Scaling Federated Learning for Fine-tuning of Large Language Models

Agrin Hilmkil[1], Sebastian Callh[1], Matteo Barbieri[1],
Leon René Sütfeld[2], Edvin Listo Zec[2], and Olof Mogren[2]

[1] Peltarion, peltarion.com
[2] RISE Research Institutes of Sweden

Abstract. Federated learning (FL) is a promising approach to distributed compute, as well as distributed data, and provides a level of privacy and compliance to legal frameworks. This makes FL attractive for both consumer and healthcare applications. However, few studies have examined FL in the context of larger language models and there is a lack of comprehensive reviews of robustness across tasks, architectures, numbers of clients, and other relevant factors. In this paper, we explore the fine-tuning of large language models in a federated learning setting. We evaluate three popular models of different sizes (BERT, ALBERT, and DistilBERT) on a number of text classification tasks such as sentiment analysis and author identification. We perform an extensive sweep over the number of clients, ranging up to 32, to evaluate the impact of distributed compute on task performance in the federated averaging setting. While our findings suggest that the large sizes of the evaluated models are not generally prohibitive to federated training, we found that not all models handle federated averaging well. Most notably, DistilBERT converges significantly slower with larger numbers of clients, and under some circumstances, even collapses to chance level performance. Investigating this issue presents an interesting direction for future research.

Keywords: Distributed · Federated learning · Privacy · Transformers

1 Introduction

Transformer-based architectures such as BERT have recently lead to breakthroughs in a variety of language-related tasks, such as document classification, sentiment analysis, question answering, and various forms of text-mining [23,2,1,21,25,10]. These models create semantic representations of text, which can subsequently be used in many downstream tasks [2]. The training process for Transformers typically includes two phases: pre-training and task-specific fine-tuning. During pre-training, the model learns to extract semantic representations from large, task-independent corpora. The pre-training is followed by task-specific fine-tuning on a separate dataset to optimize model performance further. In this paper, we study the effects of fine-tuning large language models in a federated learning (FL) setting. In FL, models are trained in a decentralized fashion on a number of local compute instances, called clients, and intermittently aggregated and synchronized via a central server. As such, FL is a solution which provides a level of privacy with regards to the sharing of personal or otherwise sensitive data. Model aggregation is commonly performed via averaging of the weights of the individual client

models, called Federated Averaging (FEDAVG) [16]. Depending on the application, the number of clients in an FL setting can differ wildly. In instances where smartphones are used as clients, their number can reach into the millions [5], whereas settings with higher compute requirements and more data per client will often range between a handful and a few dozens of clients. Here, we focus on the latter. A potential application of this is the medical field, in which automated analyses of electronic health records yield enormous potential for diagnostics and treatment-related insights [26].

Our contribution is a comprehensive overview of the applicability of the federated learning setting to large language models. To this end, we work with a fixed computation budget for each task, and use a fixed total amount of data while varying the number of clients between which the data is split up. This way, we isolate the effects of distributing data over several clients for distributed compute. We leave comparisons with a fixed amount of data per client, and varying non-i.i.d. data distributions between clients for future work. The main contributions of this paper are the following: (1) We provide a comparison of three popular Transformer-based language models in the federated learning setting, using the IMDB, Yelp F, and AG News datasets. (2) We analyze how the number of clients impacts task performance across tasks and model architectures. Finally, we share our code publicly[3].

2 Related work

Federated optimization was first introduced in [8]. The key challenges in this paradigm are communication efficiency when learning from many clients, privacy concerns with respect to leakage of client data, and variability in data distributions between clients (non-i.i.d. setting). FEDAVG [16] solves the federated optimization problem by building a global model based on local stochastic gradient descent updates and has been shown to work on non-i.i.d. data in some circumstances. Since then, many adaptations have arisen [11,18,7]. [4] proposes a one-shot FL algorithm, learning a global model efficiently in just one communication round. [28], [6] and [13] study effects of FEDAVG and non-i.i.d. client data. [17] and [5] train large recurrent language models with user-level differential privacy guarantees and for mobile keyboard prediction. [3] use federated learning for named entity recognition in heterogeneous medical data.

