2019 IEEE International Conference on Big Data (Big Data)

Privacy-preserving Heterogeneous Federated Transfer Learning

Dashan Gao*†, Yang Liu‡, Anbu Huang‡, Ce Ju‡, Han Yu§ and Qiang Yang*‡
*Department of Computer Science & Engineering, Hong Kong University of Science and Technology, Hong Kong, China
†Department of Computer Science & Engineering, Southern University of Science and Technology, Shenzhen, China
‡Webank, Shenzhen, China
§School of Computer Science & Engineering, Nanyang Technological University, Singapore
Email: dgaoaa@connect.ust.hk, yangliu@webank.com, stevenhuang@webank.com,
ceju@webank.com, han.yu@ntu.edu.sg, qyang@cse.ust.hk

***Abstract*—Federated learning (FL) allows multiple parties to collaboratively train a machine learning model without sharing raw data. However, existing approaches are mainly designed for homogeneous feature spaces and fail to tackle covariate shift and feature heterogeneity without privacy leakage. In this paper, we propose a transfer learning approach to tackle the covariate shift of the overlapped homogeneous feature spaces, and bridge different data owners' heterogeneous feature spaces with stringent privacy preservation in FL. We propose an end-to-end privacy-preserving multi-party learning approach with two variants based on homomorphic encryption and secret sharing techniques, respectively, to build a heterogeneous federated transfer learning (HFTL) framework. Finally, we not only demonstrate experimentally that the HFTL is secure, effective and highly scalable on five benchmark datasets, but also apply it into a real application of in-hospital mortality prediction from MIMIC-III dataset, where privacy is of significant concern.**

I. Introduction

In recent years, artificial intelligence (AI) has enjoyed rapid advances thanks to the availability of labeled data [4]. However, as AI becomes part of our everyday life, especially in sensitive domains such as healthcare [32], ethical and privacy concerns surrounding this technology start to attract public attention [42]. In the European Union, the General Data Protection Regulation (GDPR) [9] specified many terms for protecting user privacy and prohibit organizations from exchanging data without explicit user approval. Under this new legislative reality, the AI research field is forced to look for alternative development trajectories. Federated Learning (FL) has emerged to be one of the most promising ways forward [37].

Introduced in 2016 by Google [19], FL is a machine learning paradigm in which a group of data owners collectively train a model while keeping their data stored locally. In FL, model parameters or gradients are transmitted instead of raw data. Data owners with stringent privacy policies such as hospitals and banks may further require disclosing no information beyond the output model, as intermediate values such as gradient and instance weight may reveal certain information about training data [25]. Moreover, many real-world applications face more challenging scenarios, where data owners may not share the same features. Their data may differ greatly in the feature space, with only a limited number of common features typically of different distributions, and the rest of the features are unique. For example, patients from the hospital, nursing home, and physical examination centers may share common features such as age, blood pressure, etc. However, the marginal distribution of common features differs, and each medical institution holds additional uncommon features such as prescription and diet, as well. Existing FL approaches can not address the heterogeneous feature space problem with feature co-occurrence only. Current privacy-preserving FL approaches either assume homogeneous feature space [3] or heterogeneous feature space with instance co-occurrence [18]. We demonstrate in experiment that directly applying these approaches will lead to low test accuracy which is even worse than the model trained locally, due to the limited common features and the lack of aligned instances. Therefore, to fully leverage the data value of each party in FL, the research field faces a new challenge for dealing with heterogeneous data from multiple sources to safely and efficiently perform model training without violating data privacy and confidentiality.

To address this important challenge, we propose a new technique, called heterogeneous federated transfer learning (HFTL), to enable federated learning to deal with heterogeneous feature spaces using transfer learning. We design a privacy-preserving transfer learning approach to eliminate covariate shift of homogeneous feature spaces and bridge different data owners' heterogeneous feature spaces and propose an end-to-end secure multi-party learning protocol. Note that radically applying privacy-preserving techniques such as multi-party computation (MPC) to existing machine learning approaches to achieve privacy-preserving FL will lead to severe efficiency and scalability issues, as existing machine learning approaches extensively use non-linear operations and data transmission. Here We consider two variants based on homomorphic encryption (HE) [26] and secret sharing (SS) [28] techniques, respectively, and perform a comprehensive evaluation. Through extensive experiments based on real-world datasets, HFTL is shown to be more advantageous than common-feature-based FL approaches and self-learning approaches under challenging conditions. HFTL also demonstrates practical efficiency and scalability. To the best of our knowledge, it is the first privacy-preserving federated learning

978-1-7281-0858-2/19/$31.00 ©2019 IEEE
978-1-7281-0858-2/19/$31.00 ©2019 IEEE

approach to relax the common feature space assumption and tackles covariate shift as well as feature heterogeneity simultaneously, thereby providing a mechanism to better support the emergence of interdisciplinary vertical data federations.

