{Y}_t$ is its class label. At timestamp t, a continual learning model learns a mapping $f_\theta : \cup_{\tau=1}^t \mathcal{X}_\tau \mapsto \cup_{\tau=1}^t \mathcal{Y}_\tau$. The key property is that the model can only query the training data of T_t, and aims to update the model parameter θ to not only predict the t-th task, but also keep performances on all previous tasks (even without using their data). In the testing phase, the model predicts a sample's label from all seen labels $\cup_{\tau=1}^t \mathcal{Y}_\tau$ without knowing which task it belongs to (*class incremental*).

In this work, we will focus on the above *rehearsal-free* setting of continual learning. Several relaxations of this setting could be applied to upper-bound its performances,

- *rehearsal-based* continual learning, where some previous tasks' data can be used when training the current task. If all previous data are given, the problem is reduced to multi-task learning.
- *task incremental* continual learning, where task labels are given in testing time. It provides additional information about samples and makes prediction easier than the class incremental setting.

3 Approaches

Our method is summarized in Figure 1. In the training stage, for a new task, we assign (and save) a new private delta parameter for it and train the parameter (jointly with a shared frozen PLM) on the task's dataset. We also save some statistics about the training set to model the task as a Gaussian distribution (e.g., the averaged vector representations of samples). In the testing stage, for a given sample, we fetch a proper learned delta parameter for predicting its label by consulting the Mahalanobis distance between the sample and the distributions of all previously encountered tasks. In the following, we start by describing the parameter isolation strategy with PET methods (Section 3.1). Then, we illustrate the task identification method used in the testing stage (Section 3.2) and knowledge sharing methods (Section 3.3).

3.1 Parameter Isolation with PET

One reason that causes catastrophic forgetting in continual learning is the interference among tasks: different tasks may guide the model to move towards different directions. Therefore, a simple way to alleviate forgetting is to use different parameters for different tasks. On the other side, a sufficient model capacity for each task is crucial to get high-performance models. Directly separating models could make large storage costs. Here, we adopt the parameter-efficient tuning framework which is able

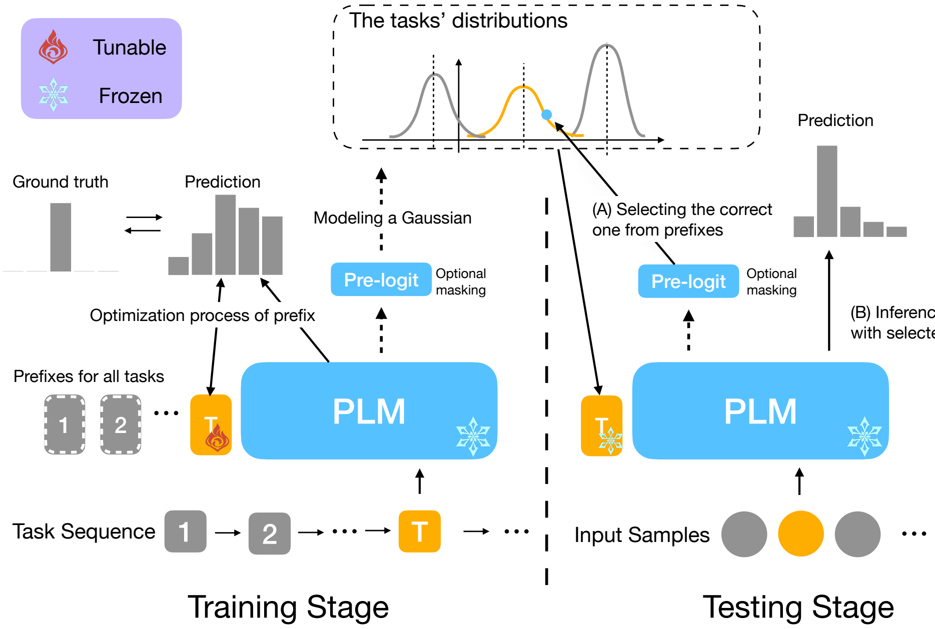

Figure 1: The process of our method. The left panel of the figure shows the training process. We counted the class prototypes and covariance matrix of each dataset while training. The right panel shows the inference process, which consists of two forward propagations: (A) obtain a PET module suitable for the input sample by PLM. (B) predict the label of it by PLM with the selected module.

to take advantage of a fixed powerful pre-trained model while keeping individual tasks moderately separated. In the following, we briefly review prefix tuning (Li and Liang, 2021) (the main PET method used in our experiments). [2]

