# Now It Sounds Like You: Learning Personalized Vocabulary On Device

**Ashish Shenoy*, Sid Wang*, Pierce Chuang, John Nguyen**

Meta

yuwang2020@meta.com, ashishvs@meta.com, pichuang@meta.com, ngjhn@meta.com

## Abstract

In recent years, Federated Learning (FL) has shown significant advancements in its ability to perform various natural language processing (NLP) tasks. This work focuses on applying personalized FL for on-device language modeling. Due to limitations of memory and latency, these models cannot support the complexity of sub-word tokenization or beam search decoding, resulting in the decision to deploy a closed-vocabulary language model. However, closed-vocabulary models are unable to handle out-of-vocabulary (OOV) words belonging to specific users. To address this issue, We propose a novel technique called "OOV expansion" that improves OOV coverage and increases model accuracy while minimizing the impact on memory and latency. This method introduces a personalized ÖOV adapter̈hat effectively transfers knowledge from a central model and learns word embedding for personalized vocabulary. OOV expansion significantly outperforms standard FL personalization methods on a set of common FL benchmarks.

## Introduction

Federated learning (FL) is a distributed machine learning paradigm that enables model training on decentralized datasets without exchanging or sharing sensitive data (McMahan et al. 2016). Personalized FL considers models unique to each client under heterogeneous data settings (Hanzely and Richtárik 2020). During personalization, the server sends a global model to the clients for local fine-tuning then the client uses that model for inference. Our work studies a class of on-device language models for next-word prediction task trained with personalized FL. The resource-constrained edge devices (Chen et al. 2019c; Qiu et al. 2022; Mathur et al. 2021; Yousefpour et al. 2023) limits the usage of subword-level tokenizers. On the one hand, subword tokenizers with large vocabulary require a large memory footprint, making deployment infeasible. On the other hand, a smaller vocabulary leads to longer tokenized sequences and increases generation latency. In order to satisfy these constraints, a *closed vocabulary* (Qin and Rudnicky 2013), a word-level vocabulary with a white-space tokenizer that treats all unknown words as a special token, must be used. Consequently, the model cannot handle out-of-vocabulary (OOV) words (Chen et al. 2019a), making it more challenging to understand the communication style of an individual user. Previous methods have demonstrated effectiveness of user specific adaptation using domain embeddings (Shenoy et al. 2021) or prompt tuning (Dingliwal et al. 2021). But the heavy tail of OOV (shown in Figure 4) motivates the need for personalized vocabulary which they do not address. In this work, we propose *OOV Expansion* to personalize on-device vocabulary, tailoring the model toward users' unique wording habits and spelling patterns. *OOV Expansion* uses an adapter – an MLP with residual connections inserted to different submodules to help the model adapt to new knowledge domain (Houlsby et al. 2019) – to compute the embedding output of the OOV words. Our main contributions are as follows :

- We propose *OOV Expansion*, a novel personalized FL method, to extract useful features for user-specific vocabulary and address the long tail OOV issue.
- We perform comprehensive experiments, showing superior results over that of the baselines, across a set of standard FL datasets.
- We demonstrate that our technique significantly outperforms previous methods with up to $5.6\%$ relative improvements on next-word prediction accuracy and reducing the averaged unknown-word-rate by at least $97\%$.

## Related Work

Federated learning (McMahan et al. 2016) has achieved significant impact in the field of natural language processing (NLP) in recent years (Chen et al. 2019a,b; Wu, Liang, and Wang 2020; Lin et al. 2021; Hilmkil et al. 2021; Liu et al. 2021; Ro et al. 2022). Unlike non-federated language models where subword-based tokenizations such as WordPiece (Schuster and Nakajima 2012), Byte Pair Encoding (BPE) (Sennrich, Haddow, and Birch 2015) or SentencePiece (Kudo and Richardson 2018) have become major choices, word-level tokenizer with a closed vocabulary is the default choice for edge device settings due to its low capacity and real-time nature. This gives rise to the commonly known OOV issue (Chen et al. 2019c,b), which has been investigated by several preceding work in federated learning: (Chen et al. 2019a) trains a separate character-level generative model to sample new words from, but by design requires

*These authors contributed equally.

