UserIdentifier: Implicit User Representations for Simple and Effective Personalized Sentiment Analysis

Fatemehsadat Mireshghallah[1]*, **Vaishnavi Shrivastava**[2], **Milad Shokouhi**[2],
Taylor Berg-Kirkpatrick[1], **Robert Sim**[3], **Dimitrios Dimitriadis**[3]
[1] University of California San Diego, [2] Microsoft Corporation, [3] Microsoft Research
[fatemeh, tberg]@ucsd.edu,
[vashri,milads,rsim,didimit]@microsoft.com

Abstract

Global models are typically trained to be as generalizable as possible. Invariance to the specific user is considered desirable since models are shared across multitudes of users. However, these models are often unable to produce personalized responses for individual users, based on their data. Contrary to widely-used personalization techniques based on few-shot and meta-learning, we propose UserIdentifier, a novel scheme for training a single shared model for all users. Our approach produces personalized responses by prepending a fixed, user-specific non-trainable string (called "user identifier") to each user's input text. Unlike prior work, this method doesn't need any additional model parameters, any extra rounds of personal few-shot learning, or any change made to the vocabulary. We empirically study different types of user identifiers (numeric, alphanumeric, and also randomly generated) and demonstrate that, surprisingly, randomly generated user identifiers outperform the prefix-tuning based state-of-the-art approach by up to 13%, on a suite of sentiment analysis datasets.

1 Introduction

Personalization arises in applications where different clients need models specifically customized to their environment and user profiles (Yang and Eisenstein, 2017; Mazaré et al., 2018; Flek, 2020). This need for customization stems from the inherent heterogeneity existing in the data and the labels, especially when the task is classification (Kulkarni et al., 2020; Wang et al., 2018). Fig. 1 shows an example of the sentence "That is just great!". This sentence could carry a positive sentiment, a neutral apathetic sentiment, or even a completely negative sentiment. A non-personalized model cannot correctly predict the label for different users.

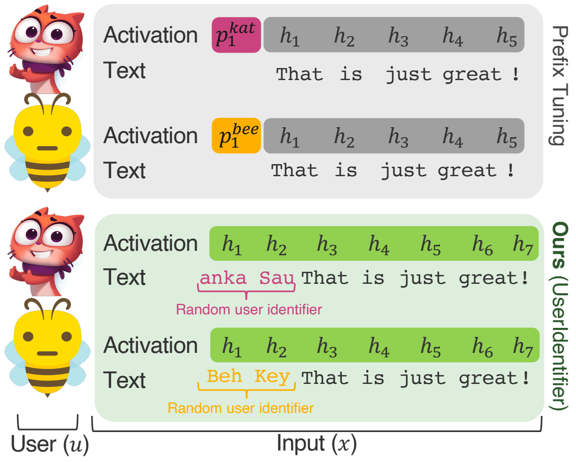


Figure 1: An overview of the proposed method, UserIdentifier, compared to its prefix-tuning counterpart. p_1^{kat}, p_1^{bee} denote the trainable prefix vector for users kat and bee, in the prefix tuning method (Zhong et al., 2021). UserIdentifier, on the other hand, does not have trainable user-specific parameters and uses random per-user (UID) strings ("anka Sau" and "Beh KY"), to condition a shared model, for each user.

Most techniques for personalization generally involve two phases: first, a shared, global model is built between all users, and then, it is personalized for each client using their data (Kulkarni et al., 2020; Schneider and Vlachos, 2019; Lee et al., 2021). In such cases, each user has either an entirely separate model, or additional personal parameters, causing significant overheads, both in terms of storage of the large models, and the computation complexity of training separate models for each user. UserAdapter (Zhong et al., 2021), the state-of-the-art in personalized sentiment analysis, takes a prefix-tuning based approach (Li and Liang, 2021), as shown in Fig. 1. In the first phase, a global model is trained in a user-agnostic way on a large dataset. In the second phase, each user u is assigned their own prefix vector, p_1^u, which is fine-tuned separately for them, on their own data. If there are N users, there would be N separate rounds of fine-tuning, producing N vectors. During this prefix-tuning phase, the underlying transformer-based classification model is frozen and shared between users,

[*] Work done as part of an MSR internship. Corresponding author email: fatemeh@ucsd.edu

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 3449 - 3456
July 10-15, 2022 ©2022 Association for Computational Linguistics

and the final N vectors are stored for inference.