Denote a Transformer-based pre-trained model with multi-head attention blocks to be $\mathrm{MHA}(\mathbf{X})$. It maps an input sentence $\mathbf{X} \in \mathbb{R}^{n\times d}$ to a hidden semantic space $\mathbf{H} \in \mathbb{R}^{n\times d}$, where n is the sentence length, d is the dimension of token embeddings and hidden vectors. For the t-th task in continual learning, we prepare a set of task-specific parameters by prepending p soft virtual tokens to the input of multi-head attention blocks at each layer. Denote parameters in the prefix as $\mathbf{P} \in \mathbb{R}^{p\times d\times L}$, where L is the number of MHA layers in the pre-trained model. [3]

Besides prefixes $\mathbf{P}$, another set of task-specific parameters are classification heads (we apply softmax activation for the multi-class classification). [4] During the training process, parameters of the PLM are frozen (thus, shared among all tasks), and only private prefixes and classification heads are trained.

Private parameters are saved after training. One key property of PET methods is that one can train well enough models for tasks with a small size of $\mathbf{P}$ and a powerful PLM. It facilitates reducing the storage cost for parameter isolation methods. while retaining high-performance models for individual tasks (e.g., decrease by 1.34% in a benchmark dataset). In our experiments, the size of $\mathbf{P}$ is 0.27% of the PLM. Hence, the same storage budget for two tasks with full PLM fine-tuning can support 370 tasks with prefix tuning.

3.2 Task Identification

As we use different prefixes for different tasks, when a test sample comes, it is necessary to determine which prefix should be applied. One approach is to try all learned prefixes, and choose one with the largest prediction probability on the sample. In practice, this method suffers from the over-confidence problem of softmax predictors: even for those unrelated tasks, a prefix might give a prediction with high probability.

Another approach is to compare the test sample with training samples and choose its nearest neighbor's prefix (Mensink et al., 2013). It works around the over-confidence problem, but still depends on

[2] The same idea can be applied with other PET methods (e.g., prompt-tuning (Hu et al., 2022), LoRA (Liu et al., 2022a)).

[3] In the implementation, virtual tokens are added by concatenating two $\mathbb{R}^{p\times d}$ matrices to key and value matrices.

[4] For simplicity, we assume tasks have disjoint label spaces, thus their classification heads are different. In practice, those heads could be shared.

the robustness of sample representation and distance metrics. Here, inspired by recent studies on out-of-distribution detection (Lee et al., 2018; Ren et al., 2021), we improve this method by comparing the test sample with full distributions of tasks' training samples. In the following, we represent an input x with the average of the last Transformer block's outputs of the PLM, denoted by $h(x)$ (i.e., the "pre-logit" vector).

First, for the t-th task, we perform a Guassian discriminant analysis as suggested by the softmax classifier: given $p(y = c|h(x))$ in the form of softmax ($c \in \mathcal{Y}_t$ is a class label), we can assume $p(h(x)|y = c)$ is a Gaussian with estimated mean μ_t^c and a shared covariance $\mathbf{\Sigma}_t$ among classes,

$$\mu_t^c = \frac{1}{N_c} \sum_{y_i = c} h(x_i), \tag{1}$$

$$\mathbf{\Sigma}_t = \frac{1}{N} \sum_c \sum_{y_i = c} \left(h(x_i) - \mu_t^c\right)\left(h(x_i) - \mu_t^c\right)^\top \tag{2}$$

where N_c is the number of training samples with label c, and the values are obtained by maximum likelihood estimation.

Next, for a test sample x, we compare it with all tasks' Gaussian and choose the nearest task's prefix as the prefix for predicting x's label. Unlike computing distance between samples, we need to apply metrics measuring distance between samples and distributions. Here, we use Mahalanobis distance. Specifically, the distance between x and class c is

$$-\left(h(x) - \mu_t^c\right)^\top \left(\mathbf{\Sigma}_t\right)^{-1} \left(h(x) - \mu_t^c\right). \tag{3}$$

In practice, we find that directly ranking tasks with $\mathbf{\Sigma}_t$ makes the distances have large numeric deviation. To make the computation more stable, we further share the covariance among all tasks $\mathbf{\Sigma} = \sum_t \mathbf{\Sigma_t}$ and change the computation of Mahalanobis distance accordingly.