Most architectures used in FL to date are relatively small (e.g., CIFG for mobile keyboard prediction: 1.4M parameters [5]), compared to BERT-based language models with hundreds of millions of parameters. How these very large models behave under FEDAVG remains underexplored. To the best of our knowledge, [12] and [14] are the first ones to train large Transformer models in a federated setting. [14] trained BERT on a medical corpus and showed that both pre-training and fine-tuning could be done in a federated manner with only minor declines in task performance. Nonetheless, the study is mainly a proof-of-concept and does not explore many of the factors that can be expected in real-world scenarios. For instance, the authors only used five clients, and evaluated them only on i.i.d. data. [12] introduces FedDF, an ensemble distillation algorithm for model fusion. The authors train a central model through unlabeled data on

[3] https://github.com/Peltarion/scaling_fl

the client models outputs, and perform fine-tuning on a pre-trained DistilBERT [20] in a federated setting as a baseline. To the best of our knowledge, no systematic variation of the number of clients and other relevant factors has previously been explored in this context.

3 Method

3.1 Federated learning

Federated learning aims to solve the optimization problem

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{K} \sum_{k=1}^{K} F_k(\theta), \tag{1}$$

where $F_k(\theta) = \mathbb{E}_{x \sim \mathcal{D}_k} [\ell_k(\theta; x)]$ is the expected loss on client k and $\mathcal{D}_k$ is the data distribution of client k. In FEDAVG, a global model f_θ is initialized on a central server and distributed to all K clients, each of which then trains its individual copy of the network using SGD for E local epochs with local batch size B. The clients' updated parameters are then averaged on the central server, weighted by the local data size at each client. The averaged model is distributed to the clients again, and the process is repeated for a defined number of communication rounds. We implement FEDAVG using distributed PyTorch [19]. For each experiment we start from a pre-trained model, and fine-tune it with federated averaging on the current task.

3.2 Models

We include BERT with 110M parameters, 12 layers [2], ALBERT with 11M parameters, 12 layers [9] and DistilBERT with 65M parameters, 6 layers [20]. This allows us to study the effect that both the parameter count and the number of layers have on FEDAVG. All models are the corresponding base models pre-trained on (cased) English. In particular, it should be noted that while the models have similar architectures, they have some key differences. ALBERT introduces factorized embedding parameterization and cross-layer parameter sharing, while the DistilBERT model is a student network trained with knowledge distillation from BERT. We use the weights and implementations of the models available in the Huggingface Transformers library [24].

3.3 Datasets

We performed experiments on three datasets to assess the performance of the proposed approach on different tasks. All of them pose classification problems with a different number of target categories and dataset sizes. **IMDB** [15] contains of a collection of 50,000 movie reviews and their associated binary sentiment polarity labels (either "positive" or "negative"), which is used to train a sentiment classifier. **Yelp F** [27] contains reviews of local businesses and their associated rating (1-5). The task is posed as a text

classification task, from the review text to its associated rating. **AG News**[4] consists of over one million categorized news articles gathered from more than 2,000 news sources. We used the common subset [27] of the whole dataset, consisting of 120,000 samples equally divided in four categories.

3.4 Experiments and hyperparameters

We construct several experiments to evaluate how well Federated Learning scales with an exponentially increasing number of clients. In all experiments, the respective dataset is divided into a number of subsets equal to the number of clients. Data points are uniformly sampled on each client (i.i.d.). Results with a single client are considered centralized training baselines for each model and dataset. We run the baselines for a fixed number of rounds based on our compute budget. The test set performance for the baselines are then compared against varying number of participating clients at the same number of rounds. Finally, since runs with a larger number of clients converge more slowly, we allow those runs to continue to a second threshold and report the number of rounds required to reach 90% of the baseline performance, similar to [16]. Runs not reaching 90% of the baseline performance within the second threshold are reported as failures. We run the baseline for 100 rounds for both IMDB and AG News while setting the second threshold to 200 rounds. However, we only run Yelp F baselines for 50 rounds due to its large size and set the second threshold at 100 rounds. Like [12], we avoid momentum, weight decay, and dynamic learning rates for simplicity. Instead, all experiments are performed with SGD. Based on [22] we choose a constant learning rate of $2 \cdot 10^{-5}$, a maximum sequence length of 128 and a batch size (B) of 32. Furthermore, the number of local epochs (E) is set to 2 per round.

