Federated Learning of Unsegmented Chinese Text Recognition Model

Xinghua Zhu, Jianzong Wang*, Zhenhou Hong, Tian Xia and Jing Xiao
Ping An Technology (Shenzhen) Co., Ltd.
Shenzhen, China
Email: {zhuxinghua889, wangjianzong347, hongzhenhou168, xiatian443, xiaojing661}@pingan.com.cn

***Abstract*—Unsegmented text recognition is a crucial component in financial document processing systems. Financial text materials, such as receipts, transcripts, identification documents, etc. often involve critical personal information. In many circumstances, these data reside in protected servers of different institutions and must not be transferred beyond the institutional firewall. The emerging technology of Federated Learning (FL) provides a data-secure way of uniting isolated datasets in model training. Using the FL framework, text recognition models can be trained with larger collection of image samples.**

In previous works, federated text recognition models only deal with single-character images and alpha-numeric corpus. Such models are not competent in industrial applications, especially in Chinese text recognition problems. In this paper, we apply federated learning with a deep convolutional network to perform variable-length text string recognition with a large corpus.

In our experiments, we compared two prevalent federated learning frameworks, namely, Tensorflow Federated and PySyft. Results show that federated text recognition models can achieve similar or even higher accuracy than models trained on deep learning framework. On a 5-client distributed dataset, the best character accuracy is achieved by TFF at 49.20%. Extensive experiments are also conducted to evaluate the effect of distributed data storage over the performance of trained models. TFF again achieved a maximum character precision of 54.33% with non-distributed dataset.

***Index Terms*—Federated learning, federated optimization, deep learning, text recognition, optical character recognition.**

I. Introduction

Deep neural networks has seen tremendous growth in its popularity in various domains. Development of back propagation method [1] and batch-based training techniques enable virtually any neural network, given sufficient amount of data. However, this data-centric training strategy is facing ever-growing barrier due to privacy protection concerns. A new training technique, Federated Learning [2]–[4], has been proposed to unite data from protected internal servers or individual devices. In federated learning frameworks, data reside only in client devices. A central server coordinates the gradient computation and parameter update. In one way, federated learning can connects portable devices carrying user-specific data, with powerful computation servers, hosting a ubiquitous model, to enable adaptive prediction based on user behaviour [5], [6]. In another way, federated learning helps to break the data boundary between institutes, and empower a more accurate prediction model by aggregating a full set of available data. The latter application of federated learning is particularly crucial for handling financial data [7]. Such data often involve sensitive information that must not be shared or transferred outside internal servers. Federated learning does not require central storage of the raw data. Instead, the central server collects the encrypted gradients of model parameters for each update. It provides a way of joint model training without tampering data security.

As federated learning frameworks develop, they have been experimented and applied in more and more scenarios. In this paper, we focus on the application of federated learning in unsegmented Chinese text recognition. Text documents comprise a large a portion of financial data. A lot of them may contain critical information about participant identity or transaction details. Existing works on federated text recognition mostly deal with single-character images of a small alpha-numeric corpus. In this paper, we propose to train a deep convolutional network to perform unsegmented Chinese text recognition using the FL platform. The proposed federated model recognizes variable-length text strings with a corpus of 5,577 Chinese characters. Experiments showed that the federated learning framework efficiently improves the aggregated performance of the text recognition model, without sharing raw image data.

Contributions of this paper include:

1) Implementation of unsegmented text recognition algorithm on federated learning framework.
2) Evaluation of the impact of dataset size, number of nodes and load balance on model performance.
3) Comparison of text recognition models on TFF and PySyft frameworks.

II. Related Work

A. Federated Learning

In [3], the authors proposed a federated learning framework, where hand-held devices communicate to one central server to train a universal text entry prediction model. The encrypted gradient communication between client and server has been proposed back in [2]. The successful implementation of a privacy preserving way to utilizing client-side data initiated the accelerated researches on secure distributed machine learning. According to [8], federated learning methods can be

*Corresponding author: Jianzong Wang, jzwang@188.com. This paper is supported by National Key Research and Development Program of China under grant No. 2018YFB1003500, No. 2018YFB0204400 and No. 2017YFB1401202.

categorized into horizontal FL, vertical FL, and federated transfer learning by the form of data and feature overlap. Numerous open-source federated learning frameworks have also been published to facilitate experiments on expanded application domains. Examples include Tensorflow Federated [9], PySyft [10], FATE [11], etc. These platforms are mostly at the development stage and bug-prone. We experiment a deep text recognition model on two of the open-source FL frameworks, Tensorflow Federated and PySyft. The private version of TFF has been successfully applied in the text entry prediction model in [3]. PySyft is the only openly-available framework we found that supports GPU computation on convolutional neural networks. Therefore, the training and testing performances are compared on these two frameworks to demonstrate the capability of federated learning for text recognition models in our paper.