To implement above non-parametric task identification method, we need additional storage for store class means $\{\mu_t^c\}$ and a shared covariance $\mathbf{\Sigma}$. Furthermore, though they are moments of distributions, there may be a chance to get information about individual samples (Dwork and Roth, 2014). Here, inspired by querying with randomized response in differential privacy, we propose to add randomly masking on sample representations $h(x)$. Specifically, we assign a random mask (with $q\%$ entries 0, other entries 1) for all tasks. For each $h(x)$, the masked dimensions are dropped during the computation of μ_t^c and $\mathbf{\Sigma}_t^c$. The model saves the masked vectors and the mask itself (or the random seed generating the mask) for testing time distance computation. Besides encrypting moments, the simple masking strategy also helps to reduce storage cost.

3.3 Knowledge Transfer

Separating parameters is effective in mitigating catastrophic forgetting, however, it also blocks knowledge transfer among tasks. Given a sequence of learned prefixes $\mathbf{P} = \{\mathbf{P}_1, ..., \mathbf{P}_{t-1}\}$, we try several ways to utilize the knowledge acquired from preceding tasks in the hope that they could improve and accelerate the current task learning,

- **Prefix Fusion.** A natural way to combine knowledge from previous tasks is through the use of the attention mechanism. This allows the model to automatically extract useful information from previous tasks and integrate it into the current learning process. To achieve this, we prepend the learned prefixes of previous tasks to the prefix of the current task. Knowledge transfer is automatically facilitated through the multi-head attention mechanism of the Transformer. During new prefix learning, we fix the prefixes of previous tasks to avoid parameter drifting and catastrophic forgetting.

- **Prefix Initialization.** Training starting from well-trained parameters is another way to promote knowledge transfer, thus we can initialize a prefix with previously learned prefixes instead of random initialization. A good initial point can also help to speed up the convergence of the training process of PET methods. We try two ways, namely initialized with the *last prefix* $\mathbf{P}_t \leftarrow \mathbf{P}_{t-1}$ and initialized with the *mean prefix* $\mathbf{P}_t \leftarrow \frac{1}{t-1} \sum_{\tau=1}^{t-1} \mathbf{P}_\tau$.

4 Experiments

Datasets To demonstrate the generalizability of our approach, we use two kinds of datasets, differentiated according to the domain relevance between tasks. ***Far-domain***, where the domain boundary among tasks are clear. Following MbPA++ (de Masson d'Autume et al., 2019) and IDBR (Huang et al., 2021a), we use 5-datasets collected by Zhang et al. (2015) to evaluate our method. It consists AG News (news), Yelp (business reviews), Amazon (product reviews), Yahoo!Answer (Q&A), and DBPedia (encyclopedic articles). These datasets are categorized into two text classification tasks: topic classification (AG News, Yahoo!Answers, and DBPedia) and sentiment classification (Yelp and Amazon). ***Near-domain*** where the tasks are more closely related. We use Web of Science (WOS) from Kowsari et al. (2017) and 20 Newsgroups from Lang (1995) to assess our method for datasets with high inter-task relevance. WOS contains seven parent classes and five sub-classes under each parent class which have close relations. We organize continual learning tasks according to parent classes. The 20 Newsgroups consists six topics of news. We rearranged it into four tasks based on the principle of maximizing inter-task correlation. The details of the two datasets are in Appendix B.

Metrics Let $a_{i,j}$ be the testing accuracy on the i-th task after training on j-th task, the metrics for evaluating are,

- **Performance of Continual Learning (CL)**. The average accuracy of all tasks after training on the last task, $\frac{1}{|\mathcal{T}|}\sum_{i=1}^{|\mathcal{T}|} a_{i,|\mathcal{T}|}$
- **Forgetting (Fgt)**. The degree of forgetting of previous tasks after training on the last task, $\frac{1}{|\mathcal{T}|-1}\sum_{i=1}^{|\mathcal{T}|-1}(\max_{k=i}^{|\mathcal{T}|-1} a_{i,k} - a_{i,|\mathcal{T}|})$
- **Accuracy of Task Identification (TI)**. The accuracy of getting the correct prefix for testing samples after training on all tasks.

Baselines We use the following continual learning techniques as baselines:

- **FT**, fine-tuning a model for each task sequentially while catastrophic forgetting occurs. This method is the lower bound of continual learning.
- **MTL**, training a model on all tasks as multi-task learning. This method is the upper bound of continual learning.
- **Replay**, saving part of the previous tasks as memory, train a model one step on the memory after every β (10 for our experiments) steps of training on the new task.
- **LwF** (Li and Hoiem, 2017), a typical regularization-based approach. We also combine LwF with the replay method as an enhancement.
- **IDBR** (Huang et al., 2021b), disentangling information by two simple auxiliary tasks (next sentence prediction and task-id prediction) for learning better generic and specific representation spaces. This approach applies both episodic memory replay and regularization techniques.
- **L2P** (Wang et al., 2022b), it first introduces a prompt-based framework to continual learning. The main difference between L2P and our method is that L2P uses a parametric task identifier, while our identifier is non-parametric. We also adopt L2P-R, which is L2P equipped with a rehearsal buffer.

