Federated Learning with Digital Gateway: Methodologies, Tools and Applications

Yang Liu[1,2] and **Mingxin Chen**[1,2,3], **Ruiyuan Li**[1,2], **Yanmeng Lu**[1], **Changbin Lu**[1], **Jiandong Gao**[1,2], **Junbo Zhang**[1,2*], **Yu Zheng**[1,2]

[1]JD Intelligent Cities Research, Beijing, China

[2]JD Intelligent Cities Business Unit, Beijing, China

[3]Southwest Jiaotong University, Chengdu, China

{liuyang.jdd, chenmingxin7, zhang.junbo}@jd.com

Abstract

Federated Machine Learning (FML) is focused on training distributed models where data are scattered in different places and not centralized, such that data privacy and security are not compromised during the model training. The FML is naturally suitable to solve machine learning problems with *data island* situation. Motivated by taking it closer to solve real-life tasks, here we introduce an intelligent architecture, termed *Digital Gateway*. It is designed to serve the federated machine learning jobs and is able to transfer academic achievements easily into usable and deployable commercial products. It provides a software development kit (SDK) for secure communication between different parties in the federated modeling and contains different modules such as database interface, authentication center, account system, and user interface. This architecture has been shown to function properly in three real-world applications. Overall, the digital gateway is highly practical and deployable for applying federated learning to solve real-life tasks.

1 Introduction

When solving urban computing problems, such as credit scoring, urban anomalies detection and city traffic flow prediction, we have to face the fact that most data exist in the form of *data island* and scattered in different organizations not centralized. For companies, data are considered as digital assets and usually not shared. For governments, data are highly secured and in most cases not utilized. Data privacy and security is always a sensitive and important topic. Cases like the data breach at Facebook will heavily impact a company's business operations. Any leakage of governments' data can cause serious social problems. Because of the above reasons, when solving urban computing problems it is important to carefully design the joint-models which can connect the isolated data together and preserve the data privacy as well.

However, this is never an easy job. First, we must comply with the laws and regulations. The European Union passed the General Data Protection Regulation (GDPR) in April 2016 and enacted it in 2018. Under the GDPR, individuals can take more control over their personal data, and strict principles and absolute transparencies on usage of personal data were stated. Moreover, any intention and plan for the data must be authorized by the customers. Similarly, the state of California will enact California Consumer Privacy Act in January 2020 and the China Cyber Security Law has already been enforced since June 2017. Second, we face the complicated cross-domain scenarios. One reason why we have data island is that different companies and government organizations have different database architectures, different data protection protocols, different data types and standards. It is a difficult job to satisfy the various circumstances. Third, any modeling work involving different parties should follow several principles. When modeling, the data privacy and network security must be protected. The model itself needs to be lossless and the whole process should be efficient in both communication and computation. Model interpretability also needs be considered since all parties have the rights to know what happened during modeling.

Recently, federated machine learning (FML) has provided a new perspective to overcome the above challenges. It was first proposed by Google in 2016 [McMahan *et al.*, 2016][Konečnỳ *et al.*, 2016b][Konecnỳ *et al.*, 2016a], with the goal of training machine learning models with data distributed in different places, while keeping the users' data secured and ensuring privacy is not compromised. In their original papers, FML was applied for predictions of mobile device keyboard input. Following their work, researchers have extended the methods to solve many other practical problems as shown in Figure 1, with successful examples including [Hardy *et al.*, 2017][Chen *et al.*, 2018] [Cheng *et al.*, 2019][Liu *et al.*, 2018b][Smith *et al.*, 2017]. A detailed survey of FML can be found in [Yang *et al.*, 2019].

FML methods are capable to solve many urban computing problems, but when we try to develop practical products based on the available research work, we realized there are many other pieces missing from the whole puzzle. For example, how to develop the architecture such that algorithm engineers can be more focused on the modeling instead of worrying the implementation of communication between different parties. In order to satisfy our developmental needs, we designed an intelligent architecture, called *Digital Gateway*, which can easily transfer the research work of federated

*Junbo Zhang is the corresponding author.

