Google Research

Paper#: 2057

icassp 2022 Singapore

Enabling On-Device Training of Speech Recognition Models with Federated Dropout

Dhruv Guliani (dguliani@google.com), Lillian Zhou, Changwan Ryu, Tien-Ju Yang, Harry Zhang, Yonghui Xiao, Françoise Beaufays, Giovanni Motta

Background

Introduction

End-to-end neural ASR models can be trained using Federated Learning (FL) [4] to preserve user privacy by removing the need to send raw user-data to servers. FL optimization proceeds in synchronous rounds [5], sending a set of clients (devices) copies of a model for local training and aggregating model updates after optimization.

Costs Associated with FL

ASR models are much larger than models previously trained with FL [6, 7, 8, 9], often containing over **100M parameters**. The following are costs associated with FL:

- Communication costs: sending and aggregating models, dealing with heterogeneous network dynamics, etc.
- On-Device costs (CPU and memory usage)

Federated Dropout

Federated Dropout [12] (FD) reduces both communication and on-device computation costs by reducing the size of models trained on clients. FD leverages the 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. See Algorithm 1 for details.

In this work, we study the applications of FD to ASR and make the following contributions:

- Show that FD can be successfully applied to ASR models to provide a useful quality/cost trade-off.
- Extend FD to Google-scale workloads and use varying per-layer dropout rates to improve quality.
- We find that FD is an effective way to train well-performing sub-models within a larger model, enabling the size of the model to be reduced for on-device inference depending on device capabilities.