To ensure a fair comparison, **FT, MTL, Replay** and **LwF** are all prefix-based. We re-implement **L2P** to make it support prefixes (instead of only prompts in the original code).

Details We use BERT (Devlin et al., 2019) from HuggingFace Transformers (Wolf et al., 2020) as the PLM of our model. We set the default prefixes length to 16. Because our methods progressively assign a prefix to new tasks, we assign the same number of parameters to all baseline methods to ensure a fair comparison. See Appendix C for other configurations.

4.1 Main Results

In Table 1, we present the results of our method and baselines on the sampled 5-datasets and WOS. On the sampled 5-datasets, the rehearsal-based approach stores 50 samples per class, equivalent to 2.5% of the entire training dataset (except for IDBR, which is 20 per class). From the results, we can find that,

- On sampled 5-datasets, our approach surpasses all rehearsal-free methods by a large margin. It is even better than rehearsal-based methods: compared with the previous SOTA method IDBR, our method improves the accuracy from 73.19% to 74.43%, reducing the gap to the upper bound (75.40%) by 56%.
- When changing task orders of the sampled 5-datasets, the standard deviation of our method is small, which indicates that it is insensitive to task orders, which is rare in approaches without replaying.
- On the more challenging WOS, our approach is still comparable to replay methods that use approximately 10% training data (20 samples per class).

Method	Buffer Size		5-datasets		Average	Buffer Size	WOS
		Order 1	Order 2	Order 3			
Replay		67.87±0.18	68.27±0.46	69.01±0.10	68.38±0.47		**77.86**±0.48
+LwF	50/class	70.08±0.15	70.12±0.09	69.65±0.36	69.95±0.21	20/class	76.78±0.66
L2P-R		68.02±0.27	67.90±0.16	68.49±0.27	68.14±0.25		77.10±0.44
IDBR[†]	(20/class)	72.63	73.72	73.23	73.19±0.44		–
FT		29.54±0.63	36.06±0.88	24.65±1.16	30.08±4.67		53.86±1.64
LwF	0	27.40±0.98	31.27±0.91	23.34±2.01	27.33±3.24	0	30.96±0.48
L2P		30.34±0.32	35.78±0.12	23.45±0.47	29.86±5.05		54.62±0.85
Ours		**74.38**±**0.04**	**74.53**±**0.07**	**74.38**±**0.01**	**74.43**±**0.07**		77.83±**0.14**
MTL	–	75.40	75.40	75.40	75.40	–	85.25

Table 1: Summarization of results on sampled setting of 5-datasets and WOS. We report the average accuracy and standard deviation over 3 runs. † denotes the results come from (Huang et al., 2021b). Dash line indicates data is not available or not meaningful. The best results of all methods are bolded, and the best results of rehearsal-free methods are underlined. We also bold the smallest standard deviation of all methods, where a smaller one indicates a more robust method.

Method	MR	TT	TI	LA			5-datasets		
Order					**1**	**2**	**3**	**4**	**Average**
MbPA++[†]	✓			✓	74.9	73.1	74.9	74.1	74.3
LAMOL[†]	✓	✓	✓		76.1	76.1	77.2	76.7	76.5
IDBR[†]	✓	✓			75.9	76.2	76.4	76.7	76.3
Ours		✓			**77.4**	**77.3**	**77.2**	**77.4**	**77.3**
Ours (oracle)		✓	✓		79.1	79.0	78.9	78.9	79.0

Table 2: Results on the full setting of 5-datasets. All results are averaged over 3 runs. † denotes the results come from (Huang et al., 2021b). To compare different methods, four method features have been defined, with a checkmark indicating the presence of the corresponding feature. Specifically, **MR** indicates whether the method requires replaying stored memory (rehearsal-based or rehearsal-free). **TT** indicates whether task-id is available during training. **TI** indicates whether task-id is available during inference. **LA** indicates whether performing local adaptation during inference.