Copyright © 2024, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

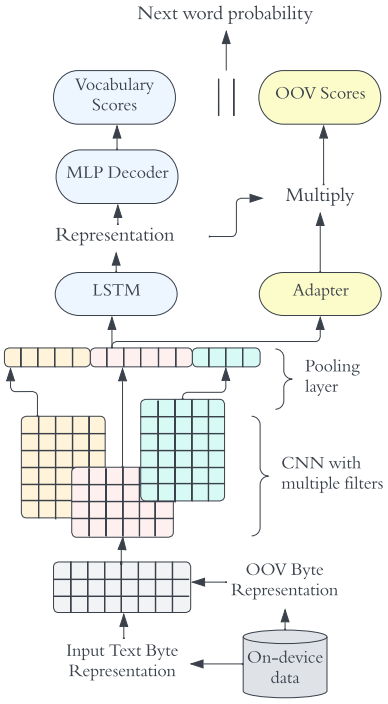



Figure 1: The mechanism flowchart of the OOV Expansion stage; here $\|$ denotes vector concatenation.

an extra training stage; (Singhal et al. 2021) introduces a new but more complex FL algorithm named FedRecon based on partial personalization; (Bagdasaryan et al. 2022) proposes to iteratively update a subword-level tokenizer using the token sequences sampled from an FL-trained language model. Whereas (Khandelwal et al. 2019) and (Bekal et al. 2021) use memorization to dynamically update the vocabulary without retraining. However, these approaches either focuses on a different problem, or is not suitable for our settings due to their requirements of high training budgets, system complexity, and latency/memory costs. Instead, our approach is based on FL personalization, a technique that has driven a large body of work to address the presumed presence of heterogeneity (Wang et al. 2019; Hanzely and Richtárik 2020; Yu, Bagdasaryan, and Shmatikov 2020; Fallah, Mokhtari, and Ozdaglar 2020; Dinh, Tran, and Nguyen 2020; Li et al. 2020; Singhal et al. 2021; Pillutla et al. 2022; Kulkarni, Kulkarni, and Pant 2020). Another key ingredient of our work is adapter (Houlsby et al. 2019; Stickland and Murray 2019), which has been widely studied in a large volume of work in the non-federated setting (Mahabadi, Henderson, and Ruder 2021; Hu et al. 2021; Li and Liang 2021; He et al. 2021; Pfeiffer et al. 2020).

# Method

## Character-Aware Language Model

Due to memory and latency constraints on on-device modeling, we opt for a character-aware LSTM-based language model instead of transformers. This architecture is commonly used in on-device applications and consists of three components: (1) a character-level CNN (CharCNN) that computes word embeddings, (2) an LSTM encoder that provides input representations, and (3) an MLP decoder that outputs vocabulary scores. The CharCNN uses a UTF-8 embedding of shape $\mathbb{R}^{256\times E}$, where $E$ is the embedding dimension and a CNN with $D$ channels and kernel size $K$. The LSTM encoder has $M$ layers with both input and output dimensions equal to $D$, while the decoder is a linear layer with input dimension $D$ and output dimension $|\mathcal{V}|$, where $\mathcal{V}$ is the closed vocabulary.

## Baseline 1: OOV-as-UNK

This is the traditional FL personalization (i.e. involving only the non-yellow stack in Figure 1). To be precise, the model is first pretrained on a general dataset using the standard "next-word prediction" task, and then sent to client devices for federated learning. Lastly, perform personalization without any additional treatments for OOVs rather than replacing them with the special token $[unk]$.

## Baseline 2: OOV-Oracle

The second baseline assumes to have the knowledge of all users' OOV on server. However, because of the tight memory budgets in practice we cannot afford full vocabulary training, we instead expand the OOV-as-UNK vocabulary by the $N$ most frequent OOV words. Upon expansion, the rest stages (including pretraining, FL, and personalization) proceed in exactly the same way as in Baseline 1.