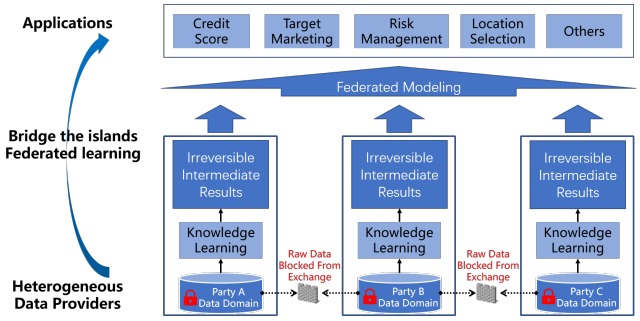

Figure 1: Federated Machine Learning

machine learning into deployable products. It is still in early stage and not all problems are solved. Our contributions are four-fold:

- A comprehensive architecture designed for easy deployment of FML in real-life tasks, and that bridges the gap between data island and practical applications.
- An end-to-end solution that assembles the overall modeling process into one pipeline, from reading the data at different organizations to making the final inference with trained FML models.
- Data privacy and security are protected with various components which to ensure no raw data will be leaked.
- Digital Gateway has been shown to function properly in three real world cases with different organizations.

2 Architecture Design

2.1 Motivations

In the work of [Hardy *et al.*, 2017], they proposed the federated logistic regression (FLR) for problems with data under vertical federated settings. Their work was clearly explained with enough information for reproducibility and we implemented our prototype of FLR based on it. But when we tried to apply it for real-life applications, we realized it is not an easy job. The most important thing here is the communication tool. During model training, the intermediate values will be frequently exchanged. Without proper tools, we would be struggling with how information are communicated between different parties, instead of the FML algorithm design. Ideally, the actual codes for communication should be concise and it will be better if specific communication logic can be supported, such as broadcast, scatter, gather and reduce. There are many other components missing too. For example, we must have a security protocol which can verify the identity of each party that joins in the modeling process, and each party has the right to know the exact amount of information exchanged during the modeling process, and how we can deal with different database environments. Given the challenges, an underlying architecture which can fully support the federated modeling work is necessary and important.

Based on the current situation, we concluded that the digital gateway should support at least four basic functions, which are connection, sharing, security, and monitoring. In the connection function, registration, domain name resolution (DNS), authentication and other related ones need to be considered. For the function of sharing, first it needs to support sharing of meta data. When multiple parties want to train federated learning models together, at the beginning they have to know what kind of data other platforms can provide. During training, intermediate values of models should be passed easily, freely and securely to each other. Besides, it also needs to support some necessary data exchange with large size, for example the exchange of encrypted sample ID in the vertical federated modeling for ID alignment. As for function of security, different data privacy and security protocols should be supported, as well as other tools such as firewall. The data exchanged between different parties follow the principles of confidentiality, completeness, non-deniablity, and freshness. Last, the entire modeling process must be monitored and functions including traffic monitoring, log analysis, and expenses settlement should be supported.

2.2 Overview

Based on the requirements of practical deployments, we designed the digital gateway to unify all the different components into one comprehensive system as shown in Figure 2. In our work, we have divided the digital gateway infrastructure into mainly three parts, applications, federated models, and backend components including communication layer, persistence layer, account system and authentication center. In practical tasks, the digital gateway must be deployed in all the data providers' servers. Only in this way can the model training be conducted using the same security protocol for all participants; it also keeps the federated modeling synchronized in the same pace without any conflicts. The detailed description of the architecture is given below.