II. Related Work

a) Federated Learning: The emergence of federated learning [17], [19] has encouraged local data storage and pushed more computation to the edge. It has the benefits of both distributed learning [29] and data privacy and confidentiality [3]. All of the above studies enable data clients to collaboratively train a single model. Federated multi-task learning was first proposed by Smith et al. [31] to address non-i.i.d. distribution of data by building multiple correlated models instead of a single model.

b) Privacy-preserving Machine Learning: Privacy-preserving machine learning typically involves multiple parties to perform machine learning, with emphasis on security guarantees. Earlier works [15], [34] mostly focus on approaches based on multi-party computation (MPC) [38]. Additively homomorphic encryption is studied for private federated logistic regression on vertically partitioned data [16]. More recently, privacy-preserving linear regression on horizontally partitioned data using homomorphic encryption and Yao's garbled circuits is proposed [21]. SecureML [20], a privacy-preserving protocol combining secret-sharing [28] and Yao's Garbled Circuit, is considered as the state-of-the-art protocol for linear regression, logistic regression, and neural networks. SecureNN [33] is also proposed using a 3-party or 4-party protocol for efficient neural network training. Differential Privacy (DP) [8] is studied to provide privacy preservation against membership-inference attack in the model inference stage. However, privacy-preserving transfer learning approaches based on DP [35], [39] are vulnerable to privacy leakage among the participants during model training.

c) Transfer Learning: Transfer learning enables knowledge sharing among related domains [24]. It can be mainly categorized into homogeneous transfer learning and heterogeneous transfer learning. In homogeneous transfer learning (a.k.a domain adaptation), the source domain and target domain only differs in marginal distribution [2], [13], [23]. Heterogeneous transfer learning takes the variations of feature sets into consideration and is more challenging. Problems with instance co-occurrence only [18], [30] and feature co-occurrence only [11], [40] are both studied without privacy concern. Hypothesis transfer is used to prevent raw data reveal [40]. However, the revealed source task model and feature meta representation of both parties reveals more information than the final transferred model contains.

III. Preliminaries

A. Problem Definition

Consider a data federation with K source parties $\{S_k\}_{k=1..K}$ and each source party S_k possess a dataset $\mathcal{D}_k := (X^{S_k}, Y^{S_k})$ where $X^{S_k} \in \mathbb{R}^{N_k \times d_k}$ and $Y^{S_k} \in \mathbb{R}^{N_k \times 1}$, N_k is the number of samples, and d_k is the number of features. Suppose a target party T with dataset $\mathcal{D}_T$. We denote the feature space as $\mathcal{X}$, and the label space as $\mathcal{Y}$. Suppose $\mathcal{Y}_{S_1} = ... = \mathcal{Y}_{S_K} = \mathcal{Y}_T$ and $\mathcal{X}_{S_k} \cap \mathcal{X}_T \neq \emptyset$ for $k = 1..K$. That is, the label spaces are the same, and the feature space of each source party has common overlap with the target party. For each source party, let X_s and X_t denote the uncommon feature space of the source and the target party respectively, and let X_c denotes the common feature space. A two-party scenario is shown in Table I.

Party	Feature space			Label
S	X_s^S	X_c^S		Y^S
T		X_c^T	X_t^T	Y^T

TABLE I
A TWO-PARTY HFTL SCENARIO

Here, we assume that parties already found their commonly shared feature spaces. Data are unevenly distributed among parties and the target party has scarce labeled data and some unlabeled data. Covariate shift also exists in the common feature spaces between source parties and target party. Given the above setting, the objective is for all parties to build a machine learning model to help the target party to predict labels as accurately as possible without exposing more information than the final model.

B. Security Model

We consider a threat model with an *honest-but-curious* adversary $\mathcal{A}$ who can corrupt at most one of the parties. That is, the corrupted party is curious about the private training sets of other parties but strictly follows the protocol without tampering with it. The security definition is that, for a protocol P performing $(O_A, O_B) = P(I_A, I_B)$, where I_i and O_i are the input and output of party $i \in \{A, B\}$. P is secure against $\mathcal{A}$ if $\mathcal{A}$ can only learn the input data I_i and the final output O_i of the party $i \in \{A, B\}$ it has corrupted, but nothing else about the other honest party's data.

C. Secure Logistic Regression

Logistic regression (LR) has a good balance of complexity and performance. Therefore, we adopt SLR to achieve secure FL. We demonstrate how to conduct privacy-preserving LR training via secure two-party computation. For convenience, we denote $\langle \cdot \rangle$ as secret values, which is in the form of either arithmetic secret sharing or additive HE ciphertext. For the secret value matrix, we denote the matrix multiplication by $\cdot$ and element-wise multiplication by $\odot$. Assume $\mathcal{D} = \{X, Y\}$ is the joint dataset from every party in FL , where $X \in \mathbb{R}^{n \times d}$ is the samples and $Y \in \{-1, 1\}^n$ is the corresponding class label. $\alpha \in \mathbb{R}^n$ is the instance weight vector. S and T want to learn a logistic regression model $w \in \mathbb{R}^d$. We consider the regularized risk minimization problem:

$$\min_w \frac{1}{n} \sum_{i=1}^{n} \alpha_i l(x_i w, y_i) + \frac{1}{\lambda} ||w||_2^2 \tag{1}$$

where $l(x_i w, y_i) = log(1 + e^{-x_i w \odot y_i})$ is the logistic loss. As only polynomial formed model is available for secure

computation, the second-order Taylor expansion of loss around 0 is adopted. The polynomial-approximated loss is:

$$\begin{aligned}\mathcal{L}(w) &= \frac{1}{n}\alpha^T log(1+e^{-Xw\odot Y}) + \frac{\lambda}{2}||w||_2^2 \\ &\approx \frac{1}{n}\alpha^T(log2 - \frac{1}{2n}Y^TXw + \frac{1}{8n}Z^TZ) + \\ &\frac{\lambda}{2}||w||_2^2\end{aligned} \tag{2}$$

where $Z = Xw \odot Y$. Considering $YY^T = \mathbf{I}_{n\times n}$, the batch gradient is:

$$\nabla\mathcal{L}(w) = \frac{1}{2n}\alpha^T(\frac{1}{2}X^TXw - X^TY) + \lambda w \tag{3}$$