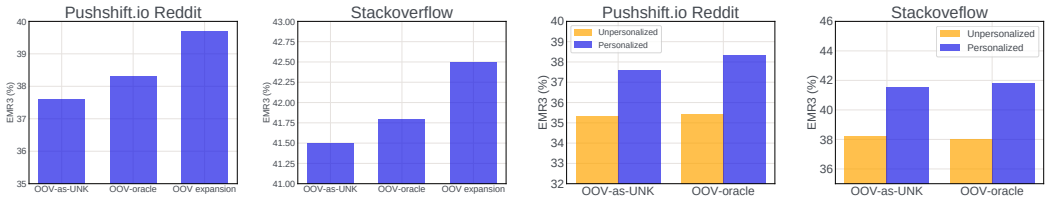
## Our Method: OOV Expansion

We start with the unpersonalized OOV-as-UNK and perform personalization as shown in Figure 1. First, we forward the input sentence using the character-aware language model to get the vocabulary likelihood. Then, we retrieve the client's top $n$ OOV words, compute their representations using CharCNN, and pass them through an adapter (a randomly initialized residual MLP) with hidden dimensions $\vec{\mathbf{H}} = (H_1, H_2, \cdots, H_L)$, where $L$ is the number of layers in the adapter. This gives us "adapted OOV word embeddings". Next, we take the inner product between the adapter outputs and the LSTM encoder outputs to get OOV likelihood. Finally, we concatenate the OOV probabilities with the vocabulary probabilities to obtain the full likelihood over the client personalized vocabulary for next word prediction. The adapter helps separate the two tasks of providing inputs for the LSTM encoder and computing OOV embeddings, while adapting the prior knowledge of CharCNN to the new "OOV task". Our design ensures that no sensitive OOV information leaves the client device, preserving privacy while allowing the model to learn OOV features.

# Experiments

## Model Setup

The model hyperparameters for our proposed model are chosen as $E = 100$, $D = 200$, $K = 4$, $M = 2$. To initialize OOV-as-UNK we extract the 5k most frequent words from a




(a) (b) (c) (d)

Figure 2: (a) and (b) showing $\text{EMR}_3$ of OOV Expansion which are the two baselines on two different datasets. (c) and (d) showing $\text{EMR}_3$ of two baselines before and after personalizations on each dataset.

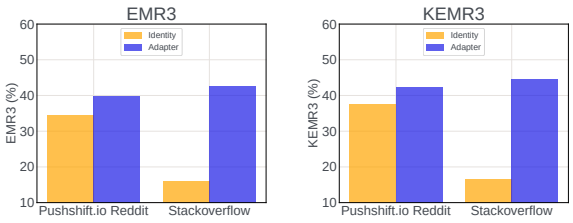



Figure 3: $\text{EMR}_3$ and $\text{KEMR}_3$ of OOV expansion on 2 datasets, with and without adapter

centralized FL dataset to form a closed vocabulary, and pre-train the model on wikitext-103 (Merity et al. 2016) using next-word prediction task for 100 epochs and is evaluated by the exact match metric $\text{EMR}_3$. The OOV-oracle baseline follows the same configuration as OOV-as-UNK, except for having a 10k-sized vocabulary. In other words, OOV-oracle expands the vocabulary of OOV-as-UNK by $N = 5000$ most frequent OOVs. The OOV-as-UNK model has 1.76M (resp. 2.76M) parameters in total. During OOV expansion, we personalize up to 1000 most frequent out-of-vocabulary words from each decentralized dataset.

## Datasets

We train, validate, and test our approach on two public benchmark datasets for federated learning: Reddit and Stack Overflow. The Reddit dataset contains preprocessed comments from the social network, while Stack Overflow consists of questions and answers from the platform. We conduct experiments on the next-word-prediction task using federated learning with natural non-IID partitionings of the datasets. Each dataset is split into training, validation, and test clients, with an $8:1:1$ ratio for personalization, early-stopping/hyperparameter search, and reporting model performance, respectively.

