# ENABLING ON-DEVICE TRAINING OF SPEECH RECOGNITION MODELS WITH FEDERATED DROPOUT

*Dhruv Guliani*　*Lillian Zhou*　*Changwan Ryu*　*Tien-Ju Yang*
*Harry Zhang*　*Yonghui Xiao*　*Françoise Beaufays*　*Giovanni Motta*

Google LLC, Mountain View, CA, U.S.A.
{dguliani, lqz, changwan}@google.com

## ABSTRACT

Federated learning can be used to train machine learning models on the edge on local data that never leave devices, providing privacy by default. This presents a challenge pertaining to the communication and computation costs associated with clients' devices. These costs are strongly correlated with the size of the model being trained, and are significant for state-of-the-art automatic speech recognition models.

We propose using federated dropout to reduce the size of client models while training a full-size model server-side. We provide empirical evidence of the effectiveness of federated dropout, and propose a novel approach to vary the dropout rate applied at each layer. Furthermore, we find that federated dropout enables a set of smaller sub-models within the larger model to independently have low word error rates, making it easier to dynamically adjust the size of the model deployed for inference.

***Index Terms***— federated learning, speech recognition, federated dropout

## 1. INTRODUCTION

End-to-end neural networks have improved to surpass conventional server-side Automatic Speech Recognition (ASR) systems [1]. Advances with the Conformer architecture [2, 3] have pushed the quality and latency envelopes further, enabling highly performant on-device ASR.

End-to-end neural ASR models can be trained using Federated Learning (FL) [4], a privacy-preserving training technique which removes the need to send raw user-data to servers. FL optimization proceeds in synchronous *rounds* of training [5], requiring a set of *clients* (devices) to receive copies of a model at the start of local training and send back model updates for aggregation after optimization.

Unlike other models which have been successfully trained on the edge under FL [6, 7, 8, 9], ASR models typically contain over 100M parameters (see Table 1), and it is likely that model sizes will keep increasing [10]. This exacerbates communication and computation costs [4, 11] associated with training such models in production. Costs include those of sending and aggregating models, dealing with heterogeneous network dynamics, performing privacy and security computations, and on-device memory.

| **Conformer** | **Conf Params** | **Total Params** |
|---|---|---|
| Non-Streaming [2] | 107.5M | 119M |
| Streaming [3] | 113M | 137M |

**Table 1**. Architecture of SOTA Conformer models.

Federated Dropout [12] (FD) is a technique developed to enable training larger models under FL by reducing the size of models trained on clients. It works by leveraging the key insight from dropout [13] that dropping intermediate activations in a network is equivalent to a structural removal of certain rows, columns (and generally, *slices*) of adjacent parameter matrices. An adaptive alternative [14] has also been proposed to better estimate which activations to drop. By reducing the size of client models directly, this technique reduces both communication and computation costs.

In this work, we study the applications of FD to ASR models with the explicit goal of training a full-sized ASR model server-side. We make the following contributions:

- We show that FD can be successfully applied to ASR models and provides a useful quality/cost trade-off.
- We extend the technique to realistic Google-scale workloads and use varying per-layer dropout rates to achieve better quality with the same size reduction.
- We find that training models with FD is an effective way to find well-performing sub-models within a larger model, enabling the size of the model to be reduced for on-device inference. This is useful for deployment on a population of devices having different capabilities.

## 2. METHODOLOGY

### 2.1. Model and Data

We use the largest *Non-Streaming* Conformer described in [2], designed for single-domain and short utterances, to conduct initial experiments with FD. This Conformer is trained from scratch under FL with the Librispeech [15] corpus split by speaker [4]. Since we are training these models using FL, we switch from batch normalization to group normalization [16], which is more suitable for a federated setting [17, 18]. In doing so, we incur a small penalty in model quality.