The gradient formulation over secret values of equation 3 is:

$$\langle\nabla\mathcal{L}(w)\rangle = \langle\alpha\rangle^T \cdot \frac{1}{2n}(\frac{1}{2}\langle X\rangle^T \cdot \langle X\rangle \cdot \langle w\rangle - \langle X\rangle^T \cdot Y) + \lambda\langle w\rangle \tag{4}$$

When the secure gradient value is computed, participants in FL perform secure model update respectively, which can be conducted locally for both secret sharing and HE:

$$\langle w\rangle = \langle w\rangle - \eta\langle\nabla\mathcal{L}(w)\rangle \tag{5}$$

The whole SLR procedure is shown in Algorithm 1. We remark that the loss check step reconstructs the secret loss. Although the loss seems useless for training data estimation, it may disclose extra information beyond the final output model. One may enlarge the loss check step size γ for less privacy leakage, or even conduct fixed iteration gradient descent, to reveal nothing but the output model.

Algorithm 1 SLR: Secure Logistic Regression

Require: Dataset $\mathcal{D}$, maximum iteration M, loss check step size γ, two parties S and T, learning rate η;

1: S and T initialize classifier parameters w_0.
2: **for** $i = 1, 2, ..., M$ **do**
3: Party S and T do:
4: Compute secret gradients $\langle\nabla\mathcal{L}(w^k)\rangle$ following Equation 4
5: Update $\langle w^k\rangle$ following Equation 5
6: **if** i mod γ = 0 **then**
7: Compute $\langle\mathcal{L}(w)\rangle$ based on Equation 2 via secure computation.
8: Reconstruct $\mathcal{L}(w)$ and check model convergence.
9: **if** $\mathcal{L}(w)$ converges **then**
10: Stop training.
11: **end if**
12: **end if**
13: **end for**

IV. Approaches

The proposed HFTL approach involves five steps:

- Secure domain adaptation.
- Secure feature mapping.
- Secure federated learning.
- Secure model integration.
- Local model inference.

In secure domain adaptation phase, each source party computes its instance weights with the target party by building a domain classifier. Then each source party along with the target party mutually reconstructs the uncommon features by learning a feature mapping. After the covariate shift and feature heterogeneity are tackled by domain adaptation and feature mapping, source parties and target party conduct federated learning to obtain the label prediction model. To avoid expensive MPC during inference phase, secure model integration is proposed to merge the secret feature mapping model and the label prediction model into one final model for disclosure. As the output model is in clear-text, it can be efficiently inferred locally without MPC. The whole approach is demonstrated in Algorithm 3.

a) Secure Domain Adaptation (SDA): As covariate shift exits in common feature spaces of different parties, we apply private instance re-weighting for domain adaptation [27]. Each source party S_k first does two-party private horizontal federated learning with the target party T to learn a domain classifier. For the sake of efficient secure computation, SLR is adopted as domain classifier. To train the domain classifier, all labeled and unlabeled instances from T are labeled as positive class and instances from S_k are labeled as negative class. The instance weight for SLR training is uniform. After the domain classifier is well trained following Algorithm 1, secure domain adaptation via instance re-weighting is conducted. For each instance x_i in S_k, we take the odds of x_i belonging to $\mathcal{D}_T$ as x_i's instance weight:

$$\alpha(x_i) = \frac{P(x_i \in \mathcal{D}_T)}{1 - P(x_i \in \mathcal{D}_T)} = e^{w_k x_i} \tag{6}$$

As only arithmetic operations are supported for our MPC schemes. We approximate the exponential function as:

$$e^z \approx 1 + z\left(1 + \frac{z}{2}\left(...(1 + \frac{z}{n-1}(1 + \frac{z}{n}))\right)\right) \tag{7}$$

n is the order of approximation. When z is a secret value, $n-1$ secure two-party multiplication operations are required to compute n-order approximation of $\langle e^z\rangle$. The whole SDA procedure is shown in Algorithm 2.

b) Secure Feature Mapping (SFM): In the feature mapping phase, each party locally learns a feature mapping that transforms the feature spaces of the common features to uncommon ones [40]. The feature mapping $\theta^{S_k,T}$ maps the common feature space to S_k's uncommon feature space, and vice versa. The feature mappings $\theta^{S_k,T}$ and θ^{T,S_k} can be learned by S_k and T respectively by solving:

$$\min_{\theta^{S_k,T}} ||diag(\alpha_k)\cdot(X_{S_k}^{S_k} - X_c^{S_k}\theta^{S_k,T})||_F^2 + \lambda\sum_{n=1}^{d_2}||\theta_n^{S_k,T}||_F^2 \tag{8}$$

$$\min_{\theta^{T,S_k}} ||X_t^T - X_c^T\theta^{T,S_k}||_F^2 + \lambda\sum_{n=1}^{d_2}||\theta_n^{T,S_k}||_F^2 \tag{9}$$

where $diag(\cdot)$ transforms a vector to a diagonal matrix. As the instance weight α_k learned in SDA is secret value. Two

Algorithm 2 SDA: Secure domain adaptation

Require: Common feature datasets $\{D_{k,c}\}_{k=1..K}$ of K Source parties $\{S_k\}_{k=1..K}$, dataset D_T of target party T, max iterations M;