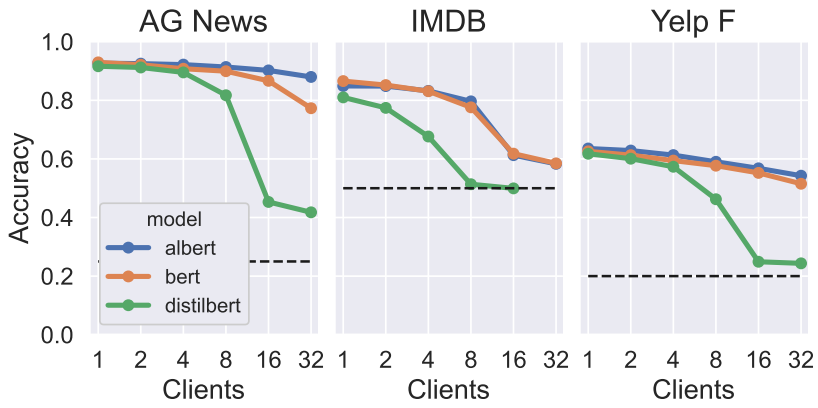

Fig. 1. Test accuracy at a fixed compute budget of 100 rounds for AG, IMDB, and 50 rounds for Yelp F. The expected accuracy of a random classifier for each task has been highlighted in the dashed line. Higher is better.

[4] http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

4 Results

4.1 Fixed compute budget

In Figure 1, we study the effect of increasing the number of clients. It shows the final accuracy after 100 rounds for IMDB and AG News, and 50 rounds for the much larger Yelp F., with an exponentially increasing number of clients. Both ALBERT and BERT are well behaved and exhibit a gradual decrease with an increasing number of clients. However, DistilBERT shows a much steeper decline when moving past 4 clients for all datasets, down to the random classifier baseline (IMDB, Yelp F).

4.2 Rounds until target performance

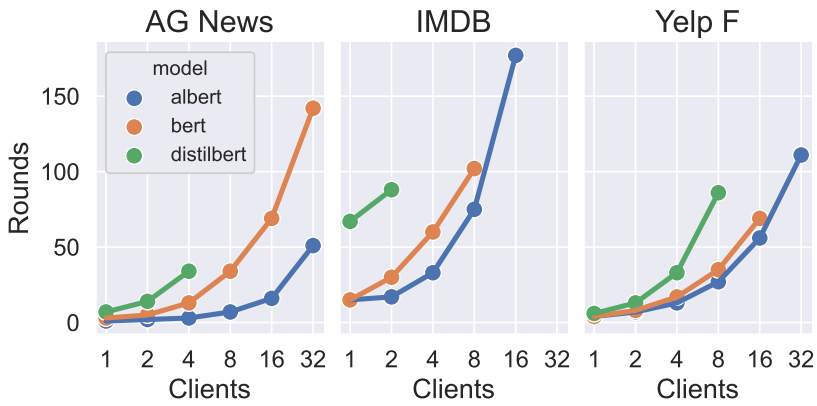

Fig. 2. Number of training rounds required to reach 90% of the non-federated baseline test accuracy. Omittions occur when the target is not reached in 100 (Yelp F) or 200 rounds (AG News, IMDB). Lower is better.

Examining the number of rounds necessary to achieve 90% of the non-federated baseline accuracy (Figure 2) yields a similar observation. While all models perform worse with more clients, ALBERT and BERT mostly reach the target accuracy within the allocated number of rounds until 32 clients are used. DistilBERT on the other is unable to reach the target accuracy at 16 clients for Yelp F, and as low as 4 clients for IMDB.