Table 2 presents the performances on the full 5-datasets. The conclusions drawn from this table are generally consistent with Table 1, but it also showcases some new findings:

- Our rehearsal-free method, even without task ID during the testing phase (class incremental), can outperform previous SOTA methods such as LAMOL (Sun et al., 2020), which relies on rehearsal and has knowledge of the task ID during testing (task incremental).
- When provided with the task ID ("oracle"), our method's performance can be further improved by 1.7%. This suggests that there is still potential for enhancing our approach.

4.2 Discussions

To further inspect the proposed methods, we investigate the following research questions.

How the task identifier performs? To show the validity of the Mahalanobis distance (MD), we compare it with two methods: (i) Maximum over softmax probability (MSP): using all prefixes for inference (task by task) and then choosing the label with the highest probability; (ii) Euclidean distance (ED) which ignores the covariance in Equation 3. The results are in Figure 2. We find that,

- MSP performs poorly (especially on 5-datasets). It shows that, at least for the vanilla MSP metric, the over-confidence problem is still essential. Another drawback of MSP is that it has to perform forward passes with all prefixes while our method only needs to compare vectors.
- Mahalanobis distance is always better than Euclidean distance for task identification. Therefore, different dimensions of sample representations should have different importance for detecting tasks, and modeling tasks' distribution with anisotropic Gaussians provides a better approximation of the actual distribution.

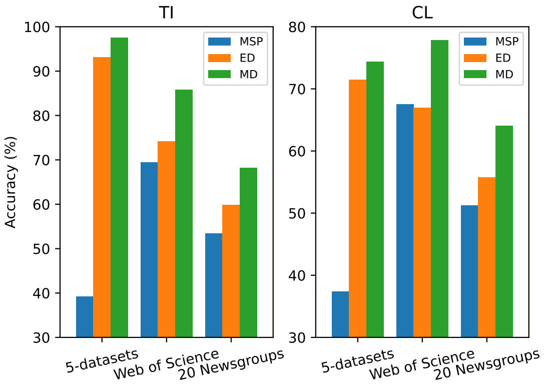

Figure 2: Comparisons of prompt selection strategies on three datasets. We report the average result over 3 runs. MSP: Maximum Softmax Probability. ED: Euclidean Distance. MD: Mahalanobis Distance.

Method	Parametric	5-datasets	WOS	20 Newsgroups
Euclidean	no	62.82	53.61	53.43
+ Prefix	yes	71.44	66.99	55.78
Mahalanobis	no	69.78	72.36	63.08
+ Prefix	yes	**74.38**	**77.84**	**64.05**

Table 3: Comparing non-parametric and parametric classifiers. The best results is highlighted in bold.

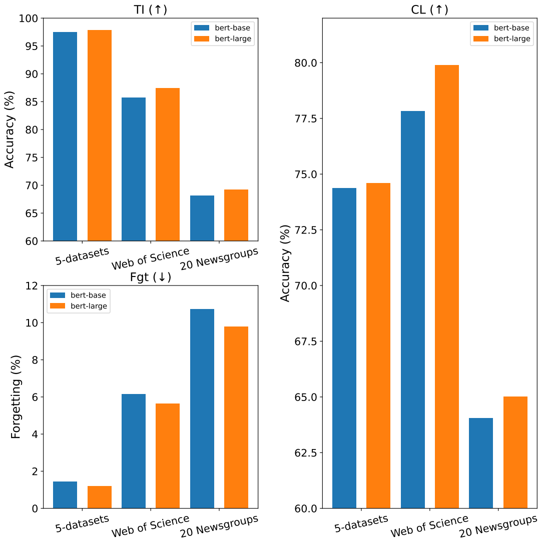

Figure 3: The results of bert-base (~110M) and bert-large (~340M) on three datasets. ↑ means larger is better, and ↓ means smaller is better.

Does prefix tuning help continual language learning? We can directly use the non-parametric task identifier to infer class labels of samples. Hence, one question is whether the additional prefix parameters provide performance gains. We build two non-parametric classifiers which use Euclidean and Mahalanobis distance. The samples are encoded only with representations from the PLM (without prefixes). As shown in Table 3, we can find that,

- Regarding the performance of continual learning (CL), the two non-parametric classifiers perform quite well: the classifier with Euclidean distance performs much better (62.8%) than all rehearsal-free method in Table 1 ($< 30\%$) on 5-datasets, and is competitive on WOS. The classifier with Mahalanobis distance is more effective (69.8%): it is even competitive with the primary reply method (**Replay**). The success of non-parametric classifiers implies that the powerful representation ability from PLM is crucial to perform continual learning.

- With the help of parametric prefix tuning, the performances of the two methods are largely boosted. It proves that task-specific information is also important. Regarding the performances of direct fine-tuning in Table 1, using separate prefixes could alleviate catastrophic forgetting.