1: Each source party S_k generate cryptography materials with T.
2: **for** $k = 1, ..., K$ **do**
3: S_k and T do:
4: Train SLR (Algorithm 1) with common features and domain index as input, and obtain the secret domain classifier model parameters $\langle w_k \rangle$
5: Compute $\langle z_k \rangle = \langle w_k \rangle \cdot \langle X_k \rangle$ for source domain instances.
6: Compute S_k's secret instance weight α_k following Equation 7, via recursive secure multiplication.
7: **end for**

party secure computation is required to compute Equation 8, which is expensive. To trade feature mapping performance for computation efficiency, S_k can set α_k to be constant and learn feature mapping locally. After the feature mapping is learnt, uncommon features of each source party S_k can be reconstructed via $\langle \hat{X}_t^{S_k} \rangle = X_c^{S_k} \cdot \langle \theta^{T,S_k} \rangle$, and uncommon features of T can be reconstructed similarly. After privacy-preserving evaluation, we obtain secure estimated features shown in Table II.

Party	Feature space			Label
S_k	$X_{S_k}^{S_k}$	$X_c^{S_k}$	$\langle \hat{X}_t^{S_k} \rangle$	Y^{S_k}
T	$\langle \hat{X}_{S_k}^{T} \rangle$	X_c^T	X_t^T	Y^T

TABLE II
SECURE FEATURE RECONSTRUCTION OF SECRET FEATURES

c) Secure Federated Learning (SFL): With data from all parties mapped to the same feature space, source and target parties can collaboratively conduct horizontal federated learning over the mapped homogeneous feature space. We now design a secure federated learning algorithm based on SLR as shown in Algorithm 3. Specifically, federated learning is conducted between each source party S_k and the target party T with the reconstructed secret full features learned in SFM as input data. The instances are privately re-weighted by the secret instance weight learned in SDA. In each iteration, both parties compute local secret gradients and update the secret model. After transfer and federated learning, the target party obtains the model parameter w_k for source party S_k without learning S_k' data.

d) Secure Model Integration (SMI): After SFM and SFL, the target task can now do label prediction by first privately estimating missing feature then evaluating the federated learning model. However, such inference requires the coordination of every source party, which is expensive. Under our stringent security requirement, each source party is not allowed to disclose their own feature transfer model and the federated learning model, because these models contain information of

Algorithm 3 HFTL: Heterogeneous federated transfer learning

Require: Datasets $\{D_k\}_{k=1..K}$ of K Source parties $\{S_k\}_{k=1..K}$, dataset D_T of target party T, max iterations M;

1: Each source party S_k generate cryptography materials with T;
2: Each source party S_k conducts SDA with T, and learns source domain instance weight α_k;
3: Each party locally learns a feature mapping to get θ^{T,S_k} and $\theta^{S_k,T}$ for $\{S_k\}_{k=1...K}$;
4: **for** $k = 1, ..., K$ **do**
5: S_k and T do:
6: Mutually do SFM to reconstruct $\langle X_t^{S_k} \rangle$ and $\langle X_s^T \rangle$;
7: Train SLR (Algorithm 1) with secret full features, labels and instance weight α_k as input, and obtain the secret domain classifier model parameter $\langle w_k \rangle$
8: Do SMI over $\langle \theta^{S_k,T} \rangle$ and $\langle w^k \rangle$ following Equation 11, and get integrated model $\langle w_T^k \rangle$.
9: Reconstruct and reveal parameters w_T^k to T.
10: **end for**
11: T gets an ensemble $\hat{Y} = \frac{1}{K} \sum_{k=1}^{K} Sigmoid(X w_T^k)$.

heterogeneous feature from their own datasets. To reach the same inference efficiency as local trained model inference while preventing extra dataset information of each party from being disclosed, we also propose a model integration phase.

We remark that $\langle \theta^{S_k,T} \rangle$ is the secret feature mapping learned by S_k in SFM, and $\langle w_T^k \rangle = [\langle w_{S_k} \rangle^T \langle w_c \rangle^T \langle w_T \rangle^T]^T$ is the secret model learned by S_k and T in SFL, where w_{S_k}, w_t and w_c corresponds to the weight of the uncommon features and common features of S_k and T. For model inference, T inputs $x = [\hat{x}_{S_k}\ x_c\ x_t]$, where $\hat{x}_{S_k} = x_c \theta^{S_k,T}$ is previously estimated via SFM. T computes $\hat{y}_k = Sigmoid(x \cdot w_T^k)$ as model w_k's estimation. However, as $\theta^{S_k,T}$ is privately held by S_k, two secure inferences are required for secure model inference. x is a row vector and w_T^k is a column vector. We find that:

$$\begin{aligned} x \cdot w_T^k &= [\hat{x}_{S_k}\ x_c\ x_t] \cdot \begin{bmatrix} w_{S_k} \\ w_c \\ w_t \end{bmatrix} \\ &= [x_c \theta^{S_k,T}\ x_c\ x_t] \cdot \begin{bmatrix} w_{S_k} \\ w_c \\ w_t \end{bmatrix} \\ &= [x_c\ x_t] \cdot \begin{bmatrix} \theta^{S_k,T} w_{S_k} + w_c \\ w_t \end{bmatrix} \end{aligned} \tag{10}$$

Therefore, to avoid expensive secure computation during inference, S_k and T can conduct secure model integration right after SFL:

$$\langle w_T^k \rangle = \begin{bmatrix} \langle \theta^{S_k,T} \rangle \cdot \langle w_{S_k} \rangle + \langle w_c \rangle \\ \langle w_t \rangle \end{bmatrix} \tag{11}$$