4.3 Dynamics of fine-tuning

The test accuracy during fine-tuning (Figure 3) allows a more complete understanding of how well FEDAVG scales for language model fine-tuning. While some scenarios (e.g. Yelp F. with BERT) show a gradual degradation with the number of clients, other

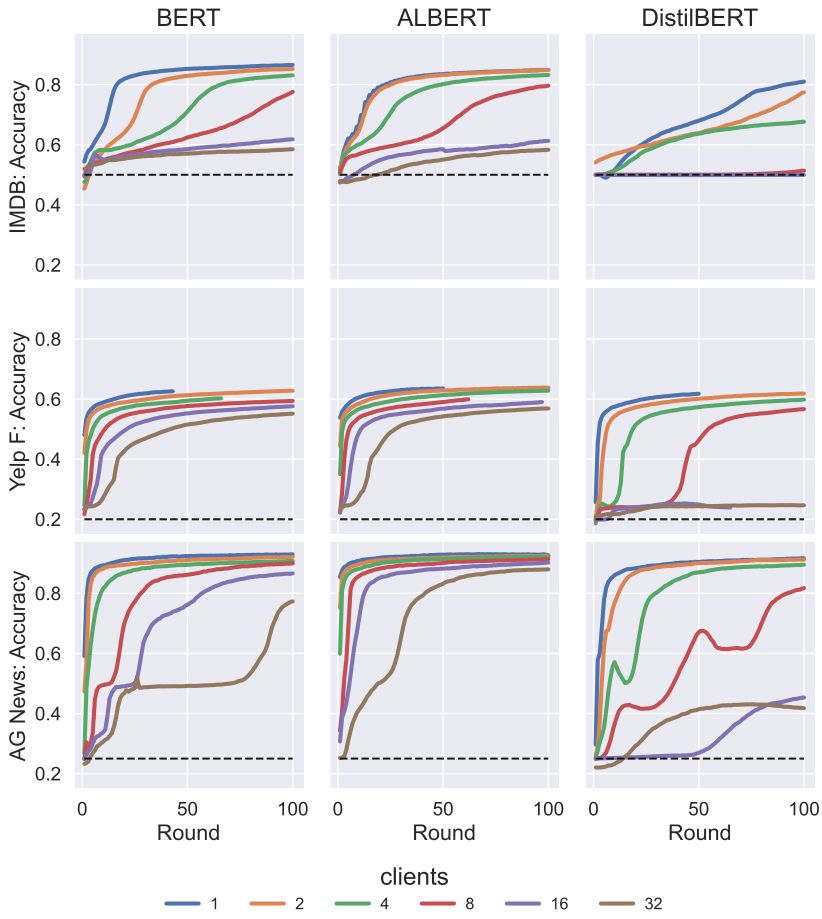

Fig. 3. Test accuracy (higher is better) over communication rounds for our scenarios. The random classifier baseline is shown as a dashed line.

configurations are more adversely affected by the increasing number of clients. In some instances the accuracy stays constant over a large period, sometimes even at the random classifier baseline for the whole (DistilBERT on IMDB) or part (DistilBERT on AG News) of the experiment when the number of clients is high.

5 Discussion

In this paper, we have evaluated the performance of Transformer-based language models fine-tuned in a federated setting. While BERT and ALBERT seem to learn each task quickly (Figure 3), DistilBERT has a much slower learning progression in the federated setup. A possible explanation is the process of distillation during pre-training, but further research is needed to fully understand the cause. We demonstrated that BERT

and ALBERT scale well up to 32 clients (Figure 1), but found a substantial drop in performance compared in DistilBERT compared to its own baseline. Furthermore, DistilBERT requires more rounds to achieve the same performance. Investigating these issues in training DistilBERT with FL may be a promising direction for future research. Conversely, these results indicate that FL can be sensitive to the number of clients, highlighting the importance of evaluating FL at different scales. In conclusion, we have demonstrated the applicability of the federated learning paradigm and evaluated it on a number of Transformer-based models up to 32 clients. Our findings show that the relatively large sizes of these models are generally not prohibitive for federated learning.

6 Acknowledgements

This work was funded by VINNOVA (grant 2019-05156). We would also like to thank AI Sweden and CGit for providing additional compute resources to this work.