B. Text Recognition

Most contemporary text recognition models are deep neural networks. Numerous deep learning frameworks have been developed to construct and train such networks, such as Tensorflow, PyTorch, MXNet, etc. Models trained on these frameworks can achieve outstanding performance in many different tasks. But training on these frameworks requires centralized dataset, which might lead to data security and privacy violations.

Nowadays, prevalent text recognition models are recurrent [12]–[14]. However, recurrent units such as LSTM [15] and GRU [16] are not supported on some federated learning frameworks. In this paper, we resolve to a simpler structure for lexicon-free text recognition, as proposed in [17]. The network used in [17] is purely convolutional. Trained on Tensorflow framework, it was proved to be effective in recognizing alphanumeric strings.

To our knowledge, federated learning frameworks have only been experimented with single-digit recognition using machine learning methods, such as linear classification, or logistic regression [18]. In this paper, we explore the application of federated learning on a deep text recognition model. There are two fundamental distinctions between our experiments and the single-digit recognition example. First, our paper deals with variable-length text strings instead of single-character images. Second, the corpus size in our experiments is much larger than that in [18]. Therefore, the problem defined in this paper is much more complicated than the MNIST example. It cannot be solved with a simple linear classification.

III. Proposed Methods

A. Deep Learning in Federated Learning Frameworks

Open-source federated learning frameworks have been experimented with various machine learning algorithms, such as logistic regression, SVM, clustering, etc. These models do not have a deep network structure and do not require large amount of training data. Some of the private implementations of these frameworks, such as Tensorflow Federated, has also been deployed with deep neural networks [3]. However, the public versions of these frameworks face more problems in general deep models.

In this paper, we explore the application of two prevailing federated learning frameworks on a deep convolutional neural network. The experimented frameworks are namely Tensorflow Federated (TFF) and PySyft. TFF is based on the core components of Tensorflow, while PySyft on PyTorch. Compared to their deep learning counterpart, TFF and PySyft lacks a number of fundamental components, as listed in table I.

Due to these limits, deep learning models might not be directly usable on federated learning frameworks. Even if the model is usable, training takes a much longer time due to lack of GPU support.

The federated text recognition application in this paper falls under the "horizontal federated learning" category by the classification proposed in [8]. In horizontal federated learning, data sets share the same feature space but hold different samples. In our application, the feature space is homogeneous across different datasets, all being image crops of text strings. All participating clients contribute their segments of data to train a universal text recognition model. That is, the experimented model is a single-task federated learning system. In such a system, participating clients are typically assumed to be honest, while the server is considered honest-but-curious [19]. Therefore, the data clients cannot tamper the data security of one another. Only the central server itself might compromise the privacy of client data sets. Security protection against a malicious server can be provided in the way of encrypted communication from data client to central server.

The training process on a federated learning framework is as follows. First, parameter gradients are computed in each client server locally on its own dataset. Gradients are applied to local model copies batch by batch. After each epoch, the cumulated gradients on each client server are then encrypted and sent to the central server. Upon receiving all client gradients, the central server aggregates model updates without learning information about any participant dataset. Aggregated model updates are then broadcast back to client servers, where client servers update their respective local model with the decrypted gradients.

B. Network Structure

When designing a deep network on federated learning frameworks, we have to take their limitations into consideration. For text recognition models, recurrent networks are often used to boost recognition performance. However, given application scenarios under the federated learning framework, a recurrent network is too complicated to train.

In this paper, we use a purely convolutional neural network to predict text strings. The CNN structure is adopted from the character sequence prediction network of [17]. The structure of the CNN is depicted in Figure 1. RGB images are resized to 100×32, pixel values are normalized to $[-1, 1]$. The CNN classifier comprises of 4 convolutional layers and 2 fully connected layers. Each convolutional layer is followed by a

	Tensorflow Federated	PySyft
GPU support	Raises error on convolution and pooling layers	Limited to single GPU
Batch normalization	✓	Not implemented
Recurrent units	Raises error during backpropagation	Not available
Optimization methods	✓	Only supports momentum-free optimizers

TABLE I
TFF AND PYSYFT LIMITATIONS IN MODEL TRAINING. NUMEROUS IMPORTANT FUNCTIONALITIES IN DEEP NETWORK TRAINING ARE MISSING OR ERRONEOUS ON BOTH FRAMEWORKS.