## Evaluation Metric

The model quality is measured by how often the actual next word appears among the top $K$ word predictions, denoted by $\text{EMR}_K$ (Exact Match Rate). More precisely, given a dataset $\mathcal{D}$ of sentences,

$$\text{EMR}_K(\mathcal{D}) = \frac{\sum_{S\in\mathcal{D}}\sum_{w\in S}\mathbb{1}_{\{w\in\text{Top}_K\text{pred}(w)\}}}{\sum_{S\in\mathcal{D}}|S|} \quad (1)$$

This is a natural metric for real-time language model in production, where $K$ next-word suggestions are surfaced when a user is typing. In our experiments we set $K = 3$. The final metric is given below:

$$\text{EMR}_3 = \frac{\sum_{u\in\mathcal{U}}\sum_{S\in\mathcal{D}_u}\sum_{w\in S}\mathbb{1}_{\{w\in\text{Top}_3\text{pred}(w)\}}}{\sum_{u\in\mathcal{U}}\sum_{S\in\mathcal{D}_u}|S|} \quad (2)$$

$$= \frac{\sum_{u\in\mathcal{U}}\sum_{S\in\mathcal{D}_u}|S|\cdot\text{EMR}_3(\mathcal{D}_u)}{\sum_{u\in\mathcal{U}}\sum_{S\in\mathcal{D}_u}|S|} \quad (3)$$

where $\mathcal{U}$ is the set of all test clients and $\mathcal{D}_u$ stands for the test segment of client $u$'s dataset. Equivalently, the model quality is measured by $\text{EMR}_3$ on the centralized test segments of all test clients.

## Experiment Details

The training recipes reported below are the same for all datasets unless stated otherwise. At each FL training round, 96 clients are randomly selected to participate in local training, with local epochs set to be 1 and training batch size as 8. Each global training epoch consists of $\#\text{users}/96$ training rounds, where the total number of global training epochs are chosen to be 6 and 3 for Pushshift.io Reddit and Stackoverflow respectively. We use SGD without momentum as client optimizer, and FedAdam for server side optimizer with weight decay of $10^{-5}$, $\beta_1$ as 0.9, $\beta_2$ as 0.999, and $\epsilon$ as $10^{-8}$. We do hyperparameter search on both client and server learning rates, with ranges $[10^{-6}, 0.05]$ and $[10^{-5}, 1]$ respectively. In Table 2 we specify the best choices for both baselines on each dataset. Every hyperparameter search contains 64 experiments (with 8 running in parallel at a time. See Appendix for the hyperparameters). For each set of hyperparameters, we run 10 local epochs that early stops when the validation $\text{EMR}_3$ did not improve.

| | OOV-as-UNK | OOV-Oracle | OOV Expansion |
|---|---|---|---|
| OOV rate on Pushshift.io Reddit | 8.8% | 5.5% | 0.2% |
| OOV rate on Stackoverflow | 4.8% | 3.0% | 0.001% |
| # Model parameters | 1.76M | 2.76M | 2.12M |

Table 1: Comparison of OOV rates and number of model parameters among 3 methods

## Results

**Improves overall performance** As shown in Figure 2, our method achieves $2.1\%$ and $1.0\%$ (resp. $5.6\%$ and $2.5\%$) absolute (resp. relative) gains of $\text{EMR}_3$ compared to the standard approach (i.e. OOV-as-UNK) on Pushshift.io Reddit and Stackoverflow respectively. It shows competitive performance even when compared with the stronger baseline OOV-Oracle, seeing $3.7\%$ and $1.7\%$ relative $\text{EMR}_3$ gain on Pushshift.io Reddit and Stackoverflow respectively. The fact that OOV expansion yields better improvements on Pushshift.io Reddit relative to Stackoverflow partially attributes to the higher heterogeneity of Pushshift.io Reddit data, which is reflected by the larger OOV rates (see Table 1) and the worse long tail behavior (ref. Figure 4).