2.3 Architecture Details

Persistence Layer. The persistence layer is implemented for data management, and the two major components in it are the Apache HBase and the Remote Dictionary Server (Redis). Apache HBase is one part of the Apache Hadoop ecosystem and designed for hosting very large tables with real-time read and write access to the big data. It is a distributed and non-relational database which runs on top of HDFS, and robust to different data types and data structure. When running federated learning tasks, other modules in Hadoop are also used. Data are recommended to be store in HDFS and Hive is used for querying data. All the jobs are scheduled by Azkaban. Redis is an in-memory key-value database which supports various data types and corresponding operations. In Redis all data are stored in high-speed memory and the high performance of it can make up the disadvantages of relational database, i.e., Apache HBase. In the federated modeling process, the intermediate model values such as gradients and loss are frequently exchanged. In our setup, Redis works as the mailbox where both sending and receiving messages will be placed in a separate queue waiting for models to use.

Account System and Authentication Center. These two modules are established for modeling management and access control. When deploying federated learning for practical tasks, there must exists at least two different data providers

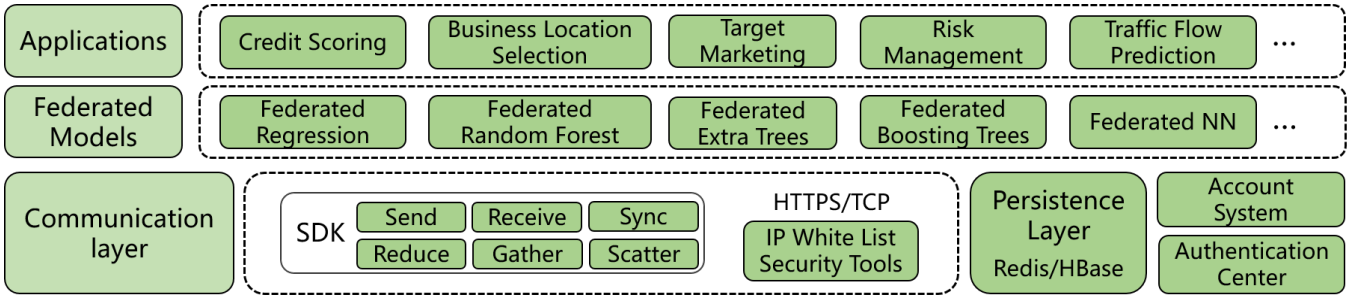

Figure 2: Digital Gateway Architecture

working together. Because of this, an account system is necessary to manage the data providers. In our setup, it is responsible for user registration, management of login, authorization, payment, and other affairs. The authentication center will assign a certificate to newly registered accounts and keep monitoring during the modeling. Although at the algorithm level, raw data will not be directly shared between different parties, it is still necessary to check the certificate for any user logged into the account. If one company or organization wants to start a federated modeling task, they must log into the account, which will run a security check for the identification of the user. And when the federated learning job starts, if the selected model follows a parameter server approach, the authentication center also takes the responsibility as the parameter server or master server.

Communication Layer. The communication layer plays an important role in this architecture. It contains the software development kit (SDK) to support the secure communication between different data providers. The SDK is developed to provide similar functions as MPI [Barney, 2019]. As in distributed machine learning, communication logic such as send, receive, reduce, gather is frequently used. However, available packages like MPI and Kafka are not suitable for our business demands. Kafka follows a data streaming mechanism and cannot support specific logic, especially *allreduce* or *ringreduce*. Besides, all the information exchanged should be encrypted before sending to the other parties when passing through public network, causing the problem that different communication methods have different secure protocols which are hard to unify. So we developed our own SDK, which supports various communication methods, and at the same time ensures that all information exchanged is encrypted and decrypted following an universal method. The SDK provides an exposed interface for message communication, and under it there is a API gateway to finish the job. When one data provider is working on the federated modeling task, it will keep sending service request to the API gateway through the SDK. If the request is an internal service, such as data pre-processing, the API gateway will call corresponding services or modules by using internal service mapping. If an external service of the API gateway is requested by the data provider, for example user identity verification, it will operate through the external service mapping. In our work, all communication is processed through HTTPS or TCP. In contrast to the blacklist that contains users whom should be blocked, the IP whitelist contains users that should be authorized and not considered as threats. Any connection from users not in the list is rejected. The security tools are implemented through both software and hardware techniques. At the software level, encryption methods such as RSA and AES, token assignment and management, digital signature and other tools are integrated. At the hardware level, a bastion host or jump host is used one or more times to provide extra protection. The bastion host mechanism can give data providers more control over the information resource access, fulfilling the requirements for operation and management. It also generates detailed operation records and auditing report. Based on these advantages, data privacy and security can be further protected.