After computing Equation 11, S_k and T reconstruct and reveal w_T^k to T, then T gets a clear-text label inference model

w_T^k built with S_k. In the end, what the target party obtains are weight parameters w_T corresponding to its own features, which is a new model that can be directly used for inferences locally.

e) Local Model Inference: Once the inference model is trained between each source party and the target party, the target party gets a model ensemble $\hat{Y} = \frac{1}{K}\sum_{k=1}^{K} f(x, w_T^k) = \frac{1}{K}\sum_{k=1}^{K} Sigmoid(Xw_T^k)$ as the label prediction model, which is in clear-text and can be efficiently inferred locally. At inference phase, a full model instead of the polynomial form of models can be applied directly since both the model weights and features are no longer secret shared or HE-encrypted.

V. Two variants of HFTL

Two variants of HFTL are proposed and implemented:

1) *SS-based HFTL* which is a secret sharing-based variant of HFTL leveraging Multi-Party Computation (MPC) and requires neither trusted third party nor collaborator.
2) *HE-based HFTL* which leverages an honest-but-curious collaborator to assist the computation of model updates from different parties. We denote this honest-but-curious collaborator as $\mathcal{S}$. The collaborator only learns double-masked intermediate values from all parties, but no other information. Each party learns nothing but its input data and output model parameters.

A. Security Analysis

Proposition 1. The secret-sharing-based HFTL protocol is secure under our security definition, provided that the underlying arithmetic secret sharing scheme is secure.

Proof. In the secret sharing-based HFTL, all the intermediate results are secret shared. The security of secret shared values is based on the uniform random sampling operation for secret share generation and the security of Oblivious Transfer extension [1] for secret-sharing multiplication. Addition is performed locally and is secure. Therefore, only the final output model of the target domain party is disclosed by S^k sending shared model from to T, and T reconstructing the shared model locally. Thus, S^k knows nothing about T's final model and data information. The final model w_T^k only consists of weights corresponding to T's own features as shown in Equation 11. T cannot solve $\theta^{S_k,T}$, w_{S_k} and w_c from w_T^k, and w_t is the weight of T's own features. Therefore, the final output model does not reveal any meaningful information of S^k's data and model information to T. Therefore, the secret sharing-based HFTL satisfies our security definition. □

Proposition 2. The HE-based HFTL protocol is secure under our security definition, provided that the underlying additively homomorphic encryption scheme is secure.

Proof. In the HE-based HFTL, all data parties learned are the encrypted features. The model parameters and intermediate data are encrypted and/or masked [5] throughout federated learning. As the masked values in secure multiplication is masked by values drawn from uniform distribution, the masked values guarantee perfect privacy. Therefore, as long as the encryption scheme is secure, the proposed protocol is secure. □

VI. Experimental Evaluation

We investigated the HFTL protocols over five datasets and evaluated the performance improvement of HFTL over common-feature-based horizontal federated learning and self-learning. Besides, we studied the HFTL performance in different task configurations and evaluated the time and communication complexity of adopting HE and secret sharing, respectively.

A. Benchmark Datasets

Experiments are conducted on several public datasets from UCI repository [7] including: 1) Wisconsin cancer dataset ("WDBC"), 2) Spambase, 3) Default of Credit Card Clients ("Default-Credit"), 4) mfeat-fourier, 5) heart disease dataset ("heart") to validate our proposed approaches. The effectiveness of the protocols is studied concerning various key factors, including the common feature ratio, the labeled instance number of target task. The scalability is also explored in terms of computation and communication time as well as data transfer volume. Nominal features are converted to numerical features and multi-class task is converted to binary classification.

B. Compared Methods

We evaluate HFTL against four baselines:

1) LocalLR: Target domain party learns a logistic regression model with its own features;
2) LocalSVM: Target domain party learns an SVM [41] model with its own features;
3) Common: privacy-preserving horizontal federated learning over common features among different parties. A second degree Taylor approximation of logistic regression is used for HE and secret-sharing related calculations;
4) $HFTL_{LR}$: A non-privacy-preserving variant of HFTL trained locally via logistic regression. This is used as a benchmark for the impact of polynomial approximation and secret arithmetic operations.

Two variants of HFTL are being evaluated. They are $HFTL_{SS}$ based on secret sharing techniques and $HFTL_{HE}$ based on Paillier's [22] additive HE scheme with a key size of 1,024 bits. Linear regression is used for feature transfer for $HFTL_{LR}$, $HFTL_{HE}$ and $HFTL_{SS}$. The configuration of secret sharing scheme is similar to that of SecureML [19]. The fixed-point 32-bit number is used to represent values in the secret-sharing arithmetic operations. 13 bits are used to represent the fractional part and a negative number z is represented as $2^{32} - |Z|$. The data size transferred by each party for multiplication operation is the same as the share size to be multiplied. We adopt ABY [6] for offline Beaver triples generation, based on Oblivious Transfer extension [1]. The overall amortized time to generate a 32-bit Beaver triple is $107\mu s$. Parties communicate following the gRPC protocol.