**Increases word-coverage rates** As Table 1 shows, the OOV expansion approach reduces the OOV rate of OOV-as-UNK (resp. OOV-oracle) by more than $97.7\%$ and $99.9\%$ (resp. $96.4\%$ and $99.9\%$) on Pushshift.io Reddit and Stackoverflow respectively.

**More parameter efficient** As Table 1 indicates, OOV expansion uses $24\%$ less parameters than the OOV-Oracle during personalization. In addition, our approach does not require any extra training parameters during pretraining or FL. Consequently, it is $36\%$ more parameter-efficient in FL compared to OOV-Oracle.

**Personalization is essential** Taking Pushshift.io Reddit for instance, the standard personalization can already improve the global model accuracy by $6.5\%$ relatively on $\text{EMR}_3$ (see Figure 2). With OOV expansion we can further boost this quality gain up to $12.5\%$.

**Adapter is necessary** As can be seen from the first plot in Figure 3, adapter yields $15\%$ relative $\text{EMR}_3$ gain over trivial adapter (i.e. identity block) on Pushshift.io Reddit. For Stackoverflow dataset, inserting adapter achieves 2.6 times accuracy of the trivial adapter approach. In the second plot we present quality comparison under a new metric denoted by $\text{KEMR}_K$ which represents top $K$ known word exact match rates, defined by formula (4).

From Figure 3, we see that adapter not only helps achieves better OOV-understanding (i.e. higher EMR), the fact that it consistently improves accuracy on known word implies the important role of adapter in suppressing the issue of forgetting the knowledge from pretraining and federated learning stage.

## Conclusion

This paper proposes a personalized federated learning method that enables out-of-vocabulary words understanding for a class of on-device language models with closed vocabulary. By evaluating on two public benchmarks, we show that our method significantly outperforms the commonly used personalization approach in terms of next-word-prediction accuracy and drastically reduce the unknown-word rate on average while preserving user privacy and being parameter efficient.

# Appendix

## Hyperparameters for Personalization

During personalizations, we perform hyperparameter search at each client among the following grid (if applied):

$$\text{lr} \in \{10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1, 10\},$$
$$\sigma \in \{0, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 0.1, 1\},$$
$$\vec{\mathbf{H}} \in \{(960,), (128, 256, 128), (256, 512, 256)\}.$$

Here lr denotes the learning rate for local training; $\sigma$ stands for the standard deviation of Gaussian distribution used to initialize the adapter parameters (except for LayerNorm); $\vec{\mathbf{H}}$ represents the hidden dimensions of adapter. Note the grid only include $\sigma$ and $\vec{\mathbf{H}}$ during OOV expansion.

| | OOV-as-UNK | | OOV-oracle | |
|---|---|---|---|---|
| | client | server | client | server |
| Pushshift.io Reddit | 0.840 | 0.003 | 0.258 | 0.004 |
| Stackoverflow | 0.168 | 0.005 | 0.129 | 0.008 |

Table 2: Best learning rates for two baselines

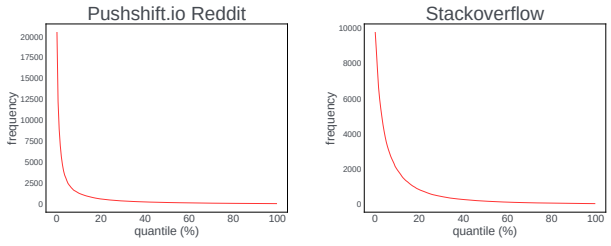



Figure 4: The quantile plot of word frequency for top 10k words from 2 datasets that demonstrates the long-tail phenomenon of OOV.

## Known Word Exact Match Rate

$$\text{KEMR}_K(\mathcal{D}) = \frac{\sum_{S \in \mathcal{D}} \sum_{w \in S \cap \mathcal{V}} 1_{\{w \in \text{Top}_K \text{pred}(w)\}}}{\sum_{S \in \mathcal{D}} |S \cap \mathcal{V}|} \quad (4)$$

where $\mathcal{V}$ is the closed vocabulary.