We then extend our work to a production-grade Google use case by performing a domain adaptation task using a *Streaming* Conformer. With 137M trainable parameters, the Streaming Conformer is similar to the one described in [3], but without two-pass re-scoring and cascaded encoders. We train this model to convergence on multi-domain (MD) utterances collected from domains of search, farfield, telephony, YouTube, etc. [19, 20] (dataset size in Table 2). All data are anonymized and hand-transcribed. In the first pass of training, we withhold one dataset (in this case, the Medium Form (MF) data, which has an average duration of 10.4 seconds), later refining on data from this domain in a FL simulation. The model is evaluated on a disjoint test set from the MF domain. This is a realistic setting for federated training of ASR models, wherein a well trained server-side model is adapted to a new domain with FL on edge devices. For federated domain adaptation, the data remain anonymously grouped into simulated clients, with each client contributing a uniform amount of data per round. The impact of non-uniform data was previously studied [4] and is left out of scope for this investigation. Both Conformers studied are encoder-decoder models trained with RNN-T loss.

| Dataset | Hours |
|---|---|
| Multi-domain (MD) | 400k |
| Medium-form domain (MF) | 26k |
| Medium-form held out (MD-MF) | 374k |

**Table 2**. Summary of data sources.

In both experiments, we studied the composition of the models to decide which layers to apply FD to, as the Conformer has various components with different types of layers. We limit our application of FD to just one layer type, and acknowledge that it can be extended to the others in future work. Based on the breakdown in Table 3, we concluded that

| Portion | Non-Streaming | Streaming |
|---|---|---|
| FeedForward | 60% | 55% |
| Attention | 19% | 15% |
| Convolution | 14% | 12% |
| Decoder | 3% | 17% |
| Other | 4% | 1% |

**Table 3**. Percentage of parameters that exist in each component of the Conformer models that were studied.

the FeedForward layers in the encoder were best suited for FD. Dropout rates throughout the paper refer to the ratio of activations dropped within the encoder's FeedForward layers, unless otherwise specified.

## 2.2. Federated Dropout

Federated Dropout differs from traditional dropout in two main ways. First, the technique is designed to reduce the size of client models for federated training [12] rather than regularizing models. Second, the technique works as a functional transformation during the FL procedure [21], wherein smaller client models are *sampled* from the full model for local optimization. Algorithm 1 illustrates this procedure, where each client participating in a round trains a different *sub-model* within the full-size model. In this procedure, the server or client must maintain a reference linking each client's sub-model to the corresponding portions of the full-size model it updates.

Let $K$ be the number of clients participating in a given federated round $r$. Let $w$ be the model parameters, with $W$ being the set of $\{w_1, w_2, \ldots, w_k\}$ sub-models that are to be trained by the participating clients in a given round. Let $M$ be the set of $\{m_1, m_2, \ldots, m_k\}$ mappings that encode which portion each client's sub-model updates within the full model.

**Algorithm 1** *FedDrop*. The $K$ clients are indexed by $k$, rounds are indexed by $r$, and dropout rate is $d$. Shrink($w$, $m$) reduces model weight matrices according to $m$, and Expand($w$, $m$) does the opposite.

```
1: for each round $r$ = 1,2,... do
2:     $M^r \leftarrow GenerateMappings(d)$
3:     $W^r \leftarrow Shrink(w^r, M^r)$        ▷ Set of client models
4:     for each client $k \in K$ in parallel do
5:         $\hat{w}_k^r \leftarrow ClientUpdate(k, w_k^r)$
6:         $\Delta w_k^r = w_k^r - \hat{w}_k^r$
7:     end for
8:     $w^{r+1} \leftarrow ServerUpdate(w^r, Expand(\Delta W^r, M^r))$
9: end for
```

As can be inferred from Algorithm 1, the selection of mappings $M$ determines how Federated Dropout is applied during each round. We study the two edge cases in which, at each round, either every mapping in $M$ is unique, or all mappings are the same. We call the former Per-Client-Per-Round (PCPR) FD, wherein each participating client updates a different sub-model. We call the latter Per-Round (PR) FD, during which each client updates the same sub-model. This sub-model always changes from one round to the next. Although we hypothesize PCPR would yield the best results, the PR scheme is simpler from a production engineering viewpoint. This is because the PR scheme allows for the identity of each client to be hidden, as the infrastructure need not remember which parts of the full model a particular client updates. This anonymization can be achieved with the PCPR scheme, but requires more engineering effort.

### 2.3. Per-Layer Federated Dropout

We explore making FD more effective by varying the amount of dropout applied across different layers. We target layers for additional dropout using the idea that certain layers of may be *ambient*, or less important to the model's performance [22]. Ablation experiments involving resetting a model's parameter values to initial/random values can determine which layers are ambient. We apply additional dropout to ambient layers to improve upon the results of uniform FD.

### 2.4. Sub-Model Evaluations

Other properties of FD are investigated by sampling and evaluating sub-models from the model after federated training. Sub-models are obtained by randomly removing activations and corresponding neurons in the same way as the FD procedure. The difference is that sub-models are sampled after training from a full-size server-side model, as opposed to dynamically during training (as per Algorithm 1). These models are directly evaluated without any further training.