Federated Models and Applications. In this architecture, the backend components were developed to serve researchers so they could focus on the algorithm design. Corresponding applications were developed based on available models, and are open to general users so FML methods can be used directly for business purpose. We also developed the user interface module for interactive display and monitoring module for flow tracking.

3 Tools and Workflow

In this part we will present more details about the communication between different participants and modeling workflow.

3.1 Usage of SDK

The SDK is developed to provide various communication methods. When programming we can directly call the exposed functions in SDK to send and receive messages. Under the SDK is our API gateway. The API gateway is responsible for service mapping and all communications are passed through URL requests. For every server where the digital gateway is deployed, list of URLs associated with the SDK functions will be automatically generated. Each call of the SDK function will request the corresponding URL with POST method.

We present an example involving sending messages. The corresponding function in SDK is named *sendData*. If we want to send data from one client to another, the request URL will be in the format of

```
https://xxx.com/digital/gateway/api/sendData
```

For different functions, the associated parameters will be different. One common parameter is *ModelCode*. In practice,

different participants may have multiple separated federated learning jobs running at the same time, so for each job a specific *ModelCode* must be assigned to ensure synchronization between the participants. As for *sendData* function, the other parameters are *source*, *tag*, *data* and *destCode*. The *source* means where the data is from. The *tag* represents the message tag, such that the receiver will know which message to retrieve. The *destCode* is the code of the destination. All the functions are placed under the *Gateway* class and the initialization is required at the beginning of modeling. The example code of the *sendData* function in Python is given below:

```
from GatewayUtil import Gateway
gateway = Gateway(ClientCode)
gateway.sendData(ModelCode = ModelCode,
                 source = source,
                 tag = tag,
                 data = dataSend,
                 destCode = destCode)
```

Correspondingly, the receiving platform has to run a receiving operation, which will look like this:

```
received_data = gateway.getData(
                ModelCode = ModelCode,
                source = source,
                tag = tag)
```

Other frequently used communication methods are also developed, such as *gather*, *scatter*, *reduce* and *allreduce*. The SDK will keep checking if each communication is successful, otherwise a warning will be given for debug purpose.

Figure 3 is the workflow of the FLR method from [Hardy *et al.*, 2017]. Here data are vertically distributed, with two data providers (client A and B) that each has different feature space but same sample space. One parameter server is also deployed. The exchanged information include homomorphic encrypted $\hat{z}_A = W_A^T X_A$ and $\hat{z}_B = W_B^T X_B$, encrypted and decrypted gradients and loss of A and B, where W_A^T and W_B^T are distributed model weights. For communication of encrypted $\hat{z}_A$ and $\hat{z}_B$, we can use *send* and *receive*, or use *allreduce* with *sum* operator. For gradients $Grad_A$, $Grad_B$ and losses $\mathcal{L}_A$, $\mathcal{L}_B$, *send* and *receive* can do the job.

Algorithm 1 presents how data samples are split on a tree node in the federated decision tree. The data in this model are still vertically distributed in A and B. In FML, since our goal is to train a global model with distributed data, then in this case after the split, the sample IDs fell into the left and right subtrees are exactly same on both clients. As for communication, logic as *send*, *receive*, *gather* and *reduce* are used.

