UserAdapter: Few-Shot User Learning in Sentiment Analysis

Wanjun Zhong[1*], **Duyu Tang**[2], **Jiahai Wang**[1], **Jian Yin**[1] **and Nan Duan**[2]
[1] The School of Computer Science and Engineering, Sun Yat-sen University.
Guangdong Key Laboratory of Big Data Analysis and Processing, Guangzhou, P.R.China
[2] Microsoft Research
{zhongwj25@mail2, wangjiah@mail,issjyin@mail}.sysu.edu.cn
{dutang,nanduan}@microsoft.com

Abstract

Adapting a model to a handful of personalized data is challenging, especially when it has gigantic parameters, such as a Transformer-based pretrained model. The standard way of fine-tuning all the parameters necessitates storing a huge model for each user. In this work, we introduce a lightweight approach dubbed UserAdapter, which clamps hundred millions of parameters of the Transformer model and optimizes a tiny user-specific vector. We take sentiment analysis as a test bed, and collect datasets of reviews from Yelp and IMDB respectively. Results show that, on both datasets, UserAdapter achieves better accuracy than the standard fine-tuned Transformer-based pre-trained model. More importantly, UserAdapter offers an efficient way to produce a personalized Transformer model with less than 0.5% parameters added for each user.

1 Introduction

Having a bespoke model by only seeing a few data of a user is increasingly important given its merits of producing customized service and protecting user privacy. In this work, we study the learning of personalized model based on Transformer-based pretrained models (Devlin et al., 2018; Liu et al., 2019), which dominate a wide range of natural language understanding problems. A standard way is fine-tuning the whole parameters. However, this is unacceptable in practice because it would result in storing a model with hundreds of millions of parameters for each user. An alternative method is in-context learning, which is adopted in GPT-3 (Brown et al., 2020). A few examples are provided as context to the pretrained model and no fine-tuning is needed. However, limited by the bounded-length context of Transformer, it cannot make full use of the training instances that exceed the context window.

In this work, we introduce UserAdapter, a lightweight method that learns personalized model in a few-shot learning scenario. Our work is inspired by the recent progress on lightweight fine-tuning (Houlsby et al., 2019; Wang et al., 2020; Li and Liang, 2021), where a small number of task-specific parameters are the only trainable ones, while the dominant parameters of Transformer are fixed. In UserAdapter, each user is represented as a continuous vector, and such vector works as a virtual "prefix" token that steers the representations produced by Transformer. When adapting an existing model to a few datapoints of a new user, we clamp the parameters of Transformer and only need to train the tiny user vector. Even if taking the parametrization strategy into account (detailed in Section 3.2), UserAdapter adds less than 0.5% parameters for each user.

As a case study, we conduct experiments on sentiment analysis in this work. The personalized user information is essential in guiding the decision stage of the model because the style and preference of reviews vary among users. We collect datasets of reviews from Yelp and IMDB respectively, and study the task of predicting the rating (e.g., 1-5 or 1-10) for the review content. In the testing stage, reviews are written by users never seen in the training set and each user is attached with a dozen instances used for few-shot learning. Results show that, on both datasets, UserAdapter consistently outperforms completely fine-tuned Transformer-based pretrained model. Taking IMDB dataset as an example, we find that adapting a standard fine-tuned model to unseen users drops the accuracy, while UserAdapter achieves comparable accuracy on unseen users with few-shot learning.

We summarize the major contributions of this

[*] Work done while this author was an intern at Microsoft Research.

work as follows.

- We introduce UserAdapter, a lightweight approach to optimize a personalized Transformer model with tiny trainable parameters.
- We create two datasets to foster research on few-shot personalized sentiment analysis.
- We show that UserAdapter is better than the de facto way of fine-tuning the whole parameters in terms of both accuracy and efficiency.

2 Task and Dataset

We consider the task of few-shot sentiment analysis here. Given a text as the input, the task of sentiment analysis is to predict the sentiment label of the text. More specifically, we study sentiment analysis in a few-shot learning scenario, where (1) instances in the test set are written by users never seen in the training set and (2) each user in the test set is also paired with a dozen of text-label pairs used for few-shot learning.