## 3. EXPERIMENTS

### 3.1. Baselines

Initial experiments established baselines for both the from-scratch and domain adaptation set-ups. These experiments used *SGD* as the client-side optimizer and *Adam* [18] server-side. SpecAugment [23] was applied to utterances in both cases. Additionally, 128 clients participated in each federated round as informed by previous investigations [4]. Table 4 summarizes the result for from-scratch training.

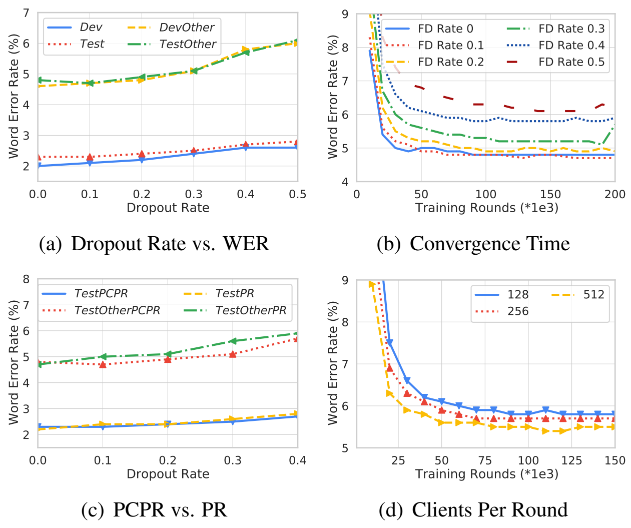
| Exp. | WER | | | |
|---|---|---|---|---|
| | *Test* | *TestOther* | *Dev* | *DevOther* |
| Non-Streaming Baseline | 2.0 | 4.6 | 2.3 | 4.8 |

**Table 4**. Baseline for from-scratch experiments.

For domain adaptation experiments, Table 5 shows a *No MF Baseline* which was constructed by training the Streaming model on MD - MF data (Table 2). This was the starting point for domain adaptation.

### 3.2. Training From Scratch

We trained the non-streaming Conformer under FL with varying FD rates, clients per round, and dropout schemes to explore the impact of FD on quality and convergence.



(a) Dropout Rate vs. WER

(b) Convergence Time

(c) PCPR vs. PR

(d) Clients Per Round

**Fig. 1**. Experimental results showing the impact of Federated Dropout on model quality and convergence with non-IID Librispeech sets. Figures 1(b), 1(c), and 1(d) report results on the *TestOther* evaluation set, although similar trends were observed across all other evaluation sets. Figure 1(d) reports results using 40% FD.

Figure 1 shows that increasing dropout rates led to slightly worse models (Figure 1(a)) and slower convergence (Figure 1(b)) across all evaluation sets. Concretely, a 0.5% absolute WER regression was observed at the highest dropout rate studied (50%). At this dropout rate, client models were approximately 30% smaller in size. Dropout rates up till 20% had minimal quality loss and changes in convergence time, showing a useful trade-off between quality and cost.

The PR scheme (Figure 1(c)) incurred some quality degradation compared to PCPR, but confirmed that the model can be trained effectively by updating only one sub-model in each federated round. With 30% dropout, the PR scheme showed a 0.4% absolute quality degradation in comparison with PCPR. We hence assert this to be a viable and simpler alternative to PCPR with small variations in quality.

Finally, although Figure 1(a) showed that having a higher dropout rate increases convergence time, we empirically show in Figure 1(d) that increasing the number of clients per round can compensate for this. For this particular configuration, model quality was also seen to improve slightly. We therefore claim that quality and convergence time with FD may be improved by tuning hyper-parameters such as clients per round and learning rate.

### 3.3. Domain Adaptation

Domain adaptation experiments (Table 5) showed that PCPR FD could reduce the size of each client model by up to 22% while providing a 3% relative WER reduction on the previ-

| **Exp.** | **FD (%)** | **Medium Form WER** | | |
|---|---|---|---|---|
| | | *Size Red. (%)* | *PCPR* | *PR* |
| No MF Baseline | 0 | None | 6.7 | 6.7 |
| MF Domain Ad. | 0 | None | 4.4 | 4.4 |
| " | 10 | 6 | 4.4 | 4.4 |
| " | 20 | 11 | 4.7 | 4.7 |
| " | 30 | 17 | 5.4 | 5.5 |
| " | 40 | 22 | 6.5 | 6.6 |

**Table 5**. Streaming Conformer experiments with FD.

ously unseen domain. If a higher quality is desired, client model sizes can be reduced by 11% with a 30% relative WER improvement on the new domain. Previous results with PR FD also held with the domain adaptation task, wherein PR performed well with some minor quality loss. Higher dropout rates (50% and greater) resulted in degradation in MF WER from the No MF Baseline.