3.2 Network Topology

When cooperating with government organizations, they have strongest requirements for data security. Most times their data are stored on the servers located inside the government internal network. If a federated modeling task is started, the exchanged model intermediate values are first encrypted and transferred to the server located in the government external network through the first bastion host, then transferred again to the server located in the public network by another bastion host, then it can be sent to the other parties in the federated modeling process. During the two times' jump by the bastion hosts, all the operations are protected and recorded, and every movement needs be tracked and the other parties cannot visit the database directly. For simpler network situations, the topology is shown in Figure 4.

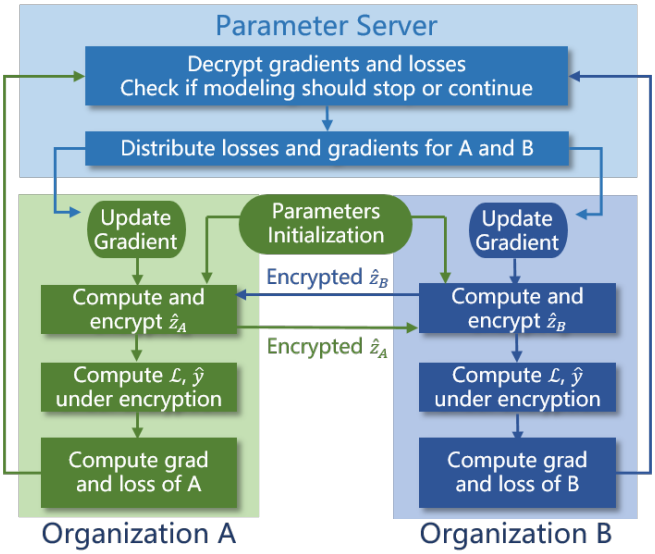

Figure 3: Secure Federated Logistic Regression

Algorithm 1: Splitter of Federated Decision Tree Node

Input : Client A with data $\mathcal{D}_A$ and feature space $\mathcal{F}_A$;
Client B with data $\mathcal{D}_B$ and feature space $\mathcal{F}_B$;
Output: Sample IDs of left and right subtrees
Function `Splitter` $(\mathcal{D}_A, \mathcal{F}_A, \mathcal{D}_B, \mathcal{F}_B)$
  Client A and B each finds local best split feature f_A and f_B, corresponding threshold th_A and th_B, and impurity improvement ii_A and ii_B;
  Master *gathers* f_A and f_B from client A and B;
  Master *reduces* ii_A and ii_B with *max* operation from client A and B;
  Master finds corresponding best split feature f^* from f_A and f_B;
  // Assume f^* is f_B and from client B;
  Master *sends* split flag to Client B;
  Client B *receives* split flag, use th_B to find sample IDs S_{left} and S_{right}, which will go to the left and right subtrees;
  Client B *sends* S_{left} and S_{right} to client A;
  Client A *receives* S_{left} and S_{right};
Return S_{left} and S_{right} on both client A and B;

3.3 Modeling Workflow

With the digital gateway, the whole federated modeling process can be divided into several steps, which is shown in Figure 5 and described below: Any company or government organization must first register in the account system, and make their digital gateway discoverable by others, at which point we call it a participant organization. At the same time, the owner of the joined organization can create different user groups with different levels of operating permissions.

Each new participant organization chooses the data tables which are open for FML jobs. For each table the participant can define which specific participants have access privileges.