Algorithm 1 *FedDrop*. Let K be the number of clients (indexed by k) 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. The dropout rate is d and Shrink(w, m) reduces model weight matrices according to m, while 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
```

Method

Model & Dataset

We use two variations of the Conformer architecture shown in Figure 1: a Non-Streaming Conformer [2] with 119M parameters and a Streaming Conformer [3] with 137M parameters. The Non-Streaming Conformer is trained from scratch under FL using a speaker-split Librispeech corpus [4]. The Streaming Conformer is first trained on a production-grade Multi-Domain (MD) dataset containing 376k hours of audio server-side, and then trained on Medium-Form (MF) data from a new domain under FL. We call these tasks "training from scratch" and "domain-adaptation" respectively.
Feed Forward layers are the largest parts of both models, making up 60% of the Non-Streaming Conformer and 55% of the Streaming Conformer. Our application of federated dropout is hence limited to just these layers for this investigation.

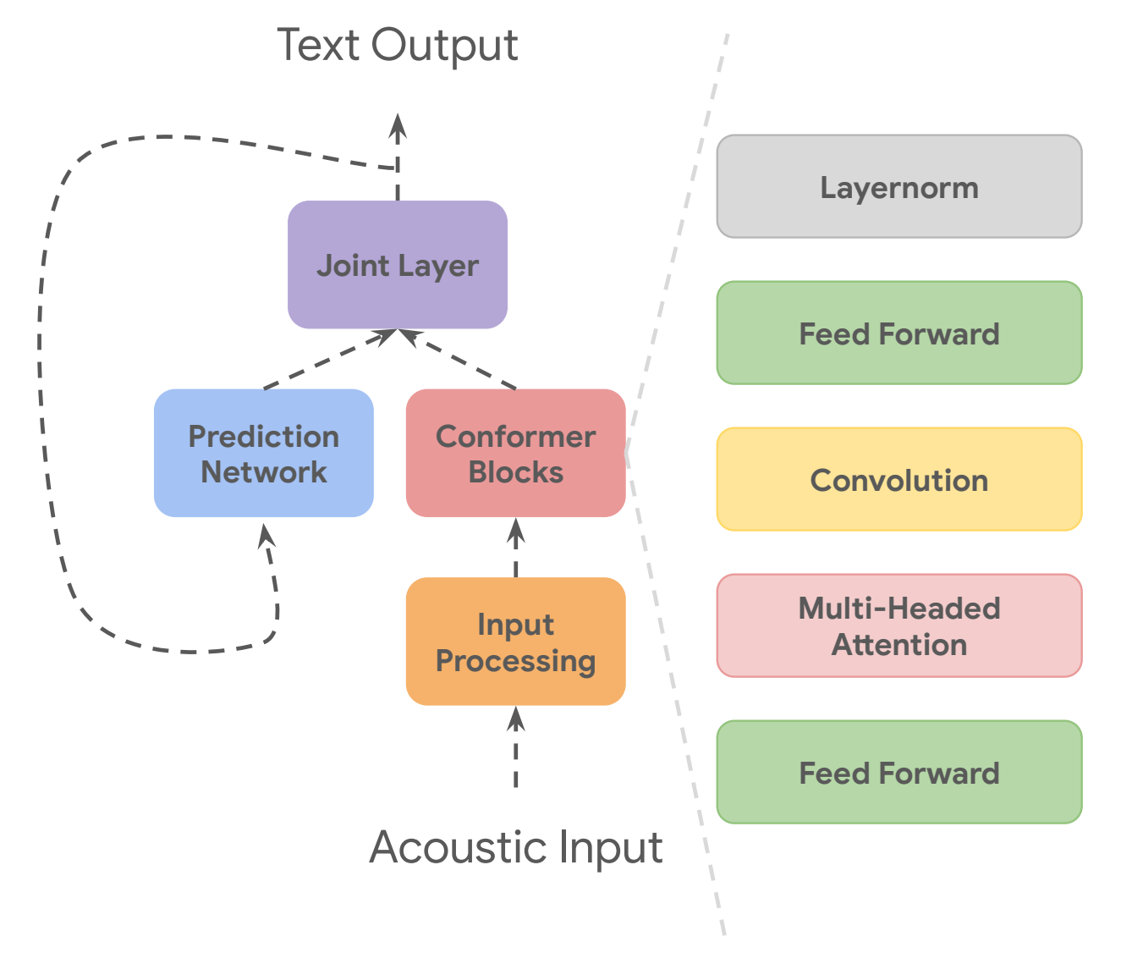


Figure 1: Conformer model architecture.

Training from Scratch

The training from scratch task is used to study the general characteristics of FD with ASR. Note that in Algorithm 1, two edge cases exist:

- all maps in M are unique: Per-Client-Per-Round (PCPR)
- all maps in M are the same: Per-Round (PR)

We explore dropout rates, report goals, PCPR vs PR FD, and comment on convergence time and quality.

Domain Adaptation

The domain adaptation task is a more 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.

Per-Layer FD

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 may be ambient, or less important to the model's performance [22], with the aim of improving upon the results of uniform FD.

Sub-Model Evaluations

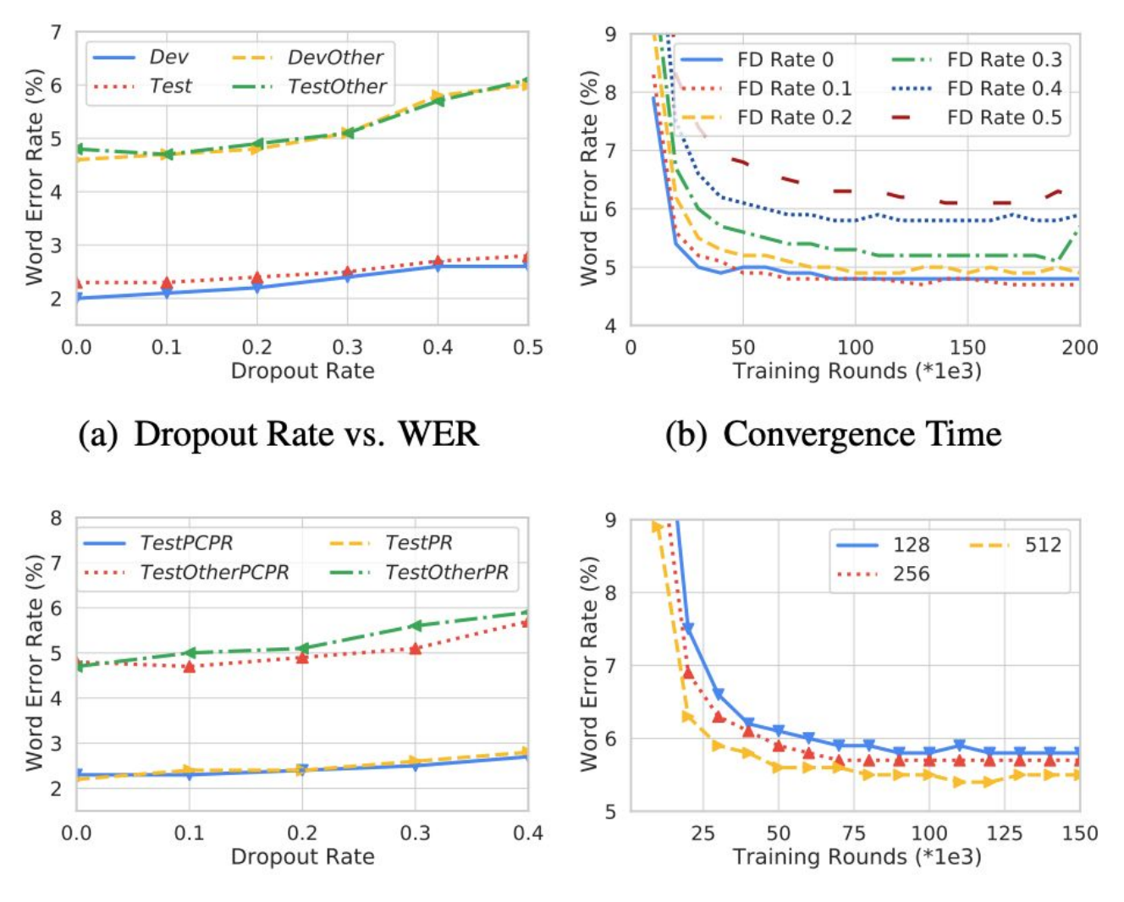