Rectified Linear Unit (ReLU). Consecutively, the convolution kernels are 5×5, 5×5, 3×3 and 3×3, with filter sizes 64, 128, 256, 512, respectively. The first fully connected layer transform the flattened $4\times13\times512$ tensor into 4096-dimensional feature vector. The output fully connected layers transform the 4096-dimensional feature vector into character classification logits. In our experiments, the maximum length of text is 4 characters. Therefore, 4 output layers are defined in our CNN network, each comprising a 5,578-class prediction, among which the first 5,577 classes represent the frequently-used Chinese character corpus, while the 5,578-th class represents the null character.

The CNN classifier performs lexicon-free text recognition on the Chinese character corpus. Parameters in the CNN are optimized to minimize the aggregated negative log-likelihood of the character sequences, i.e.,

$$\mathcal{L}=\sum_{i=1}^{N}\sum_{k=1}^{4}\sum_{j=1}^{M}-p_{ij}^{(k)}\log\hat{p}_{ij}^{(k)}, \tag{1}$$

where N is the size of training dataset, M the total number of classes, $p_{ij}^{(k)}$ and $\hat{p}_{ij}^{(k)}$ are the probability that the k-th character of sample i being label j.

IV. EXPERIMENT RESULTS

In this section, we report our experimental details and results on two federated learning frameworks, and compare their performance with the Tensorflow deep learning framework, on an authentic dataset from industrial application.

A. ID name dataset

The experiments are conducted with a private authentic dataset. It comprises of 11,000 images of Chinese names cropped from Chinese citizen identity cards. Each text label consists of 2 to 4 Chinese characters. The corpus has 5,577 Chinese characters, with severe class imbalance in its lexicon. We separate the dataset into 10,000 training samples, and 1,000 testing samples. All images are resized to 32 x 100 without aspect ratio preservation. Model performance is evaluated with character prediction accuracy. No data augmentation technique is applied in all our experiments.

B. Experiment on Tensorflow

We first implemented the CNN text recognizer on Tensorflow to verify its viability and set the baseline for comparison. In this experiment, the training set is used as a whole entity. We use the Adadelta optimizer to train the model parameters. The convolutional weights are initialized with the Xavier initializer. Fully connected weights are initialized with a random normal initializer with mean 0.0 and standard deviation 0.001. The experiment is carried out on a work station with an NVIDIA Tesla P100 GPU with 16GB memory. The best character accuracy, 53.77%, is achieved at the 50-th epoch.

C. Experiment on Federated Learning Frameworks

On Tensorflow Federated and PySyft, we simulate distributed data by randomly separating the ID name dataset into client data subsets. During training, data batches are randomly sampled from each client set. On each epoch, based on a copy of the current federated model, a client iterates over its own subset of data, calculating the gradients and updating model parameters at each step. At the end of the epoch, cumulated updates of the parameters at each client are aggregated across all clients to refresh the federated model. Considering our experiments are to verify the performance of deep learning on federated learning frameworks, we do not apply any encryption on the aggregated information.

On TFF, the federated models are trained on CPU, because the GPU version of convolution and pooling is not working. The batch size is set to 8. All weights in convolution and fully connected layers are initialized with random normal initializer, while biases are initialized with zeros. Parameters are optimized with the Adadelta optimizer with initial learning rate 1.0. On PySyft, federated models are trained on a workstation with an NVIDIA Tesla P100 GPU with 16GB memory. The only available optimizer on PySyft is SGD, for which we set the learning rate to 0.01. The batch size is set to 64. All weights are initialized with the He initializer [20], while biases are initialized with zeros.

We experiment two schedules to evaluate the federated text recongnizer models. First, we fix the number of clients, while varying the number of samples per client from 200 to 2000. We run the federated optimization for at most 50 epochs. The results are depicted in Table II and Figure 2.

Figure 2 shows that, the character prediction accuracy with the number of samples in each client. In comparison, the models converge more smoothly on TFF than on PySyft. Also, the test accuracy on TFF is higher than that on PySyft. With 2,000 images per client, the test performance on TFF is as high as 44.90%, while the highest accuracy achieved by PySyft is only 42.30%. The adaptive step size in the Adadelta optimizer in TFF might be the main contributor to TFF's advantage over PySyft.