For the first five experiments, we randomly split features of all instances into three parts, corresponding to source-task-specific features, target-task-specific features, and common features. Their feature proportion are 45%, 45%, 10%, respectively. The number of source party is two, and source domain instances are evenly distributed. To simulate the unbalanced distribution of labeled data, 50% of the dataset is sampled as source party data, four instances are sampled as target party data and the remaining data are taken as test data of target task. Such configuration is similar to [40]. The learning rate is 0.05, regularization factor is 0.01, the batch size is 256, and the maximum iteration number is 1,000. Each experiment is repeated for 15 times, and test accuracy is used as the measurement. The experiments are performed on a PC with Intel(R) Core(TM) i5 CPU and 12G Memory.

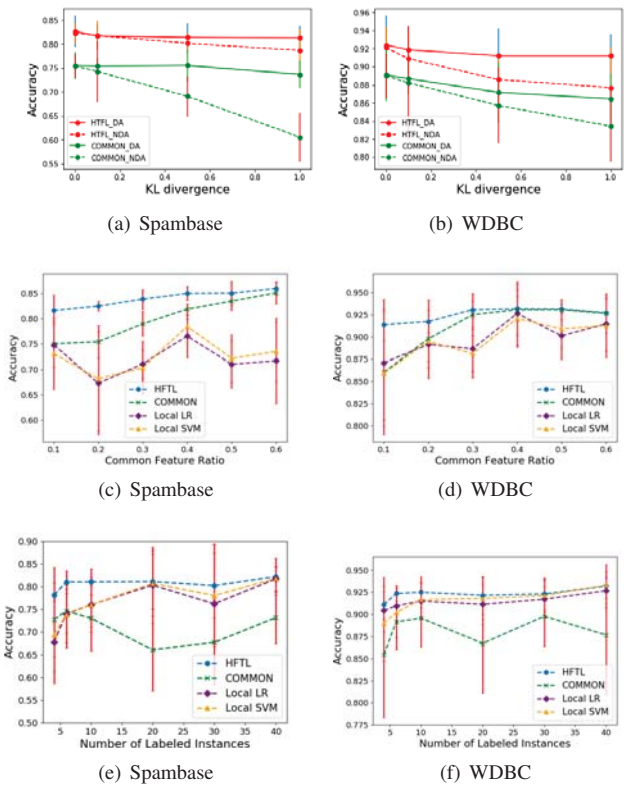
C. Performance

The comparison of test accuracy (mean $\pm$ std) is shown in Table III. Results with the highest average accuracy on each dataset is in bold. We show that HFTL approaches outperform other approaches, especially when there is only a small amount of labeled data in target party and when the number of common features of different parties is limited.

Figure 1(a) and 1(b) demonstrate the performance of SDA and explore the influence of the covariate shift, by adjusting the KL-divergence between common features in source domain and target domain from 0.0 to 1.0. DA denotes approaches with privacy-preserving domain adaptation, and NDA denotes approaches without domain adaptation. It can be observed that, when covariate shift increases, methods with SDA are more robust. When domain adaptation is not applied, the performance of $COMMON_{NDA}$ drops significantly compared to $HFTL_{NDA}$, it shows that $HFTL$ is more robust than $COMMON$ approach towards covariate shift. Figure 1(c) and 1(d) explores the influence of the common feature ratio by adjusting it from 10% to 60% for the Spambase and WDBC data sets. The Common approach responses to this increase the most, with accuracy increasing dramatically. The accuracy of the HFTL approach also improves as more features are used to bridge the feature spaces, and it outperforms all the other approaches over the entire feature ratio space studied. Figure 1(e) and 1(f) explores the influence of target party instance number ranging from 4 to 40. It shows that the HFTL approach outperforms other approaches in the scenario of limited target instances, and the local models (LocalLR and LocalSVM) are more sensitive to this factor than other models. The performance of $HFTL_{SS}$ and $HFTL_{HE}$ are very close to $HFTL_{LR}$, demonstrating that the polynomial approximation, fixed-point decimal representation, and secret sharing arithmetic operations do not lead to a significant performance drop.

D. Scalability

Figure 2 shows the running time complexity of $HFTL_{SS}$ and $HFTL_{HE}$ as a function of the number of common features. It can be observed that overall $HFTL_{SS}$ is around



(a) Spambase (b) WDBC

(c) Spambase (d) WDBC

(e) Spambase (f) WDBC

Fig. 1. Changes of accuracy over Spambase and WDBC. Plots in the first row show the performance with different KL-divergence (from 0.0 to 1.0); the middle row show performance with different common feature ratio (from 0.1 to 0.6); while the bottom row lists the plots when the number of training instances grows (from 4 to 40).

30 times faster than $HFTL_{HE}$. This is because secret sharing operations leveraging Beavers triples does no cryptographic computation during online secure computation and the shared data transferred during multiplication operation is no more than two times costly than plain-text values. In contrast, and encryption time accounts for 57% of computation time on average. The communication cost of HE, however, is almost negligible compared to the computation cost whereas the communication time of SS-based approach increases steadily as the system grows.

Figure 3(a) and 3(b) show how the running time per iteration changes with feature number and batch size on $HFTL_{SS}$ respectively. It can be observed that the iteration time grows linearly with respect to the feature number, as well as the batch size. Such observation is consistent with our theoretical analysis above.