Other properties of FD are also investigated by sampling and evaluating sub-models from the full size model after training. Sub-models are obtained by removing activations and corresponding neurons in the same way as the FD procedure and are evaluated without any further training.

Experiments

Non-Streaming Conformer Results

Figure 2 shows:

a. FD can provide a quality/cost trade-off as WER gets slightly worse with increasing FD rate
b. Higher FD rates usually converge slower
c. PR is slightly worse than PCPR, but usable if engineering resources are limited
d. Higher report goals can improve convergence speed and quality. Further improvements are hypothesized to be possible with hparam tuning.



(a) Dropout Rate vs. WER (b) Convergence Time (c) PCPR vs. PR (d) Clients Per Round

Figure 2: Non-Streaming Conformer Results

Streaming Conformer Results

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 1: Streaming Conformer Initial Sweep

Domain adaptation in Table 1 shows:

- FD is effective with production-grade exps.
- PR performs very well; minor quality loss
- Higher dropout rates (50% and greater) result in degradation in MF WER from the No MF Baseline.

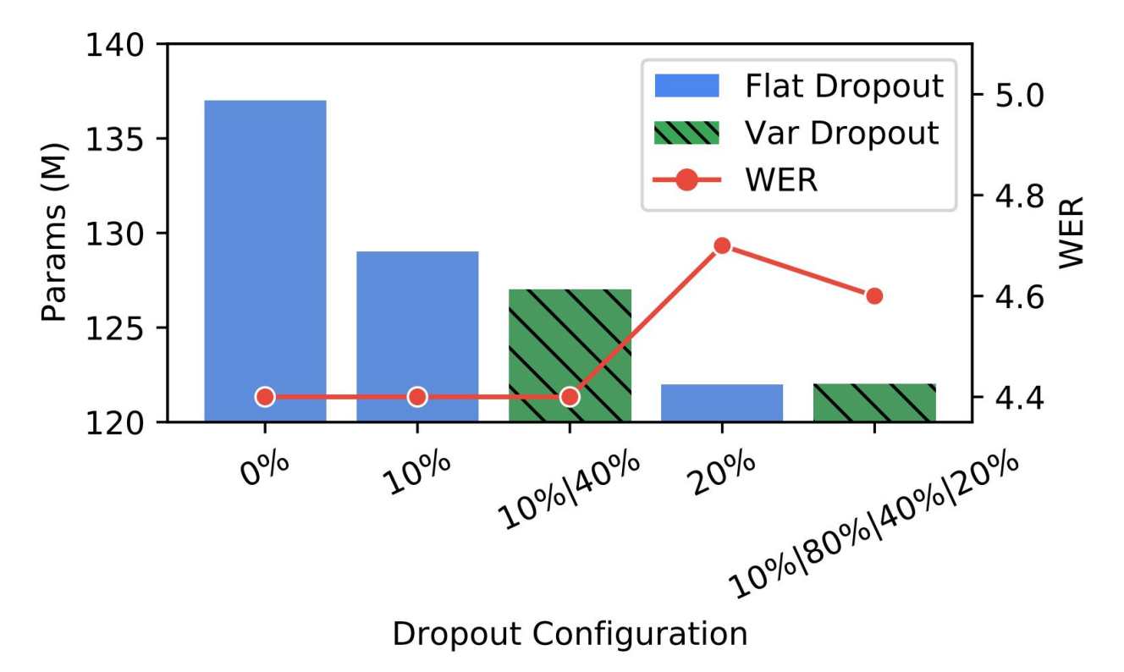


Figure 4: Per-Layer Dropout Rates

Per-layer dropout rates (Figure 4) show:

- It is possible to achieve better quality or lower cost with per-layer FD

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

Table 2: Quality of Sub-Models

Submodels were sampled (50) with 50% dropout from two streaming conformers, one trained under FL with FD and one without, with the result that:

- FD enabled sub-models to achieve a much lower WER with lower variance.

Conclusion

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.

Key References

[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-toend 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.

[12] S. Caldas, J. Konecnˇ y, 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

[22] C. Zhang, S. Bengio, and Y. Singer, "Are all layers created equal?" CoRR, vol. abs/1902.01996, 2019. http://arxiv.org/abs/1902.01996