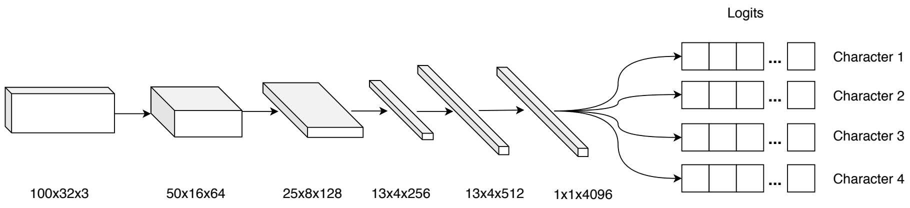

Fig. 1. Structure of the proposed neural networks.

Images per client	TFF			PySyft		
	Best Epoch	Best Accuracy	Avg. Time / Epoch	Best Epoch	Best Accuracy	Avg. Time / Epoch
200	48	31.25%	**1.0267hr**	13	31.00%	**0.2424hr**
600	50	34.33%	2.2042hr	37	32.08%	0.7394hr
1000	49	41.29%	3.2773hr	50	33.70%	1.1900hr
1400	48	43.75%	4.4693hr	50	36.64%	1.4223hr
2000	49	**49.20%**	6.0674hr	49	**41.30%**	2.0718hr

TABLE II
BEST ACCURACY USING 5 CLIENTS WITH VARYING NUMBER OF IMAGES PER CLIENT.

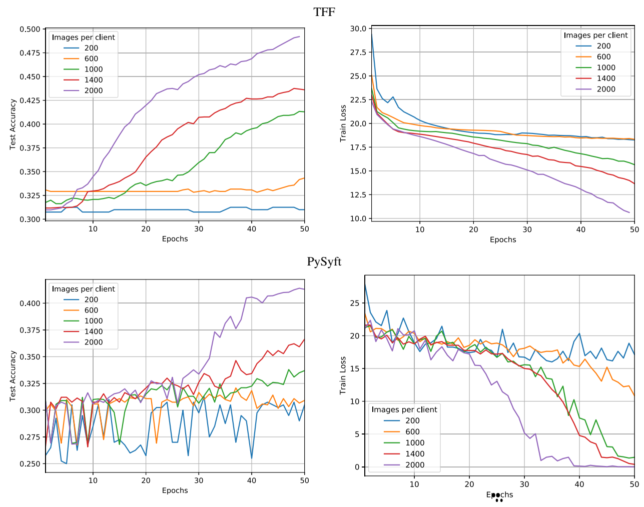

Fig. 2. Character accuracy and training loss with fixed number of clients.

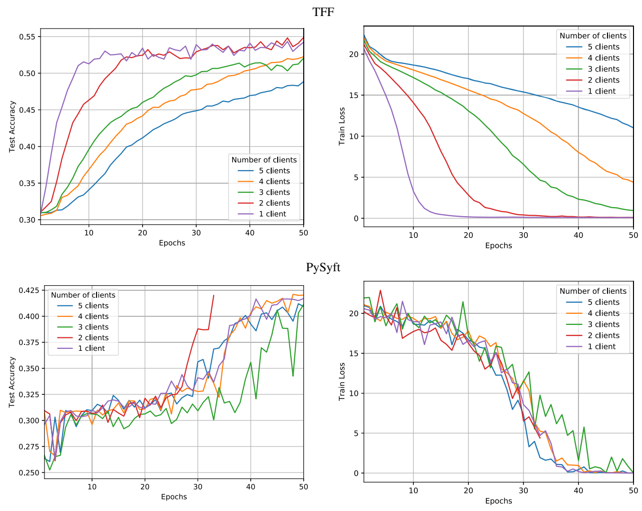

Fig. 3. Character accuracy and training loss with fixed total sample number.

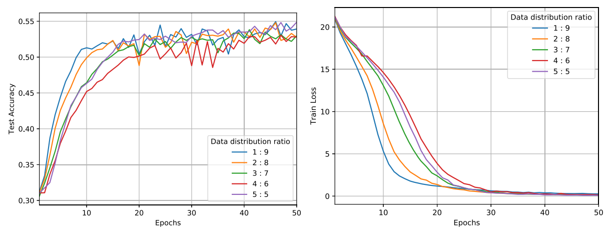

Fig. 4. Training process on imbalanced data distributions on TFF.

In the second schedule, we keep the total number of training samples unchanged, while equally splitting them into 1 to 5 clients. With the 1-client configuration, it is equivalent to training the model with non-distributed dataset. By this experiment, we want to show the possible loss of performance caused from distributing data storage across client servers.

On TFF, the performance of 1 client and 2 clients are similar, both reaching character accuracy above 53%. However, as the number of splits further increases, the prediction accuracy dropped noticeably. It is also worth noting that the convergence rate varies as the number of client splits increases. The larger number of clients, the slower the convergence becomes.