To alleviate these training and storage costs and also improve overall performance, we propose training a single, shared personalized model, which can capture user-specific knowledge by conditioning on a unique, user-specific sequence of tokens from the classifier's vocabulary. We name this sequence "user identifier", and dub the underlying method of adding user identifiers to the input UserIdentifier. This is shown in Fig. 1, where we add the randomly generated, and non-trainable user identifiers "`anka Sau`" and "`Beh KY`" to each user's sample, and then train the transformer classifier model, on these augmented samples. The user identifiers just use the underlying model's vocabulary and embeddings and do not add any tokens nor any user embeddings to the model. They are also static over time, and unique to each user, which means the user "bee" in Fig. 1 will have "`Beh KY`" pre-pended to all their samples, and no other user has this identifier. This is similar to the prompting of models like GPT-3 (Brown et al., 2020), however, here the prompt is fixed and used as data augmentation during training, and the model is not generative. As such, we only do training once and have one set of shared parameters for all users. The approach is similar in essence to those of Daumé III (2009); Kocoń et al. (2021); Kocoń et al. (2021), which augments each individual feature with domain annotations.

We experiment with different types of strings for user identifiers, such as real usernames from the dataset, consecutive numbers, random digits, random non-alphanumeric tokens, and random tokens (all types), and observe that, surprisingly, random identifiers, sampled from all possible tokens in the vocabulary perform best, providing $1.5\%-13\%$ classification accuracy improvement on average, over the prefix-tuning based method UserAdapter (Zhong et al., 2021). We also study different lengths of identifiers. We report our results on three different sentiment analysis datasets (Sentiment 140, IMDB, and Yelp). We also show that UserIdentifier is effective in a federated learning setup (Appendix A.1), which is a real-world application of such personalization (Kulkarni et al., 2020).

2 UserIdentifier

In this section, we first explain how UserIdentifier operates, then we go over the parameterization and learning procedure.

2.1 Method

UserIdentifier is a data augmentation method which consists of adding a sequence of user-specific tokens (user identifier, u_{id}, drawn from the tokenizer's vocabulary) to each sample, x, to provide user-related cues to the model and help it learn individual user behaviour and preferences, all in one shared model. Figure 1 shows how this augmentation works. Each utterance is appended by the user identifier to create the augmented sample $[u_{id};x]$, and then used as input to the model, for the training stage.

There is no restriction on what the make-up or the length of the user identifier sequence can be (as long as it is not longer than the maximum sequence length the model can input). However, we propose randomly generating each user's identifying sequence, through uniformly sampling from the tokenizer vocabulary, for a given length L, which we ablate in section 4.2. This random sampling step creates a diverse while unique set of user identifiers, potentially allowing the model to distinguish different users more efficiently.

2.2 Parameterization

For parameterizations of the user identifiers, we use parameter tying (He et al., 2019), where the user identifiers use the same set of parameters for their embeddings as the rest of the user utterance. In other words, in this setup the user embedding parameters are tied to the embedding parameters of the main transformer classification model, parameterized by θ. This form of parameterization is both simpler and has highere performance (we try separate parametrization in our experiments and show its inferior performance).

2.3 Learning

The training stage doesn't change compared to the original fine-tuning process, with parameters θ of the transformer model being trained to minimize the cross-entropy loss for the classification (Devlin et al., 2018):

$$\mathcal{L}_{\text{CE}}(x,u_{id},y;\theta) = -\log \Pr(y|[u_{id};x];\theta) \quad (1)$$

$$\theta = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}_{\text{CE}}(x,u,y;\theta) \quad (2)$$

Where x denotes the input utterance, u_id denotes the user identifier for the user to whom utterance x belongs, and y is the class label for x.

Table 1: Dataset specifications

Dataset	# Users	# Samples	# Classes
IMDB	1,012	137,710	10
Yelp	4,460	428,369	5
Sent140	1,100	56,557	2
Sent140 (skewed)	473	23,155	2

3 Experimental Setup

3.1 Tasks, Datasets, and Models

We evaluate the proposed method on the task of sentiment analysis. Table 1 shows a summary of the datasets used in our experiments. We use the IMDB (Diao et al., 2014) and Yelp (Tang et al., 2015) datasets for comparison with the UserAdapter method (Zhong et al., 2021) and for the ablation studies. Each user's data is split into train, test, and validation sets, with 0.8, 0.1, and 0.1 ratios. For comparison purposes, we are using a subset of the available users, i.e. those with fewer than 50 samples, as done by Zhong et al. in support of few-shot learning, for reporting test accuracy. We use the RoBERTa-base model for this set of experiments.