## 3.4. Varying Dropout According to Ambient Properties

Next, we investigated making FD more effective by using variable dropout rates across the Conformer blocks. We leveraged *ambient layer* findings from [22] to more efficiently target FD. We increased the amount of dropout applied to ambient layers, enabling us to drop more parameters overall without sacrificing model quality. The results in Figure 2 show that this can provide configurations with smaller client models that give better WER compared to uniform FD throughout the model.

Compared to a flat 10% dropout, we show a setting with variable dropout that gives the same WER, using 1.5% fewer parameters, representing a savings of 2 million parameters. In comparison with a flat 20% dropout, we show a setting with the same number of parameters with a relative 2% WER improvement. We hypothesize that more exhaustive searches of varying dropout rates could yield even more compelling configurations of client model size versus quality.

## 3.5. Quality of Sub-Models

Finally, we studied sub-model WERs using the domain adaptation experimental setup. A Streaming Conformer was trained under FL with a 50% FD rate on the MF task, and a second model was trained without any FD as the control. Sub-models were sampled (50) from each of the models after training, randomly applying 50% dropout for each sample. Table 6 compares the WERs of sampled sub-models, showing that WERs of sub-models without FD degraded to catastrophic values higher than 50% on average. In contrast, it was found that FD effectively enabled a flexible set of sub-models to achieve a much lower (sub-10%) WER, with lower variance. We hypothesize that FD implicitly regularizes the model towards sparsity by making it robust to the

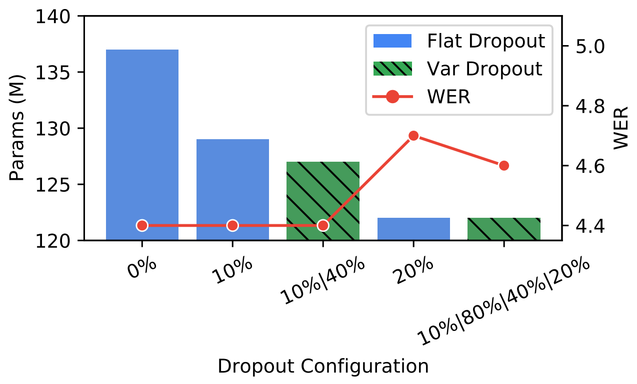


**Fig. 2**. Comparison of uniform dropout (solid) and varying dropout (hatched). The labels are of format $d_f|d_1|d_2$... where $d_f$ is flat dropout throughout the model, and $d_n$ denotes dropout applied to the $n^{th}$ most ambient layer.

removal of certain paths. We conclude that FD can improve the quality of sub-models and enable the same full model to be trimmed down for deployment to devices with different compute capabilities.

| **Exp.** | **Mean WER** | **Std. Dev** |
|---|---|---|
| Without FD | 50.3 | 5.6 |
| With FD | 9.5 | 0.2 |

**Table 6**. WERs of sub-models sampled from the full model.

## 4. CONCLUSION

Federated Learning is key to user privacy and ensures that raw user data never leave the device. To leverage this, we must be able to fit model training onto edge devices. End-to-end neural ASR models can contain well over 100 million parameters, creating significant communication and computation cost hurdles on the edge. We argued that Federated Dropout is a promising technique to reduce this cost and explored various configurations to improve its effectiveness. We illustrated a usable quality-cost trade off allowing for client model size reduction between 6-22%, with WER improvements in a domain adaptation setting ranging from 34-3% respectively. We also showed that FD causes capable sub-models to form within the full model, allowing the same model to be down-sampled for inference. We hope this work inspires deeper investigations and applications of both client model size reduction and sub-model training.

## 5. ACKNOWLEDGEMENTS

We would like to thank Khe Chai Sim and Jakub Konečný, for providing valuable insights on this paper's contents.

## 6. REFERENCES

[1] T. N. Sainath, Y. He, B. Li *et al.*, "A Streaming On-Device End-to-End Model Surpassing Server-Side Conventional Model Quality and Latency," *CoRR*, vol. abs/2003.12710, 2020. https://arxiv.org/abs/2003.12710

[2] A. Gulati, C.-C. Chiu, J. Qin *et al.*, Eds., *Conformer: Convolution-augmented Transformer for Speech Recognition*, 2020.