How do different PLMs influence performances? The above analyses on non-parametric classifiers suggest us to explore the influence of different PLMs (Figure 3 and Table 4). We find that,

- Owing to the presence of a larger number of parameters, the sentence representations generated by larger models enhance task identification. Moreover, previous studies also suggest that the larger the PLM, the more effective the PET (Lester et al., 2021; Liu et al., 2022b). Figure 3 presents the performance across model scales, which are consistent with the previous findings in continual language learning.

- The PLM pre-trained on the same domain of continual learning tasks is able to extract a more valuable representation. For WOS, we replace BERT with SciBERT (Beltagy et al., 2019), which is a BERT-like model pre-trained on the scientific (WOS domain) corpus. As shown in Table 4, the closer SciBERT performs even better than bert-large though it is smaller (with the same size as bert-base). Therefore, if some unsupervised texts from the same domain of continual learning are given, one could fine-tune the PLM for a better performance.

Model	Size	TI	CL	Fgt
bert-base	110 M	85.78	77.84	6.14
bert-large	340 M	87.48	79.90	5.63
sci-bert-base	110 M	**87.98**	**79.92**	**5.01**

Table 4: Comparisons between models of varying sizes (base vs. large) or trained on different datasets (bert vs. sci-bert) on the WOS dataset. The average results over 3 runs are reported, with the best results highlighted in bold.

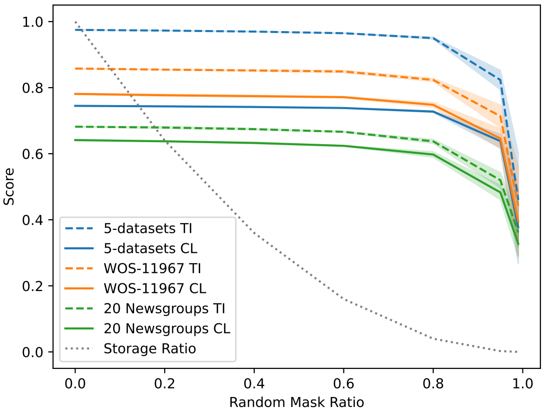

Figure 4: Accuracy of task identification and continual learning vs. random masking ratio. The results are averaged over 3 runs. Storage Ratio: The ratio of storage space required for mask to that without mask.

How random masking influence the task identifier? To validate the effect of random masking, we use five masking ratios equally spaced from 0% to 80% and two high masking ratios (95% and 99%) The results are in Figure 4. Our method shows a slight performance degradation in the range of masking ratio from 0% to 80%. Compared with no mask (i.e., masking ratio equal to 0), a masking ratio of 80% still gives more than 90% of the performance, and it only needs about 4% of the storage space for all three datasets. It suggests that pre-logit vectors ($h(x)$) is highly redundant for distinguishing tasks. However, as the masking ratio increases further, the task identification performances decrease rapidly, which causes the end continual learning performances to decrease as well.

Does knowledge transferability of prefixes matter? Finally, we evaluate knowledge transfer methods (Section 3.3) in few-shot settings. As shown in Table 5, we observe knowledge transfer in all settings on WOS for all transfer methods but only in the 10-shot setting on 5-datasets for **Mean**. This is because the tasks in the WOS dataset are related, and thus knowledge can be shared among them. Moreover, we do not observe forward transfer in the full-shot setting on any benchmark, because knowledge transfer is unnecessary when the data are sufficient.

Datasets	# shot	None	Fusion	Mean	Last
5-datasets	50	**63.56**	62.99	62.98	63.35
	20	**55.30**	54.58	54.44	54.41
	10	54.10	53.89	**54.31**	53.89
WOS	50	67.20	67.32	**68.28**	67.75
	20	55.13	55.69	**56.18**	55.97
	10	51.21	52.36	52.83	**52.91**

Table 5: The results of knowledge transfer methods in few-shot settings. **Fusion** refers to **Prefix Fusion**. **Mean** (**Last**) refers to initialize a new prefix as *mean prefix* (*last prefix*).

5 Related work

Continual Learning We discuss three main categories of continual learning methods: *rehearsal-based*, *regularization-based*, and *parameter isolation* methods.

Rehearsal-based methods alleviate catastrophic forgetting by replaying stored examples (Rebuffi et al., 2017; Rolnick et al., 2019; de Masson d'Autume et al., 2019) or pseudo-generative examples (Shin et al., 2017; Su et al., 2020; Sun et al., 2020) of previous tasks. Unfortunately, all of them carry the risk of privacy leakage and need nontrivial storage space.