In addition to IMDB and Yelp, we also report the performance of the proposed method on the Sentiment140 dataset citesent140, caldas2018leaf, which is a set of Tweets collected from Twitter and labeled positive or negative based on the emojis in each Tweet. For this dataset, We use the methodology provided by Li et al. (2019) to preprocess and partition this dataset. We create a second version of this dataset, and mark it as "skewed". For this skewed data, the users have been selected such that their sentiments are mostly skewed, i.e. we only include users with 80% or more positive or negative Tweets. We do this to create a setup where data is more heterogeneously distributed. We use BERT-base-uncased for evaluations on the Sentiment140 dataset.

3.2 Baselines

Conventional Training. Conventional finetuning of the pre-trained transformer model on the full dataset, without personalization.

UserAdapter. In UserAdapter, the work closest to ours, a per-user embedding is learned through few-shot learning and stored. These personal vectors are prepended to the users' data to create personal responses. This work proposes prefix-tuning (Li and Liang, 2021) on a user-level. Unlike our method, UserAdapter consists of two phases, as discussed in the introduction.

Trainable User Embeddings. UserIdentifier uses the same set of parameters (BERT embeddings) for embedding both the sample content and the user identifiers. In other words, the text and user embedding parameters are tied. To untie these parameters, we introduce a third baseline, with trainable user embeddings. In this setup, while the tokens used for the user identifier are still drawn from the pre-trained model's tokenizer vocabulary, we're creating and training a separate set of global parameters for the user embedding, instead of using the pre-trained model's embedding. These extra embedding parameters are placed in parallel to the model's existing embedding layer. Each input sequence is partitioned to the content and the UID, the content is fed to the model's existing embedding layer and the UID is fed to the new embedding.

3.3 Types of User Identifiers

We investigate five scenarios (types of sequences) for the user identifiers. The length of the user identifier sequences can vary in terms of the number of tokens (L) for the last three of these scenarios.

Default (Def.): This scenario uses the real user id (e.g., username) of that user, when provided by the dataset and if they are not private. We only have this option available for the Sentiment140 dataset.

Consecutive Numbers (Num.): We assign each user a unique number, from 1 to N, representing each user (up to N users).

Random sequence of digits (Rand. Dig.): In this scenario, L independent and identically distributed (i.i.d) samples from the set of digits (0 to 9) are drawn, creating a sequence of length L for each user.

Random sequence of tokens with non-alphanumeric characters (Rand. Non.): L i.i.d samples are drawn from a subset of tokens (with size 400) that contain non-alphanumeric characters, e.g., the token `Ã""`. The motivation for this scenario is that such user identifiers might be easier for the model to distinguish from the text (if we make sure the textual content in the sample has no overlapping tokens with the identifier).

Random sequence of all tokens (Rand. All): This scenario draws L i.i.d samples from the set of all available tokens in the tokenizer vocabulary.

4 Results

Apart from the evaluations here, We have also provided evaluations of applying our method to federated learning in Appendix A.1, and applying it to new unseen user samples in 4.4.

Table 2: Comparison of sentiment classification accuracy of UserIdentifier, with the baselines of Section 3.2. Num., Def. and Rand. refer to the different types of user identifiers introduced in Section 3.3.

	Dataset	Conventional	UserAdapter	Trainable User Emb.			UserIdentifier		
				Num.	Def.	Rand. All	Num.	Def.	Rand. All
RoBERTa	IMDB	45.1	46.2	45.5	–	48.9	50.1	–	**52.5**
	Yelp	68.3	70.2	68.3	–	70.6	69.5	–	**71.3**
BERT	Sent140	84.7	–	84.7	86.3	86.5	84.9	87.1	**87.1**
	Sent140 (Skewed)	86.3	–	87.2	89.3	90.0	87.5	90.3	**90.4**

Table 3: Classification accuracy vs the length (#tokens) and type (Section 3.3) of user identifier sequence)

	Seq. Len.	Rand. Dig	Rand. Non.	Rand. All
IMDB	5	48.8	51.3	52.2
	10	47.4	51.7	52.5
	20	47.1	50.2	51.1
	50	46.5	48.7	50.8
	200	33.3	32.8	40.1
Yelp	5	68.6	69.3	70.8
	10	68.7	69.6	71.3
	20	68.4	68.6	71.0
	50	67.8	69.0	70.6
	200	63.2	60.2	65.1