On PySyft, the difference between number of splits is less obvious. Still, the non-distributed experiment achieved highest accuracy compared to the distributed ones.

Experiments with unequal dataset sizes on TFF are presented in Figure 4. In these experiments, 10,000 training samples are randomly divided into two client datasets with ratios 1:9, 2:8, 3:7, 4:6, 5:5, respectively. As the training loss and test accuracy over epoch curves suggests, the performance at convergence has no obvious difference among different data distributions. However, the convergence rate is much faster with the 1:9 split than the 5:5 split. It suggests that given

Number of clients	TFF			PySyft		
	Best Epoch	Best Accuracy	Avg. Time / Epoch	Best Epoch	Best Accuracy	Avg. Time / Epoch
5	50	48.85%	5.4886hr	49	41.20%	1.7360hr
4	50	52.25%	5.4011hr	48	**42.10%**	1.0504hr
3	50	52.05%	5.3045hr	50	41.15%	**1.0368hr**
2	50	**54.88%**	5.2156hr	33	41.99%	1.2763hr
1	47	54.33%	**5.1546hr**	50	41.83%	1.4789hr

TABLE III
BEST ACCURACY USING 10,000 TRAINING SAMPLES DIVIDED INTO VARYING NUMBER OF CLIENTS.

identically distributed datasets, the federated training process tend to be dominated by the largest client dataset.

V. CONCLUSION AND FUTURE WORK

In this paper, we experimented a deep learning network on federated learning frameworks. Experiments are carried out on authentic text samples from industrial applications. Federated learning frameworks have a number of fundamental drawbacks that have to be addressed before it can be successfully applied with deep neural networks. We demonstrate the potential of applying a federated learning approach on protected text image data. By aggregating datasets from various sources, we can effectively improve prediction accuracy on text strings.

Federated learning is a promising technology, that erases the data boundaries between institutes and companies. With a privacy preserving federated system, they may enable training of united models that benefit from broad combination of data.

In its current stage, public federated learning frameworks still suffer from series of errors and lacking features for deep model training. For one thing, FL frameworks should fulfill the implementation with GPU support, so as to harness the computation power of fast-increasing computation power of GPUs. Secondly, in current configurations, the workload of a federated learning system mainly lies on client servers, where iterative gradient computation and model update are conducted. Thirdly, most current researches in the federated learning literature focus on machine learning methods, such as SVM, logistic regression and linear regression, etc. It would be more interesting to see how federated learning enables training of new models that was hindered by the lack of data collection.

ACKNOWLEDGEMENT

This paper is supported by National Key Research and Development Program of China under grant No. 2018YFB1003500, No. 2018YFB0204400 and No. 2017YFB1401202. Corresponding author is Jianzong Wang (jzwang@188.com) from Ping An Technology (Shenzhen) Co., Ltd.

REFERENCES

[1] D. E. Rumelhart, G. E. Hinton, R. J. Williams *et al.*, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.
[2] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. ACM, 2015, pp. 1310–1321.
[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," *arXiv preprint arXiv:1602.05629*, 2016.
[4] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 5, pp. 1333–1345, 2018.
[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for federated learning on user-held data," *arXiv preprint arXiv:1611.04482*, 2016.
[6] Y. Dong, J. Cheng, M. Hossain, V. Leung *et al.*, "Secure distributed on-device learning networks with byzantine adversaries," *arXiv preprint arXiv:1906.00887*, 2019.
[7] A. Ghosh, J. Hong, D. Yin, and K. Ramchandran, "Robust federated learning in a heterogeneous environment," *arXiv preprint arXiv:1906.06629*, 2019.
[8] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, p. 12, 2019.
[9] Google, "Tensorflow federated," 2019. [Online]. Available: https://www.tensorflow.org/federated
[10] OpenMined, "Pysyft," 2019. [Online]. Available: https://github.com/OpenMined
[11] W. B. F. Tech, "Fate," 2019. [Online]. Available: https://github.com/WeBankFinTech/FATE
[12] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 11, pp. 2298–2304, 2017.
[13] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 369–376.
[14] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4835–4839.
[15] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[16] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.
[17] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Synthetic data and artificial neural networks for natural scene text recognition," *arXiv preprint arXiv:1406.2227*, 2014.
[18] Google, "Federated learning for image classification," 2019. [Online]. Available: https://www.tensorflow.org/federated/tutorials/federated_learning_for_image_classification
[19] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 1175–1191.
[20] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.