[3] B. Li, A. Gulati, J. Yu *et al.*, "A Better and Faster end-to-end Model for Streaming ASR," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 5634–5638.

[4] D. Guliani, F. Beaufays, and G. Motta, "Training Speech Recognition Models with Federated Learning: A Quality/Cost Framework," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3080–3084.

[5] K. Bonawitz, H. Eichner *et al.*, "Towards Federated Learning at Scale: System Design," in *SysML 2019*, 2019, to appear. https://arxiv.org/abs/1902.01046

[6] F. Beaufays, K. Rao, R. Mathews *et al.*, "Federated Learning for Emoji Prediction in a Mobile Keyboard," 2019. https://arxiv.org/abs/1906.04329

[7] A. Hard, K. Rao, R. Mathews *et al.*, "Federated Learning for Mobile Keyboard Prediction," *CoRR*, vol. abs/1811.03604, 2018. http://arxiv.org/abs/1811.03604

[8] T. Yang, G. Andrew, H. Eichner *et al.*, "Applied Federated Learning: Improving Google Keyboard Query Suggestions," *CoRR*, vol. abs/1812.02903, 2018. http://arxiv.org/abs/1812.02903

[9] A. Hard, K. Partridge, C. Nguyen *et al.*, "Training Keyword Spotting Models on Non-IID Data with Federated Learning," 2020.

[10] B. Li, R. Pang, T. N. Sainath *et al.*, "Scaling end-to-end models for large-scale multilingual ASR," *CoRR*, vol. abs/2104.14830, 2021. https://arxiv.org/abs/2104.14830

[11] J. Wang, Z. Charles, Z. Xu *et al.*, "A Field Guide to Federated Optimization," *CoRR*, vol. abs/2107.06917, 2021. https://arxiv.org/abs/2107.06917

[12] S. Caldas, J. Konečný, B. McMahan *et al.*, "Expanding the Reach of Federated Learning by Reducing Client Resource Requirements," 2018. https://arxiv.org/abs/1812.07210

[13] N. Srivastava, G. Hinton, A. Krizhevsky *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014. http://jmlr.org/papers/v15/srivastava14a.html

[14] N. Bouacida, J. Hou, H. Zang *et al.*, "Adaptive federated dropout: Improving communication efficiency and generalization for federated learning," *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1–6, 2021.

[15] V. Panayotov, G. Chen, D. Povey *et al.*, "Librispeech: An ASR Corpus Based on Public Domain Audio Books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[16] Y. Wu and K. He, "Group normalization," *CoRR*, vol. abs/1803.08494, 2018. http://arxiv.org/abs/1803.08494

[17] K. Hsieh, A. Phanishayee, O. Mutlu *et al.*, "The non-iid data quagmire of decentralized machine learning," *CoRR*, vol. abs/1910.00189, 2019. http://arxiv.org/abs/1910.00189

[18] S. J. Reddi, Z. Charles, M. Zaheer *et al.*, "Adaptive federated optimization," *CoRR*, vol. abs/2003.00295, 2020. https://arxiv.org/abs/2003.00295

[19] A. Misra, D. Hwang, Z. Huo *et al.*, "A Comparison of Supervised and Unsupervised Pre-Training of End-to-End Models," in *Proc. Interspeech 2021*, 2021, pp. 731–735.

[20] A. Narayanan, R. Prabhavalkar, C.-C. Chiu *et al.*, "Recognizing Long-Form Speech Using Streaming End-to-End Models," *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 920–927, 2019.

[21] H. B. McMahan, E. Moore *et al.*, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, ser. Proceedings of Machine Learning Research, A. Singh and X. J. Zhu, Eds., vol. 54. PMLR, 2017, pp. 1273–1282. http://proceedings.mlr.press/v54/mcmahan17a.html

[22] L. Zhou, D. Guliani, A. Kabel *et al.*, "Exploring heterogeneous characteristics of layers in ASR models for more efficient training," *CoRR*, vol. abs/2110.04267, 2021. https://arxiv.org/abs/2110.04267

[23] D. S. Park, W. Chan, Y. Zhang *et al.*, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," *Interspeech 2019*, Sep 2019. http://dx.doi.org/10.21437/Interspeech.2019-2680