4.1 Comparison with Baselines

A comparison of UserIdentifier with the state-of-the-art UserAdapter method, and the other baselines is presented in Table 2. For the **Num.** (consecutive numbers) and **Def.** (default username) scenarios, as detailed in Section 4.2, the length of the user identifier sequences depends solely on the tokenization process. For the case of **Rand. All** (randomly sampled from all vocabulary tokens), however, it is shown that the sequence length of 10 tokens provides the best performance through the ablation study, therefore the results are reported for this length. Since the default usernames for IMDB and Yelp datasets are not provided, the corresponding results are not reported here.

It is shown that UserIdentifier with randomly generated identifiers outperforms all baselines, in all tasks. Our intuition is that UserIdentifier outperforms UserAdapter because of collaborative learning and personalization happening simultaneously, unlike in the case of UserAdapter where personalization is performed separately for each user. The performance of trainable user embeddings appears inferior to that of UserIdentifier, which could be attributed to the parameter tying used in UserIdentifier. This parameter tying couples the learning problems for both domains (user identifier and text) and allows us to jointly learn from the full data, as in (He et al., 2019). For the Sentiment140 dataset, we can see that increasing the heterogeneity or skew in the dataset boosts the benefits brought about by UserIdentifier. This shows that the proposed method performs better in setups where personalization is actually needed (Deng et al., 2020).

4.2 Ablation Studies

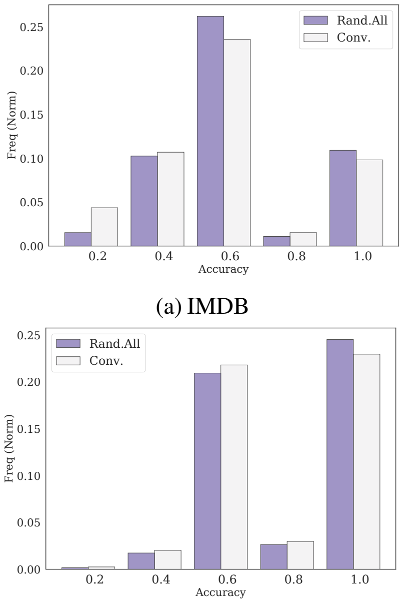
Table 3 shows our ablation study into the length and the type of the user identifier sequence, for IMDB and Yelp datasets. The most evident trend is that performance significantly degrades in both datasets when the length of the user identifier sequence exceeds 20 tokens, holding for all identifier types. This is because the length of the input text itself is essentially decreased (the maximum sequence length for RoBERTa is 512, and the textual content of the sample is truncated to fit the user identifier) when increasing the length of the identifier. This decreases the useful information which could be used to infer sentiment, and in turn, it has an adverse effect on accuracy.

A rather surprising observation is that randomly sampling from the tokenizer's entire vocabulary outperforms sampling only from digits or from the non-alphanumeric tokens. This can be attributed to the different sizes of the sampling spaces for these three types, and the probability of overlap in user identifier from user to user. For the random digits (**Rand. Dig.**) the sample space size for each token position is 10, the number of possible digits. For the non-alphanumeric tokens, we have limited them to 400, and for the token type all (**Rand. All**), the possible sample space is 47,400. This means that the probability of having token overlaps in user identifiers is much much smaller in the last scheme than it is for the other two, or in other words, the hamming distance between different user identifiers is higher with this method.

One hypothesis that might explain the success of random user identifiers: random user identifiers are similar to random feature projections (Rahimi et al., 2007), but, in contrast with learnable embeddings, they are defined in terms of the pre-trained model's original token embeddings. This may have a positive effect on optimization during fine-tuning.

4.3 User-level Study Accuracy

Figure 2 shows the distribution of test-accuracy changes across users, for conventional training (Conv.) and the Rand. All scheme from UserIdentifier. We have chosen the best version of our model from Table 2 for this figure. We can see that the number of users with low accuracy decreases in both datasets. Also, the standard deviation of accuracy across users decreases compared to conventional training when using UserIdentifier, it drops from 27.0% to 25.6% for IMDB, and from 21.2% to 21.0% for Yelp. We provide more plots and analysis on this in A.2.