Regularization-based methods restrict the updating of the model weights by knowledge distillation (Li and Hoiem, 2017; Triki et al., 2017) or parameter importance (Kirkpatrick et al., 2016; Zenke et al., 2017; Aljundi et al., 2018) to preserve the knowledge of previous tasks. However, these methods emphasize the model's stability to previous tasks while weakening its plasticity to the new task (Parisi et al., 2019).

Parameter isolation methods assign a task-specific parameter to a new task by splitting (Fernando et al., 2017; Mallya and Lazebnik, 2018; Serrà et al., 2018) or extending (Rusu et al., 2016; Xu and Zhu, 2018) the current model to prevent interference between tasks. Because these methods require task-id to choose the proper model at testing time, they only apply to task incremental continual learning. Although our method also belongs to this category, our method applies to class incremental by introducing task identification.

6 Conclusion

In this work, we propose a new rehearsal-free parameter isolation continual learning method that leverages the capabilities of a pre-trained language model (PLM). Extensive experiments show that our method surpasses all rehearsal-free methods by a significant margin and is comparable (or even better) than previous start-of-the-art rehearsal-based methods on two benchmarks, whether the tasks are near or far. Meanwhile, we introduce random static masking to reduce the storage required by our method to adapt it to more demanding scenarios.

Limitations

Although our proposed knowledge transfer methods work well on WOS in the few-shot setting, it is less effective on 5-datasets. Moreover, all methods fail in the full-shot setting. Based on our approach, a more general approach to knowledge transfer is expected in future works. In addition, our approach requires a well-trained language model for task identification and a Transformer-based model (well-trained also) for parameter efficient tuning. Therefore, it is challenging to cooperate our approach with a language model with random initialization or non-transformer architecture.

Acknowledgement

The authors wish to thank all reviewers for their helpful comments and suggestions. The corresponding authors are Zhicheng Wang, Tao Ji and Yuanbin Wu. This research was (partially) supported by NSFC(62076097), National Key R&D Program of China (2021YFC3340700), and East China Normal University International Conference Grant Programme.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III*, volume 11207 of *Lecture Notes in Computer Science*, pages 144–161. Springer.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. 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)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Magdalena Biesialska, Katarzyna Biesialska, and Marta R Costa-jussà. 2020. Continual lifelong learning in natural language processing: A survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6523–6541.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385.

Cyprien de Masson d'Autume, Sebastian Ruder, Lingpeng Kong, and Dani Yogatama. 2019. Episodic memory in lifelong language learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 13122–13131.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

Cynthia Dwork and Aaron Roth. 2014. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407.

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. *CoRR*, abs/1701.08734.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021a. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2736–2746. Association for Computational Linguistics.

Yufan Huang, Yanzhe Zhang, Jiaao Chen, Xuezhi Wang, and Diyi Yang. 2021b. Continual learning for text classification with information disentanglement based regularization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2736–2746, Online. Association for Computational Linguistics.

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.

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.

Kamran Kowsari, Donald E Brown, Mojtaba Heidarysafa, Kiana Jafari Meimandi, Matthew S Gerber, and Laura E Barnes. 2017. Hdltex: Hierarchical deep learning for text classification. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 364–371. IEEE.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339.

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. 2018. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 7167–7177.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

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

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022a. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7765–7773. Computer Vision Foundation / IEEE Computer Society.

Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. 2013. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2624–2637.

German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71.

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

Jie Ren, Stanislav Fort, Jeremiah Liu, Abhijit Guha Roy, Shreyas Padhy, and Balaji Lakshminarayanan. 2021. A simple fix to mahalanobis distance for improving near-ood detection. *CoRR*, abs/2106.09022.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. 2019. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32.

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

Joan Serrà, Didac Suris, Marius Miron, and Alexandros Karatzoglou. 2018. Overcoming catastrophic forgetting with hard attention to the task. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4555–4564. PMLR.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. *CoRR*, abs/1705.08690.

Xin Su, Shangqi Guo, Tian Tan, and Feng Chen. 2020. Generative memory for lifelong learning. *IEEE Trans. Neural Networks Learn. Syst.*, 31(6):1884–1898.

Fan-Keng Sun, Cheng-Hao Ho, and Hung-Yi Lee. 2020. LAMOL: language modeling for lifelong language learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Amal Rannen Triki, Rahaf Aljundi, Matthew B. Blaschko, and Tinne Tuytelaars. 2017. Encoder based lifelong learning. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1329–1337. IEEE Computer Society.