## References


Bagdasaryan, E.; Song, C.; van Dalen, R. C.; Seigel, M. S.; and Cahill, A. 2022. Training a Tokenizer for Free with Private Federated Learning. *ArXiv*, abs/2203.09943.

Bekal, D.; Shenoy, A.; Sunkara, M.; Bodapati, S.; and Kirchhoff, K. 2021. Remember the Context! ASR Slot Error Correction Through Memorization. In *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 236–243.

Chen, M.; Mathews, R.; Ouyang, T. Y.; and Beaufays, F. 2019a. Federated Learning Of Out-Of-Vocabulary Words. *ArXiv*, abs/1903.10635.

Chen, M.; Suresh, A. T.; Mathews, R.; Wong, A.; Allauzen, C.; Beaufays, F.; and Riley, M. 2019b. Federated Learning of N-Gram Language Models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 121–130. Hong Kong, China: Association for Computational Linguistics.

Chen, M. X.; Lee, B.; Bansal, G.; Cao, Y.; Zhang, S.; Lu, J.; Tsay, J.; Wang, Y.; Dai, A. M.; Chen, Z.; Sohn, T.; and Wu, Y. 2019c. Gmail Smart Compose: Real-Time Assisted Writing. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Dingliwal, S.; Shenoy, A.; Bodapati, S.; Gandhe, A.; Gadde, R. T.; and Kirchhoff, K. 2021. Efficient domain adaptation of language models in ASR systems using Prompt-tuning. *CoRR*, abs/2110.06502.

Dinh, C. T.; Tran, N. H.; and Nguyen, T. D. 2020. Personalized Federated Learning with Moreau Envelopes. *ArXiv*, abs/2006.08848.

Fallah, A.; Mokhtari, A.; and Ozdaglar, A. E. 2020. Personalized Federated Learning: A Meta-Learning Approach. *ArXiv*, abs/2002.07948.

Hanzely, F.; and Richtárik, P. 2020. Federated Learning of a Mixture of Global and Local Models. *ArXiv*, abs/2002.05516.

He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; and Neubig, G. 2021. Towards a Unified View of Parameter-Efficient Transfer Learning. *ArXiv*, abs/2110.04366.

Hilmkil, A.; Callh, S.; Barbieri, M.; Sütfeld, L. R.; Zec, E. L.; and Mogren, O. 2021. Scaling Federated Learning for Fine-tuning of Large Language Models. In *International Conference on Applications of Natural Language to Data Bases*.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; de Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-Efficient Transfer Learning for NLP. In *International Conference on Machine Learning*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. *ArXiv*, abs/2106.09685.

Khandelwal, U.; Levy, O.; Jurafsky, D.; Zettlemoyer, L.; and Lewis, M. 2019. Generalization through Memorization: Nearest Neighbor Language Models. *CoRR*, abs/1911.00172.

Kudo, T.; and Richardson, J. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Conference on Empirical Methods in Natural Language Processing*.

Kulkarni, V.; Kulkarni, M. V.; and Pant, A. 2020. Survey of Personalization Techniques for Federated Learning. *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 794–797.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2020. Ditto: Fair and Robust Federated Learning Through Personalization. In *International Conference on Machine Learning*.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. *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)*, abs/2101.00190.

Lin, B. Y.; He, C.; Zeng, Z.; Wang, H.; Huang, Y.; Soltanolkotabi, M.; Ren, X.; and Avestimehr, S. 2021. FedNLP: Benchmarking Federated Learning Methods for Natural Language Processing Tasks. In *NAACL-HLT*.

Liu, M.; Ho, S.; Wang, M.; Gao, L.; Jin, Y.; and Zhang, H. 2021. Federated Learning Meets Natural Language Processing: A Survey. *ArXiv*, abs/2107.12603.