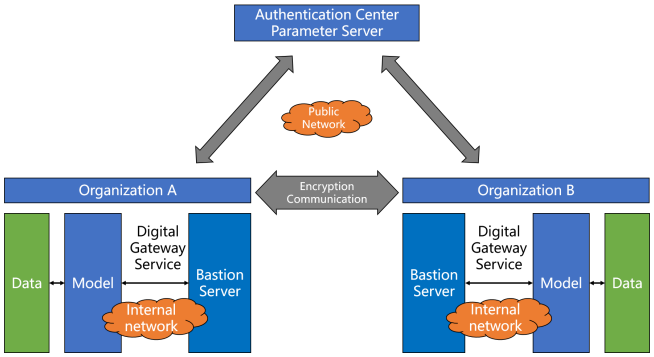

Figure 4: Network Topology

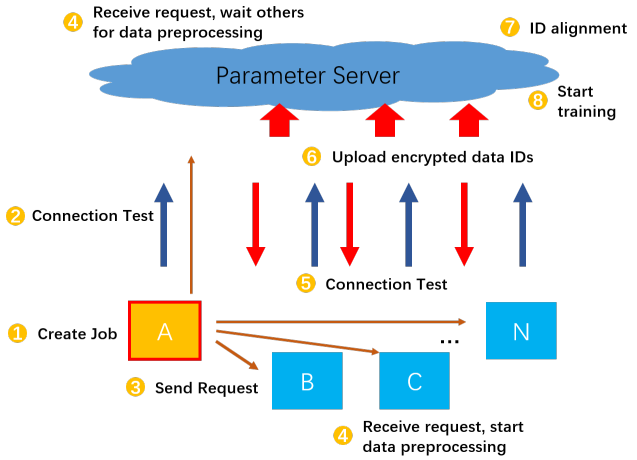

Figure 5: Federated Learning Workflow

Then the organization submit the meta-info of data tables to the digital gateway.

If one participant wants to initiate a federated learning task, it first checks the network connection and chooses what kind of models it needs to use, e.g., credit scoring model or business location selection model. Then the participant chooses one or multiple partners from the available participants. The initiator chooses what data to use from each partner. After that, the initiator sets the model parameters, and sends federated modeling invitation with all the configurations above to other partners.

For the participants who received invitations, they will decide to accept or not. If one of them rejects, then the task fails. If the task accepted by all, all participants will start to run data pre-processing scripts, including secure data ID matching if the task is for a vertical federated learning problem, as shown in step 6 and 7 of Figure 5. The data ID matching will be processed with different security protocols such that no real identity will be leaked. Besides, all participants will test its network connection with the parameter server (if needed by the model) and each of the participants.

After data from all participants are ready, the model training will be automatically started. During the whole process, the digital gateway will monitor every step, generate logs, and keep running security checks.

If every participant has a copy of the whole model, then they can use it at any time they want. But if it is a vertical federated model, then in the prediction stage the initiator has to run a similar process to connect with each partner as mentioned above to achieve the final predictions.

4 Experiments

In this section, we tested FLR and federated random forest (FRF) with different data sets from UCI [Dua and Graff, 2017], as shown in Table 1 and 2. For FLR, binary classification problems were tested and for FRF we added multi-class classification problems. In all tests each data set was vertically and randomly segmented on feature space and placed on two different servers as clients, each of which contains half of the feature space from original data. The labels were copied to each client server and a parameter server was also deployed to control the overall modeling process, without any raw data on it. All three servers were deployed with the digital gateway and connected to each other through public internet. The purpose of the experiments is to simulate and verify that our digital gateway is capable of conducting federated modeling for real world tasks and that the lossless criterion can be satisfied.

Data Set	Size	Features	Accuracy		F-1 Score	
			LR	FLR	LR	FLR
BC	569	30	0.8078	0.8078	0.8011	0.8011
spambase	4601	57	0.8830	0.8830	0.9167	0.9167
internet-ads	3279	1558	0.9255	0.9255	0.9561	0.9561

Table 1: Experiments of Federated Logistic Regression

Data Set	Size	Features	Classes	RF	**FRF**
adult	32561	14	2	0.844 ± 0.017	0.842 ± 0.037
gene	801	20531	5	0.981 ± 0.003	0.981 ± 0.007
waveform	5000	21	3	0.830 ± 0.010	0.833 ± 0.011

Table 2: Experiments of Federated Random Forest

As shown in Table 1, the FLR model achieved the same results compared with regular logistic regression. In Table 2, accuracy of all tests was given and we can find RFR's results are at the same level as RF. Overall, in all tests the lossless criterion was satisfied and there was no precision loss during multiple rounds of encrypted communication. The digital gateway is shown to be functioning properly and served the federated modeling tasks very well.