(a) IMDB

(b) Yelp

Figure 2: Distribution of test accuracy across users.

4.4 Performance on Unseen Users

To measure how robust the proposed method is to new users that have never been seen before, we run an evaluation on new users and report the results in Table 4. For this experiment, we have used the best models from Tables 2, and tested them on samples from new users, without appending any user identifiers. It is noteworthy that there is some distribution shift between these unseen users and the seen users from Table 2, especially for Yelp, as we used samples that were not used in the original training/test/val setup (this test set contains 5000 samples for Yelp and 1357 samples for IMDB).

Table 4: Evaluation results on unseen users.

	UserIdentifierAccuracy (%)	Conventional Model Accuracy (%)
IMDB	50.4	50.9
Yelp	50.1	49.8

The UserIdentifier column refers to the accuracy of those datapoints on models trained with user identifiers, and the conventional column shows the accuracy but on a conventionally trained model, which would be the baseline. We can see that both models behave similarly, which suggests that for unseen data points, the UserIdentifier trained model falls back to a conventional model, and does not behave even worse.

5 Conclusion

In this work, we present a novel approach for learning global models, producing personalized classification responses. This method which doesn't require model extensions or specialized training algorithms, consists of appending a fixed, non-trainable, unique identifier string to each sample during training and inference.

Acknowledgments

The authors would like to thank the anonymous reviewers and meta-reviewers for their helpful feedback. We also thank Huseyin Inan and Guoqing Zheng for insightful discussions and Wanjun Zhong for helping with datasets. Additionally, we thank our colleagues at the UCSD and Microsoft for their helpful comments and feedback.

Ethical Considerations

Our proposed model is intended to be used for addressing the problem of personalization, by learning one shared model for all users, and querying it using a personal identifier. One potential measure that needs to be taken for deployment of such technology is to setup proper authentication tools, so that each user can only query with their own identifier and prevent users from breaching privacy by querying other users' models. However, this could be a concern in other personalization setups too.

The datasets used in our experiments are all publicly available (Yelp, IMDB and Sentiment 140), and we have not collected any information about the users who have contributed their data beyond what is originally provided in the dataset, which is only the user-based partitioning of the data.

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*.

Hal Daumé III. 2009. Frustratingly easy domain adaptation. *arXiv preprint arXiv:0907.1815*.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. 2020. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*.

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*, KDD '14, page 193–202, New York, NY, USA. Association for Computing Machinery.

Lucie Flek. 2020. Returning the N to NLP: Towards contextually personalized classification models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7828–7838, Online. Association for Computational Linguistics.

Junxian He, Xinyi Wang, Graham Neubig, and Taylor Berg-Kirkpatrick. 2019. A probabilistic formulation of unsupervised text style transfer. In *International Conference on Learning Representations*.

Jan Kocoń, Marcin Gruza, Julita Bielaniewicz, Damian Grimling, Kamil Kanclerz, Piotr Miłkowski, and Przemysław Kazienko. 2021. Learning personal human biases and representations for subjective tasks in natural language processing. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 1168–1173. IEEE.

Jan Kocoń, Alicja Figas, Marcin Gruza, Daria Puchalska, Tomasz Kajdanowicz, and Przemysław Kazienko. 2021. Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. *Information Processing & Management*, 58(5):102643.

Jakub Konečnỳ, H Brendan McMahan, X Yu Felix, Ananda Theertha Suresh, Dave Bacon, and Peter Richtárik. 2018. Federated learning: Strategies for improving communication efficiency.

Viraj Kulkarni, Milind Kulkarni, and Aniruddha Pant. 2020. Survey of personalization techniques for federated learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, pages 794–797. IEEE.

Hung-yi Lee, Ngoc Thang Vu, and Shang-Wen Li. 2021. Meta learning and its applications to natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Tutorial Abstracts*, pages 15–20, Online. Association for Computational Linguistics.

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. 2019. Fair resource allocation in federated learning. In *International Conference on Learning Representations*.

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, Online. Association for Computational Linguistics.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. 2020. Three approaches for personalization with applications to federated learning.

Pierre-Emmanuel Mazaré, Samuel Humeau, Martin Raison, and Antoine Bordes. 2018. Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779, Brussels, Belgium. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh. 2020. Privacy in deep learning: A survey. In *ArXiv*, volume abs/2004.12254.

Ali Rahimi, Benjamin Recht, et al. 2007. Random features for large-scale kernel machines. In *NIPS*, volume 3, page 5. Citeseer.