Mahabadi, R. K.; Henderson, J.; and Ruder, S. 2021. Compacter: Efficient Low-Rank Hypercomplex Adapter Layers. In *NeurIPS*.

Mathur, A.; Beutel, D. J.; de Gusmão, P. P. B.; Fernández-Marqués, J.; Topal, T.; Qiu, X.; Parcollet, T.; Gao, Y.; and Lane, N. D. 2021. On-device Federated Learning with Flower. *ArXiv*, abs/2104.03042.

McMahan, H. B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2016. Communication-Efficient Learning of Deep Networks from Decentralized Data. In *International Conference on Artificial Intelligence and Statistics*.

Merity, S.; Xiong, C.; Bradbury, J.; and Socher, R. 2016. Pointer Sentinel Mixture Models. *ArXiv*, abs/1609.07843.

Pfeiffer, J.; Kamath, A.; Rücklé, A.; Cho, K.; and Gurevych, I. 2020. AdapterFusion: Non-Destructive Task Composition for Transfer Learning. *ArXiv*, abs/2005.00247.

Pillutla, K.; Malik, K.; Mohamed, A.; Rabbat, M. G.; Sanjabi, M.; and Xiao, L. 2022. Federated Learning with Partial Model Personalization. *ArXiv*, abs/2204.03809.

Qin, L.; and Rudnicky, A. I. 2013. Finding recurrent out-of-vocabulary words. In *INTERSPEECH*.

Qiu, X.; Fernández-Marqués, J.; Gusmão, P.; Gao, Y.; Parcollet, T.; and Lane, N. D. 2022. ZeroFL: Efficient On-Device Training for Federated Learning with Local Sparsity. *ArXiv*, abs/2208.02507.

Ro, J. H.; Breiner, T.; McConnaughey, L.; Chen, M.; Suresh, A. T.; Kumar, S.; and Mathews, R. 2022. Scaling Language Model Size in Cross-Device Federated Learning. *ArXiv*, abs/2204.09715.

Schuster, M.; and Nakajima, K. 2012. Japanese and Korean voice search. *2012 IEEE International Conference on*

*Acoustics, Speech and Signal Processing (ICASSP)*, 5149–5152.

Sennrich, R.; Haddow, B.; and Birch, A. 2015. Neural Machine Translation of Rare Words with Subword Units. *ArXiv*, abs/1508.07909.

Shenoy, A.; Bodapati, S.; Sunkara, M.; Ronanki, S.; and Kirchhoff, K. 2021. "What's The Context?" : Long Context NLM Adaptation for ASR Rescoring in Conversational Agents. *CoRR*, abs/2104.11070.

Singhal, K.; Sidahmed, H.; Garrett, Z.; Wu, S.; Rush, K.; and Prakash, S. 2021. Federated Reconstruction: Partially Local Federated Learning. In *Neural Information Processing Systems*.

Stickland, A. C.; and Murray, I. 2019. BERT and PALs: Projected Attention Layers for Efficient Adaptation in Multi-Task Learning. In *International Conference on Machine Learning*.

Wang, K.; Mathews, R.; Kiddon, C.; Eichner, H.; Beaufays, F.; and Ramage, D. 2019. Federated Evaluation of On-device Personalization. *ArXiv*, abs/1910.10252.

Wu, X.; Liang, Z.; and Wang, J. 2020. FedMed: A Federated Learning Framework for Language Modeling. *Sensors (Basel, Switzerland)*, 20.

Yousefpour, A.; Guo, S.; Shenoy, A.; Ghosh, S.; Stock, P.; Maeng, K.; Krüger, S.-W.; Rabbat, M.; Wu, C.-J.; and Mironov, I. 2023. Green Federated Learning. In *Federated Learning and Analytics in Practice: Algorithms, Systems, Applications, and Opportunities*.

Yu, T.; Bagdasaryan, E.; and Shmatikov, V. 2020. Salvaging Federated Learning by Local Adaptation. *ArXiv*, abs/2002.04758.