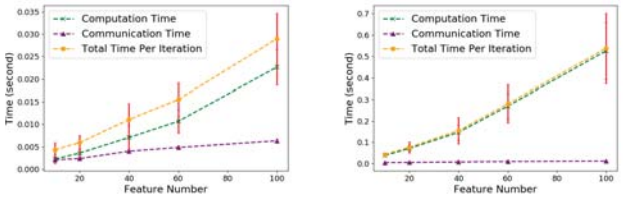
E. In-hospital Mortality Prediction

Our proposed HFTL is particularly suitable for real-world applications where small data and data privacy are critical challenges. For example, in healthcare, patients' data are extremely sensitive and are not allowed to be shared. Labeling data requires tremendous professional effort. Currently, hospital data still exist in the form of silos and knowledge can not be

Dataset	$HFTL_{SS}$	$HFTL_{HE}$	$HFTL_{LR}$	Common	Local LR	Local SVM
Spambase	0.8286 ± 0.0311	$\mathbf{0.8315} \pm 0.0355$	0.8161 ± 0.0346	0.7568 ± 0.0246	0.7382 ± 0.0513	0.7400 ± 0.0699
WDBC	0.9243 ± 0.0216	0.9307 ± 0.040	$\mathbf{0.9491} \pm 0.0482$	0.8909 ± 0.0541	0.9102 ± 0.0389	0.9095 ± 0.0389
mfeat-fourier	$\mathbf{0.6923} \pm 0.0162$	0.6886 ± 0.0385	0.6911 ± 0.0391	0.6678 ± 0.0280	0.5972 ± 0.0896	0.5902 ± 0.0892
heart	$\mathbf{0.7230} \pm 0.0659$	0.7220 ± 0.0521	$\mathbf{0.7230} \pm 0.0498$	0.6408 ± 0.0850	0.7019 ± 0.0572	0.6897 ± 0.0759
default_credit	0.5140 ± 0.0402	$\mathbf{0.5212} \pm 0.0423$	0.5030 ± 0.0500	0.4865 ± 0.0381	0.5037 ± 0.0395	0.5072 ± 0.0494
MIMIC-III	0.7421 ± 0.0202	0.7403 ± 0.0423	$\mathbf{0.7432} \pm 0.0315$	0.7077 ± 0.0313	0.6085 ± 0.0255	0.6327 ± 0.0264

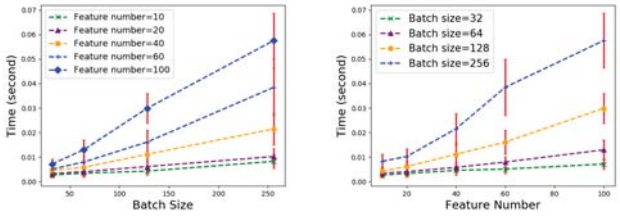
TABLE III

COMPARISON OF DIFFERENT APPROACHES (TEST ACCURACY, MEAN± STD.). THE BEST PERFORMANCE ARE IN BOLD.



(a) $HFTL_{SS}$ (b) $HFTL_{HE}$

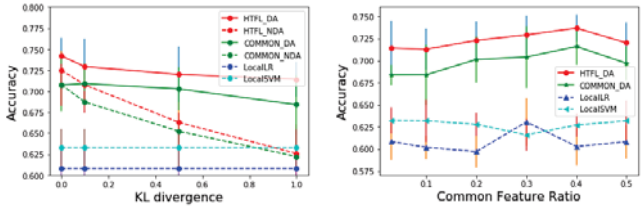
Fig. 2. Time complexity with different feature number ranging from 10 to 100 for $HFTL_{SS}$ and $HFTL_{HE}$. Computation, communication and total time per iteration are shown.



(a) time vs. batch size (b) time vs. feature number

Fig. 3. Changes of iteration time on SFL phase with different feature number and batch size for $HFTL_{SS}$.

transferred across different hospitals. In this section, we apply our proposed approach to a real-world dataset MIMIC-III [14] for in-hospital mortality prediction task [10]. This task aims to predict in-hospital mortality based on the first 48 hours of an ICU, with 714 features and 20,000 records. We split the data to simulate multiple hospitals and they share a limited number of common services (features). The common feature ratio is 3% and the uncommon features are evenly split for source and target parties. The target party has 10 instances in each class. The other settings are the same as former experiments. Partial HFTL is leveraged to reconstruct the uncommon feature space of source parties and predict the mortality privately. The performance of different approaches when KL-divergence=0 is shown is Table III. Figure 4(a) compares the performance of different approaches under different covariate shift. As can be seen, SDA constantly improves the model's robustness. Figure 4(b) explores the impact of common feature ratio while KL-divergence=1. It can be observed that the performance of FL approaches (HFTL and COMMON) grows as common feature ratio increases.



(a) KL divergence impact (b) common feature ratio impact

Fig. 4. Change of accuracy over in-hospital mortality prediction with different KL-divergence (from 0.0 to 1.0) and different common feature ratio (from 0.03 to 0.5).

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a novel privacy-preserving heterogeneous federated transfer learning framework. Such a framework enables multiple parties with sufficient labeled data to help boost a party with a limited amount of labeled data. The experiments show that our proposed HFTL can outperform local models and homogeneous federated learning under challenging conditions. We provide privacy-preserving protocols in both HE and secret sharing settings and compared the security and efficiency of both protocols. For future work, we will explore the adaptation of other transfer mechanisms and other classifiers, such as decision tree and deep networks.

REFERENCES

[1] Gilad Asharov, Yehuda Lindell, Thomas Schneider, and Michael Zohner. More efficient oblivious transfer and extensions for faster secure computation. In *Proceedings of the 2013 ACM SIGSAC*, pages 535–548. ACM, 2013.

[2] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128, 2006.

[3] 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. In *CCS*, pages 1175–1191, 2017.

[4] Min Chen, Shiwen Mao, and Yunhao Liu. Big data: A survey. *Mobile Networks and Applications*, 19(2):171–209, 2014.