5 Applications

In this part we present three practical cases where the digital gateway and FML methods can be applied for better business outcomes.

5.1 Credit Evaluation

For companies and individual persons, credit score is important. However, it is difficult to build a fair credit evaluation model since data are not shared between companies and government organizations. In our work we implemented a credit evaluation model which applies FML methods with data provided by different organizations. The digital gateway was deployed for each participant for secure communication. This model can be separated into three parts. The

first part is data pre-processing and feature engineering. The global feature space can be divided into several dimensions, and usually each data provider contributes one or two segments. The sample IDs are also aligned between each party in this step. The second part is bootstrap with logistic regression. Here we trained multiple FLR models from the work of [Hardy *et al.*, 2017] as mentioned in last section. In each FLR model, a probability of debt default was given and the loss L was calculated with cross-entropy function. Then the overall loss F of the ensemble model can be expressed as $F = \sum_{i=1}^{n}(beta_i \times L_i)$. Here n is the number of FLR models and $beta_i$ is the sub-model weight. The last part is the score calculation. The modeling outputs are transformed through self-designed score functions and a final credit score is given. Compared with real world debt default's data, our modeling outputs followed an similar statistical distribution.

5.2 Target Marketing

Target marketing is another example of where we can apply federated learning for practical applications. For example, if one company wants to promote its products to potential customers, in the past the company can only rely on its own data and plan the marketing strategy correspondingly. With federated learning methods, data from more fields can be utilized and better results could be achieved. In our implementation, we tried a different approach. First we deployed the digital gateway to both parties. We ran the data pre-processing and aligned the sample IDs. After that, we trained independent deep neural networks (DNN) on each party's data, and a higher level presentation of data was generated. We also kept the sample IDs for later alignment. This step is not reversible and the data privacy was further secured. Then we ran a LR model with both parties' outputs and gave the final predictions. The DNN may lower the performance of the overall modeling result, but it effectively protects the data privacy so it is still suitable for applications like target marketing.

5.3 Business Location Selection

Business location selection is often a difficult question for retail companies. With FML, this issue may be eased. In this case, we partnered with a retail company and trained a federated model together to help them evaluate current stores and where to open new stores. The data used for training were business secrets and highly sensitive. In our work, both companies divided the target city's map into multiple grids with an agreed rule, and uniformly assigned an ID to each grid. Based on grids, both companies generated corresponding sample values for each grid and prepared the data for training. In the modeling process, all the communications between two companies were processed by the digital gateway through a secure connection. The final outcomes were sent back to their company and a web page based interactive visualization was generated. Based on it, we can have a better understanding of the city and know where new locations are recommended. In our work, there are five different indicators to decide the score of each grid in five different dimensions. For each dimension, a corresponding model is trained. We have deployed the digital gateway in the partner's servers to help us with the whole training process. Then all we need to do is to run the model training process and achieve the final predictions for each index. Each achieved index is shared by both parties and used for the final prediction of the best business location. In the whole modeling process, each company's data privacy is fully secured, no business secrets are leaked, and the final predictions can be used by both parties to make business decisions.

6 Related Work

By far most work have been done in the area of Secure Multi-party Computation (SMC), which involves multiple data providers and secret sharing is desired during the modeling process. FML can be considered as a combination of machine learning and SMC. In particular, the vertical federated learning follows an almost identical approach as SMC, where a semi-honest third party (STP) is often deployed and the model parameters are scattered in different parties, with examples including [Bogdanov *et al.*, 2008], [Mohassel and Rindal, 2018], [Bonawitz *et al.*, 2017] and [Volgushev *et al.*, 2019]. Distributed Machine learning (DML) is another popular topic which is closely related to FML. As summarized in [Liu *et al.*, 2018a], currently there are mainly three types of DML architectures. *MapReduce* [Dean and Ghemawat, 2008] is the first one in which the data are processed distributively in the *Map* procedure and the results are synchronized and reduced in the *Reduce* stage. The second is the *parameter server* where most of the computations are distributed to different client servers and a master server is responsible for synchronizing the overall modeling process. Most FML methods follow this approach and examples include [Bonawitz *et al.*, 2019], [Konečnỳ *et al.*, 2016b] and [Cheng *et al.*, 2019]. The third one is the data stream based architecture, and a typical example is TensorFlow [Abadi *et al.*, 2016].