Johannes Schneider and M. Vlachos. 2019. Mass personalization of deep learning. *ArXiv*, abs/1909.02803.

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, Lisbon, Portugal. Association for Computational Linguistics.

Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. 2021. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*.

Weichao Wang, Shi Feng, Wei Gao, Daling Wang, and Yifei Zhang. 2018. Personalized microblog sentiment classification via adversarial cross-lingual multi-task learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Yi Yang and Jacob Eisenstein. 2017. Overcoming language variation in sentiment analysis with social attention. *Transactions of the Association for Computational Linguistics*, 5:295–307.

Wanjun Zhong, Duyu Tang, Jiahai Wang, Jian Yin, and Nan Duan. 2021. UserAdapter: Few-shot user learning in sentiment analysis. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1484–1488, Online. Association for Computational Linguistics.

A Appendix

A.1 Federated Learning as an Application

Federated learning is a form of distributed learning where data never leaves each user's device (Wang et al., 2021; Konečnỳ et al., 2018; Mireshghallah et al., 2020). Instead, the user trains a model on their device locally and then shares the gradients (model updates) with a centralized server, which aggregates the gradients from different users and sends the updated model back to all of them, for further training. We target this setup since it is a good candidate for personalization, given how a conventionally trained global model often fails to accommodate all users (Kulkarni et al., 2020; Mansour et al., 2020). Table 5 shows the performance gain of applying UserIdentifier, in a federated setup. UserIdentifier can be readily applied in federated learning, by assigning identifiers to each user and then asking them to append it to all their samples. We have used the Rand. All type of user identifier for this experiment, since we observed in previous sections that it was the most effective. In general, the baseline performance and the performance gain in the federated setup is slightly lower than in centralized learning, which is due to the distributed nature of FL, and the fact that only the average of multiple gradient updates are shared with the server for aggregation.

Table 5: Performance of UserIdentifier for sentiment classification in a federated learning setup.

	Dataset	Conventional	User Identifier
RoBERTa	IMDB	44.30	47.23
	Yelp	68.40	70.60
BERT	Sent140	84.40	86.30
	Sent140 (Skewed)	86.50	90.00

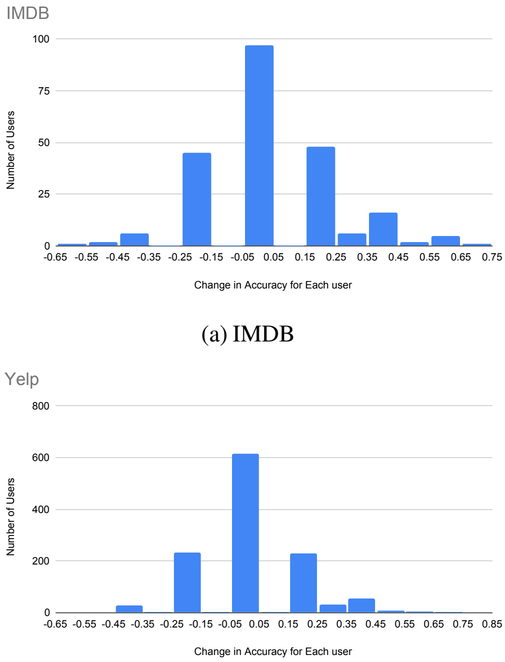
A.2 Further User-level Accuracy Studies

Figure 3 shows the change in user accuracy, when we use UserIdentifier for training, instead of conventional training for each user. In other words, the horizontal axis shows $conventional_{acc} - UID_{acc}$ for each user, and the vertical axis shows the count of users.

As the plots show, on average across the two datasets, 32.1% of the users see improvements in accuracy, whereas 54.2% don't see any change.

A.3 Maximally Distant User Identifiers

To better understand the effect of edit distance between user identifiers, We also experimented with **maximally distanced** identifiers (for the Rand. All setup), where the maximum distance would be the length of the identifier here since each token in the identifier can take a substantially large number of values. For this experiment, we used rejection sampling for user ids, as in if a new random sample had any token overlaps with existing user ids, we would reject it and sample a new one. We observed results very similar to the ones with the random identifiers, which we hypothesize is because the random identifiers are already highly distanced and rarely overlap (less than 10% of the users have non-maximal distance).



(a) IMDB

(b) Yelp

Figure 3: Distribution of test accuracy **change** across users.