To the best of our knowledge, there is no existing datasets meeting our demands, so we create two datasets by ourselves. One dataset comes from Diao et al. (2014), where each text is a movie review on IMDB and the sentiment label (rating) is from 1 to 10. The other dataset is from Tang et al. (2015), where each text is a restaurant review from Yelp and the sentiment label is from 1 to 5. Each dataset includes two parts: (1) part A consisting of massive user data for training a general classification model; (2) part B used for few-shot learning. To ensure that each user in part B is never seen in the training set of A, we separate these datasets based on users. To support few-shot learning, we have a constraint on the users in part B that they only write no more than 50 reviews. The data statistics are shown in Table 1.

Dataset	Part	# of reviews	# of users
IMDB	A	127,626	783
	B	10,084	229
Yelp	A	375,029	3,247
	B	53,340	1,213

Table 1: Data statistics of datasets.

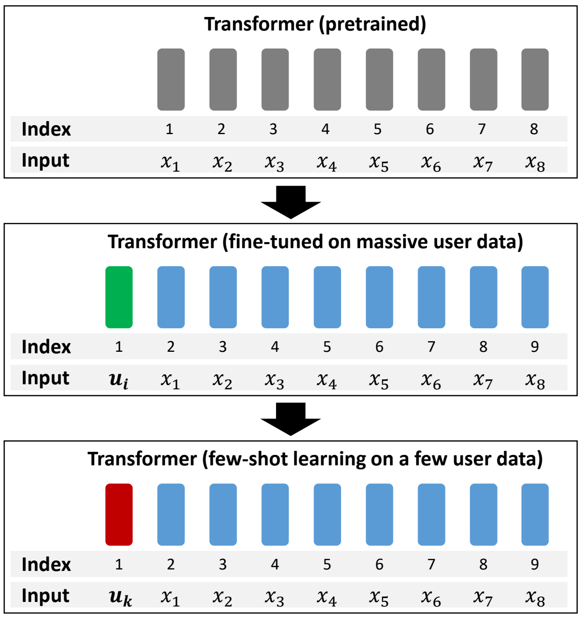
Transformer (pretrained)
Index 1 2 3 4 5 6 7 8
Input x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
Transformer (fine-tuned on massive user data)
Index 1 2 3 4 5 6 7 8 9
Input u_i x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8
Transformer (few-shot learning on a few user data)
Index 1 2 3 4 5 6 7 8 9
Input u_k x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8

Figure 1: An overview of our approach. Top box represents a Transformer model, whose parameters (in grey) come from a pretrained model like RoBERTa. Middle box shows a general user-aware model trained on massive user data. User-specific vector (in green) works as a virtual token and Transformer parameters are fine-tuned (in blue). Bottom box shows the few-shot learning stage. Transformer parameters (in blue) are fixed and only user-specific vector (in red) is learned.

3 Methodology

In this section, we propose using UserAdapter as an alternative to fine-tuning all the parameters of the Transformer-based pretrained model when new users involved. UserAdapter learns a lightweight user-specific vector, while the dominant parameters of the Transformer are fixed during the few-shot learning. An overview of our approach is shown in Figure 1. We first train a general user-aware model with massive data based on a pre-trained Transformer. Afterwards, in the few-shot learning stage, the parameters of the Transformers are fixed and only parameters of tiny user-specific vector are learned and stored for each new user. Details of our approach are introduced as follows.

3.1 Model

Specifically, UserAdapter adds a trainable user-specific vector $u_\theta \in R^d$ for each user, where d denotes its dimension. For each input x, we prepend a trainable user-specific vector u_θ to the input embeddings $E = Embeddings(x)$, which is taken as the input of a Transformer-based encoder. Then we produce the last hidden vector H of the user-aware sequential vectors:

$$H = Transformer_\phi([u_\theta; E]) \quad (1)$$

where $[;]$ denotes concatenation. The final hidden vector H is taken for classification:

$$p(x) = classifier_{\phi}(H) \tag{2}$$

where classifier is two linear layers followed by a softmax layer and $p(x)$ is the predicted score for classes. The parameters ϕ include the parameters of the Transformer and the classifier. During the few-shot learning stage, dominant parameters ϕ are fixed and only user-specific parameters θ are learned.