References

1. Adhikari, A., Ram, A., Tang, R., Lin, J.: Docbert: Bert for document classification. arXiv preprint arXiv:1904.08398 (2019)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1). pp. 4171–4186 (2019)
3. Ge, S., Wu, F., Wu, C., Qi, T., Huang, Y., Xie, X.: Fedner: Medical named entity recognition with federated learning. arXiv preprint arXiv:2003.09288 (2020)
4. Guha, N., Talwalkar, A., Smith, V.: One-shot federated learning. arXiv preprint arXiv:1902.11175 (2019)
5. Hard, A., Rao, K., Mathews, R., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C., Ramage, D.: Federated learning for mobile keyboard prediction. CoRR **abs/1811.03604** (2018)
6. Hsu, T.M.H., Qi, H., Brown, M.: Measuring the effects of non-identical data distribution for federated visual classification. arXiv preprint arXiv:1909.06335 (2019)
7. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U., Suresh, A.T.: Scaffold: Stochastic controlled averaging for on-device federated learning. arXiv preprint arXiv:1910.06378 (2019)
8. Konečnỳ, J., McMahan, B., Ramage, D.: Federated optimization: Distributed optimization beyond the datacenter. arXiv preprint arXiv:1511.03575 (2015)
9. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations. In: International Conference on Learning Representations (2020)
10. Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C.H., Kang, J.: Biobert: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
11. Li, T., Sanjabi, M., Beirami, A., Smith, V.: Fair resource allocation in federated learning. arXiv preprint arXiv:1905.10497 (2019)
12. Lin, T., Kong, L., Stich, S.U., Jaggi, M.: Ensemble distillation for robust model fusion in federated learning. arXiv preprint arXiv:2006.07242 (2020)
13. Listo Zec, E., Mogren, O., Martinsson, J., Sütfeld, L.R., Gillblad, D.: Federated learning using a mixture of experts (2020)
14. Liu, D., Miller, T.: Federated pretraining and fine tuning of bert using clinical notes from multiple silos (2020)

15. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 142–150. Association for Computational Linguistics, Portland, Oregon, USA (June 2011)
16. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-Efficient Learning of Deep Networks from Decentralized Data. In: Singh, A., Zhu, J. (eds.) Proceedings of the 20th International Conference on Artificial Intelligence and Statistics. vol. 54, pp. 1273–1282. PMLR, Fort Lauderdale, FL, USA (20–22 Apr 2017)
17. McMahan, H.B., Ramage, D., Talwar, K., Zhang, L.: Learning differentially private recurrent language models. arXiv preprint arXiv:1710.06963 (2017)
18. Mohri, M., Sivek, G., Suresh, A.T.: Agnostic federated learning. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of Machine Learning Research. vol. 97, pp. 4615–4625. PMLR, Long Beach, California, USA (09–15 Jun 2019)
19. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019)
20. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In EMC^2 Workshop at NeurIPS 2019 (2019)
21. Sun, C., Huang, L., Qiu, X.: Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. arXiv preprint arXiv:1903.09588 (2019)
22. Sun, C., Qiu, X., Xu, Y., Huang, X.: How to fine-tune bert for text classification? In: Sun, M., Huang, X., Ji, H., Liu, Z., Liu, Y. (eds.) Chinese Computational Linguistics. pp. 194–206. Springer International Publishing, Cham (2019)
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
24. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T.L., Gugger, S., Drame, M., Lhoest, Q., Rush, A.M.: Huggingface's transformers: State-of-the-art natural language processing. ArXiv **abs/1910.03771** (2019)
25. Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., Li, M., Lin, J.: End-to-end open-domain question answering with bertserini. arXiv preprint arXiv:1902.01718 (2019)
26. Zeng, Z., Deng, Y., Li, X., Naumann, T., Luo, Y.: Natural language processing for ehr-based computational phenotyping. IEEE/ACM transactions on computational biology and bioinformatics **16**(1), 139–153 (2018)
27. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Cortes, C., Lawrence, N.D., Lee, D.D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems 28, pp. 649–657. Curran Associates, Inc. (2015)
28. Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., Chandra, V.: Federated learning with non-iid data. arXiv preprint arXiv:1806.00582 (2018)