Zifeng Wang, Zizhao Zhang, Sayna Ebrahimi, Ruoxi Sun, Han Zhang, Chen-Yu Lee, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer G. Dy, and Tomas Pfister. 2022a. Dualprompt: Complementary prompting for rehearsal-free continual learning. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXVI*, volume 13686 of *Lecture Notes in Computer Science*, pages 631–648. Springer.

Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. 2022b. Learning to prompt for continual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 139–149.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 38–45. Association for Computational Linguistics.

Ju Xu and Zhanxing Zhu. 2018. Reinforced continual learning. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 907–916.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3987–3995. PMLR.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Supplementary Material for
Rehearsal-free Continual Language Learning via Efficient Parameter Isolation

A Potential Risks

Pre-trained language models (PLMs) may inherit biases from the training corpus, resulting in offensive behaviors. Combining our approach with these toxic language models and deploying them in realistic scenarios might cause negative social impacts.

B Datasets Details

Order	Task Sequence
1	ag → yelp → amazon → yahoo → dbpedia
2	yelp → yahoo → amazon → dbpedia → ag
3	dbpedia → yahoo → ag → amazon → yelp
4	yelp → ag → dbpedia → amazon → yahoo

Table 6: Four task orders of the 5-datasets.

Task ID	1	2	3	4	5	6	7
# of examples	1499	1132	1959	1925	2107	1617	1728

Table 7: The statistic of original WOS-11967

For 5-datasets, the sampled setting samples 2,000 training examples and 2,000 validation examples from the original training set. The full setting is the same as the dataset used by de Masson d'Autume et al. (2019). We show task orders of 5-datasets we used in Table 6. We did not merge the label space of Yelp and Amazon as MbPA++ and IDBR did to create a more challenging setup. Our label space can be mapped to the label space used by MbPA++ and IDBR and not vice versa.

We download WOS-11967 from Huggingface Datasets (Lhoest et al., 2021), and we show the statistics of it in Table 7. We split the whole dataset to train/val/test set in the ratio of 0.6:0.2:0.2.

We access 20 Newsgroups from Huggingface Datasets corresponding to the 20news-bydata version on the official site. We show the task separation in Table 8. We take one-sixth of the training set as a validation set for a train/val/test splitting ratio 0.5:0.1:0.4.

Class Name	Train	Test
Task 1		
comp.graphics	584	389
rec.autos	594	396
sci.crypt	595	396
misc.forsale	585	390
talk.politics.misc	465	310
talk.religion.misc	377	251
Task 2		
comp.os.ms-windows.misc	591	394
rec.motorcycles	598	398
sci.electronics	591	393
talk.politics.guns	546	364
alt.atheism	480	319
Task 3		
comp.sys.ibm.pc.hardware	590	392
rec.sport.baseball	597	397
sci.med	594	396
talk.politics.mideast	564	376
soc.religion.christian	599	398
Task 4		
comp.sys.mac.hardware	578	385
rec.sport.hockey	600	399
sci.space	593	394
comp.windows.x	593	395

Table 8: The task separation and statistic of original 20 Newsgroups.

C Experimental Details

We train all models using AdamW (Loshchilov and Hutter, 2017) with $\beta_1 = 0.9$ and $\beta_2 = 0.99$ coupled with a linear scheduler with a warm-up ratio of 0.1. For sampled 5-datasets, WOS, and 20 Newsgroups, we set the identical learning rate $\lambda = 0.03$. For the full 5-datasets, we do grid searching for the learning rate for each task respectively, and the final learning rates are $0.003, 0.009, 0.005, 0.007, 0.003$ for AG News, Amazon, Yelp, Yahoo!Answer and DBpedia, respectively. All experiments are conducted on NVIDIA RTX 3090 with 24GB video memory with a batch size of 32 and the maximum length of a sentence is 256.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section Limitations (Line 558-570)

☑ A2. Did you discuss any potential risks of your work?
Appendix A (Line 838-844)

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Abstract and section 1 (Line 1-112)

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4 (Line 297-323)

☑ B1. Did you cite the creators of artifacts you used?
Section 4 (Line 297-323)

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
We do not find any license in original papers.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4 (Line 297-323)

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
All of datasets comes from publicly available news or published articles, so we don't think there are such problems.

☒ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
That information can be found in original papers.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix B (Line 845-865)

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4 and append C (Line 876-879)

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4 and appendix C (Line 867-876)

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 4 (Line 368-370)

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.