3.2 Parametrization

In order to enhance the expressive ability of the user-specific vector and make the optimization more stable, we follow Li and Liang (2021) and employ parametrization strategy. Specifically, we reparametrize the user prefix vector $u_{\theta} = MLP_{\theta}(u'_{\theta})$ with an MLP layer MLP_{θ} for each user. The parameters of the MLP layer are user-specific. The dimension of parametrization vector u'_{θ} is noted as k, which can vary in practice. The impact of changing different variants of k is analyzed in § 4.4.

3.3 Learning

In the few-shot learning stage, the parameters ϕ of the Transformer and the classifier are fixed, and only the user-specific parameters θ are trainable. The objective follows cross-entropy objective:

$$\theta = \arg\min_{\theta} - \sum_{i=1}^{N} y_i \log p_i(x) \tag{3}$$

where N is the number of classes. y_i is the expected label and $p_i(x)$ is the predicted score.

4 Experiments

4.1 Experiment Setup

Datasets and Evaluation Metrics We evaluate our approach on the IMDB and Yelp datasets, detailed in § 2. The overall evaluation metrics is multi-class label accuracy on the test set. As mentioned above, the datasets have two parts: part A and part B. Specifically, we split the data of each user in each part with the ratio: $(train/val./test) = (0.8/0.1/0.1)$. Therefore, the train, validation and test sets in each part have the same number of users. For each user in part B, we only use the few data from the same user for few-shot learning. Numbers of instances are shown in Table 2.

Dataset	Part	Train	Val.	Test
IMDB	A	101,778	12,715	13,133
	B	7,975	990	1,119
Yelp	A	298,713	37,328	38,988
	B	42,184	5,249	5,907

Table 2: Numbers of instances in the datasets.

Few-shot Learning Strategy We adopt a few-shot learning strategy. We first train a general user-aware UserAdapter model with massive user data in part A. Then, we employ part B for few-shot learning. For each new user in the part B, we fix the parameters ϕ of the Transformer and the classifier, and only train and store the user-specific parameters θ for that user. Then, we independently test the user-specific model for each user in test B. The overall accuracy on test B is the average of the accuracy on the test sets of all users.

Model Parameters We employ RoBERTa$_{\text{Base}}$ (Liu et al., 2019) as the backbone model, which has nearly 125M parameters. If we consider the parametrization strategy and set $k = 768$, the number of user-specific parameters is less than 0.5% of the total parameters. We use AdamW as the optimizer. When training the general model, the learning rate is 1e-5 and the batch size is 6. In the few-shot learning stage, the learning rate is 1e-4 and the batch size is 12.

4.2 Models

We evaluate following methods on the two datasets:

- RoBERTa (w/o ft): The RoBERTa trained on part A and directly tested on test sets without further fine-tuning on part B.
- RoBERTa (ft): The RoBERTa trained on part A and completely fine-tuned by the data of each user in part B.
- RoBERTa (few-shot): The RoBERTa first trained on part A and only the parameters of the classifier is tuned by the data of each user in part B under the few-shot learning setting.
- UserAdapter (retrieve w/o ft): The User-Adapter trained on part A. When being tested on new user data in test B, it retrieves the user in part A that wrote the most similar reviews and adopt its user-specific vector. The similarity of two users are measured by the cosine distance of the tfidf vectors of their reviews.

Models	IMDB-A	Yelp-A
RoBERTa (w/o ft)	40.98%	66.69%
UserAdapter (retrieve w/o ft)	46.79%	70.39%
	IMDB-B	Yelp-B
RoBERTa (w/o ft)	38.82%	64.82%
UserAdapter (retrieve w/o ft)	39.80%	66.75%
RoBERTa (ft)	44.89%	68.25%
RoBERTa (few-shot)	40.18%	65.78%
UserAdapter	46.15%	70.22%

Table 3: Accuracy on the test A and test B of Yelp and IMDB datasets. UserAdapter is our approach under the few-shot learning setting.

- UserAdapter: The UserAdapter trained on part A, then trained on the data of each user in part B under the few-shot learning setting.