[5] Ronald Cramer, Ivan Damgård, and Jesper Buus Nielsen. Multiparty computation from threshold homomorphic encryption. In *Proceedings of the ICTACT*, EUROCRYPT '01, pages 280–299, 2001.

[6] Daniel Demmler, Thomas Schneider, and Michael Zohner. Aby-a framework for efficient mixed-protocol secure two-party computation. In *NDSS*, 2015.

[7] Dheeru Dua and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[8] Cynthia Dwork. Differential privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[9] EU. REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL Directive 95/46/EC (General Data Protection Regulation). *Available at: https://eur-lex. europa. eu/legal-content/EN/TXT*, 2016.

[10] Hrayr Harutyunyan, Hrant Khachatrian, David Kale, and Aram Galstyan. Multitask learning and benchmarking with clinical time series data. *Scientific Data*, 6, 03 2017.

[11] Bo-Jian Hou, Lijun Zhang, and Zhi-Hua Zhou. Prediction with unpredictable feature evolution, 2019.

[12] Chenping Hou and Zhi-Hua Zhou. One-pass learning with incremental and decremental features. *IEEE transactions on pattern analysis and machine intelligence*, 40(11):2776–2792, 2018.

[13] Jiayuan Huang, Arthur Gretton, Karsten Borgwardt, Bernhard Schölkopf, and Alex J Smola. Correcting sample selection bias by unlabeled data. In *Advances in neural information processing systems*, pages 601–608, 2007.

[14] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[15] Murat Kantarcioglu and Chris Clifton. Privacy-preserving distributed mining of association rules on horizontally partitioned data. *IEEE TKDE*, 16(9):1026–1037, 2004.

[16] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption, 2017.

[17] Jakub Konecný, H. Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *CoRR*, abs/1610.02527, 2016.

[18] Yang Liu, Tianjian Chen, and Qiang Yang. Secure federated transfer learning, 2018.

[19] H. Brendan McMahan, Eider Moore, Daniel Ramage, and Blaise Agüera y Arcas. Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629, 2016.

[20] Payman Mohassel and Yupeng Zhang. SecureML: A system for scalable privacy-preserving machine learning. *IACR Cryptology ePrint Archive*, page 396, 2017.

[21] Valeria Nikolaenko, Udi Weinsberg, Stratis Ioannidis, Marc Joye, Dan Boneh, and Nina Taft. Privacy-preserving ridge regression on hundreds of millions of records. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP '13, pages 334–348. IEEE Computer Society, 2013.

[22] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, pages 23–38, 1999.

[23] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010.

[24] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[25] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE TIFS*, 13(5):1333–1345, May 2018.

[26] R L Rivest, L Adleman, and M L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation*, pages 169–179, 1978.

[27] Jing Jiang. A literature survey on domain adaptation of statistical classifiers. *URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey*, 3:1–12, 2008.

[28] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to share a secret. *Communications of the ACM*, 22(22):12–13, 1979.

[29] Reza Shokri and Vitaly Shmatikov. Privacy-preserving deep learning. In *CCS*, pages 1310–1321, 2015.

[30] Xiangbo Shu, Guo-Jun Qi, Jinhui Tang, and Jingdong Wang. Weakly-shared deep transfer networks for heterogeneous-domain knowledge propagation. In *MM*, pages 35–44, 2015.

[31] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *NIPS*, pages 4424–4434. 2017.

[32] Daniel S. W. Ting, Yong Liu, Philippe Burlina, Xinxing Xu, Neil M. Bressler, and Tien Y. Wong. AI for medical imaging goes deep. *Nature Medicine*, 24:539–540, 2018.

[33] Sameer Wagh, Divya Gupta, and Nishanth Chandran. Securenn: Efficient and private neural network training. *IACR Cryptology ePrint Archive*, 2018:442, 2018.

[34] Li Wan, Wee Keong Ng, Shuguo Han, and Vincent C. S. Lee. Privacy-preservation for gradient descent methods. In *KDD*, pages 75–83, 2007.

[35] Yang Wang, Quanquan Gu, and Donald Brown. Differentially private hypothesis transfer learning. In Michele Berlingerio, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim, editors, *ECML PKDD*, pages 811–826, Cham, 2019. Springer.

[36] Liyang Xie, Inci M. Baytas, Kaixiang Lin, and Jiayu Zhou. Privacy-preserving distributed multi-task learning with asynchronous updates. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1195–1204. ACM, 2017.

[37] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM TIST*, 10(2), 2019.

[38] Andrew C. Yao. Protocols for secure computations. In *SFCS*, pages 160–164, 1982.

[39] Quanming Yao, Xiawei Guo, James T. Kwok, WeiWei Tu, Yuqiang Chen, Wenyuan Dai, and Qiang Yang. Differential private stack generalization with an application to diabetes prediction, 2018.

[40] Han-Jia Ye, De-Chuan Zhan, Yuan Jiang, and Zhi-Hua Zhou. Rectify heterogeneous models with semantic mapping. In *International Conference on Machine Learning*, pages 5630–5639, 2018.

[41] Jieping Ye and Tao Xiong. Svm versus least squares svm. In *Artificial Intelligence and Statistics*, pages 644–651, 2007.

[42] Han Yu, Zhiqi Shen, Chunyan Miao, Cyril Leung, Victor R. Lesser, and Qiang Yang. Building ethics into artificial intelligence. In *IJCAI*, pages 5527–5533, 2018.