7 Discussion and Future Work

When deploying federated learning into real world tasks, we often face many unexpected situations, such as outdated servers, small network bandwidth and various network deployments from companies and governments. For methods like [Hardy *et al.*, 2017], the amount of information exchanged between different participants is small and mostly includes encrypted loss and gradients. But for deep learning methods like what Google used in their keyboard typing predictions or federated medical image studies, the information transferred between parameter server and clients can exceed the maximum capabilities of either software or hardware. We are interested in how to reduce the communication cost and improve the efficiency. Another important topic is security. We face many difficulties when coupling security methods with machine learning. Homomorphic encryption supports encrypted linear operation, but does not fit non-linear functions such as Sigmoid or Logarithmic function. Approximation of the non-linear functions can introduce large noise to the model. Besides, homomorphic and similar encryption methods are time consuming. Differential privacy is an alternative approach, but it cannot fully guarantee lossless criterion, which is critical to FML. New methods need to be studied for better practical applications.

References

[Abadi *et al.*, 2016] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pages 265–283, 2016.

[Barney, 2019] Blaise Barney. Message passing interface (mpi). lawrence livermore national laboratory. Available at https://computing.llnl.gov/tutorials/mpi, 2019.

[Bogdanov *et al.*, 2008] Dan Bogdanov, Sven Laur, and Jan Willemson. Sharemind: A framework for fast privacy-preserving computations. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5283 LNCS:192–206, 2008.

[Bonawitz *et al.*, 2017] Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy preserving machine learning. Cryptology ePrint Archive, Report 2017/281, 2017. https://eprint.iacr.org/2017/281.

[Bonawitz *et al.*, 2019] Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, and Jason Roselander. Towards Federated Learning at Scale: System Design. *SysML*, 2019.

[Chen *et al.*, 2018] Fei Chen, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning for recommendation. *arXiv: Learning*, 2018.

[Cheng *et al.*, 2019] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, and Qiang Yang. Secureboost: A lossless federated learning framework. *arXiv preprint arXiv:1901.08755*, 2019.

[Dean and Ghemawat, 2008] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[Dua and Graff, 2017] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[Hardy *et al.*, 2017] Stephen James Hardy, Wilko Henecka, Hamish Iveylaw, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv: Learning*, 2017.

[Konecnỳ *et al.*, 2016a] Jakub Konecnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[Konečnỳ *et al.*, 2016b] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[Liu *et al.*, 2018a] Tie-Yan Liu, Wei Chen, Taifeng Wang, and Fei Gao. *Distributed Machine Learning, Theories, Algorithms, and Systems*. China Machine Press, 2018.

[Liu *et al.*, 2018b] Yang Liu, Tianjian Chen, and Qiang Yang. Secure federated transfer learning. *CoRR*, abs/1812.03337, 2018.

[McMahan *et al.*, 2016] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*, 2016.

[Mohassel and Rindal, 2018] Payman Mohassel and Peter Rindal. ABY 3: A Mixed Protocol Framework for Machine Learning. *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security - CCS '18*, pages 35–52, 2018.

[Smith *et al.*, 2017] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[Volgushev *et al.*, 2019] Nikolaj Volgushev, Malte Schwarzkopf, Ben Getchell, Mayank Varia, Andrei Lapets, and Azer Bestavros. Conclave: secure multi-party computation on big data (extended tr). *arXiv: Cryptography and Security*, 2019.

[Yang *et al.*, 2019] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.*, 10(2):12:1–12:19, January 2019.