4.3 Few-shot Learning Evaluation

We test the models on the test set of part A (test A) and test set of part B (test B) and report the results in Table 3. The results on test A (the first and the second row) show the performance of fine-tuned RoBERTa and the general user-aware UserAdapter model. We can see that the general UserAdapter model outperforms RoBERTa by a large margin. This observation indicates that modeling personalized user-specific information is essential for the sentiment analysis task as it captures the style and preference of the reviews of different users.

Then, to evaluate the performance of few-shot learning, we test the models on each user in test B. The results on the third row show that directly adapting the models (i.e., RoBERTa (w/o ft)) to unseen users drops the performance. Further results (the fifth row) show that completely fine-tuning RoBERTa on the data of each new user can improve the performance. However, this approach requires heavy optimization and storage. Furthermore, we can see that by only optimizing a user-specific vector with few data of each new user, UserAdapter (the last row) outperforms fine-tuned RoBERTa and RoBERTa with few-shot learning (the sixth row), and achieves comparable performance with the general user-aware model (the second row). These phenomena show that our UserAdapter approach can alleviate the burden of heavy model fine-tuning and storage by only tuning and storing tiny parameters.

4.4 Parametrization Evaluation

In this part, we evaluate the impact of changing different variants of dimension k of the parametrization vector to find a better trade-off between the performance and the user-specific parameter size.

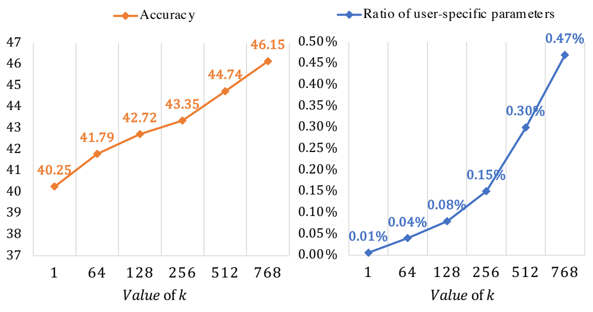

Figure 2: Performance on test B (left) and the ratio of user-specific parameters θ (right) versus the value of k, which is the dimension of parametrization vector. Performance improves with increased value of k.

We choose six values of dimension k from 1 to 768 and test the performance of UserAdapter on the test B of the IMDB dataset. Figure 2 shows the variation of the performance and the ratio of user-specific parameters with different values of k. $k = 1$ indicates that we don't adopt parametrization strategy. Our finding is consistent with Li and Liang (2021) that without utilizing parametrization strategy, the learning process is less stable and the final performance drops. Moreover, we can see that the performance improves as the dimension k increases, which also leads to the increasing size of the user-specific parameters. Therefore, we can find a balance between the performance and the user-specific parameter size in practice. In our experiments, we adopt $k = 768$.

5 Conclusion

In this paper, we introduce a lightweight few-shot learning approach, dubbed UserAdapter, which clamps dominant parameters of the Transformer and only optimizes and stores a tiny user-specific vector for each new user. UserAdapter prepends a trainable user-specific vector to the input of the Transformer. We first train a general user-aware UserAdapter model with massive user data. Then, we fix the dominant parameters of the Transformer and only optimize and store the user-specific vector for each new user. We take sentiment analysis as a test bed and create two datasets for few-shot personalized sentiment analysis. Experiments on the two datasets show that modeling user-specific information empowers our approach to outperform fine-tuned RoBERTa significantly. More importantly, results show that our approach achieves comparable results when being adopted to new users with only tuning less than 0.5% of all the parameters.

Acknowledgement

Wanjun Zhong, Jiahai Wang and Jian Yin are supported by the National Natural Science Foundation of China (U1711262, U1711261, U1811264, U1811261, U1911203 ,U2001211), Guangdong Basic and Applied Basic Research Foundation (2019B1515130001), Key R&D Program of Guangdong Province (2018B010107005). The corresponding author is Jian Yin.

References

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Qiming Diao, Minghui Qiu, Chao-Yuan Wu, Alexander J Smola, Jing Jiang, and Chong Wang. 2014. Jointly modeling aspects, ratings and sentiments for movie recommendation (jmars). In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 193–202.

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.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

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*.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1422–1432.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Cuihong Cao, Daxin Jiang, Ming Zhou, et al. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *arXiv preprint arXiv:2002.01808*.