Privacy-Preserving Distributed Transfer Learning and its Application in Intelligent Transportation

Zhi Li, Hao Wang, *Member, IEEE*, Guangquan Xu, *Member, IEEE*, Alireza Jolfaei, *Senior Member, IEEE*, Xi Zheng, *Member, IEEE*, Chunhua Su, and Wenying Zhang

***Abstract*— With the rapid development of intelligent transportation systems (ITS), more and more intelligent applications for ITS have received widespread attention, such as the vehicle detection, inference of typical routes, and traffic forecasting. In these applications, deep learning is widely used as a key artificial intelligence technology. However, most ITS providers fail to collect enough labeled traffic data for model training. As a complement to deep learning, transfer learning is an effective way to solve the scarcity of labeled data, which can transfer knowledge from labeled datasets to unlabeled datasets, thus improving the accuracy of prediction and classification. Nevertheless, when the labeled dataset and the unlabeled dataset are held by different entities, it is still unrealistic for two mutually distrustful entities to cooperate in transfer learning regarding data security and privacy preservation. Although some existing works provide privacy-preserving transfer learning methods, such methods fail to apply to traffic data with high sample dimensions due to their high computational cost and round complexity. To address this problem, we design an efficient privacy-preserving distributed transfer learning protocol, which is appropriate for traffic data. Compared to existing works, our protocol addresses the privacy-preserving problem of transfer learning for traffic data with high sample dimensions. In addition, our protocol has fewer interaction rounds and can be proved in the semi-honest model. Finally, we validate the effectiveness, efficiency and security of the proposed protocol via experiments. Furthermore, we show the application of the proposed protocol in intelligent transportation systems.**

***Index Terms*— Privacy-preserving, transfer learning, transfer component analysis, intelligent transportation system.**

Manuscript received 4 August 2021; revised 5 September 2022; accepted 13 October 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62071280, Grant 62272282, Grant 62172297, and Grant 61902276; in part by the National Key Research and Development Program of China under Grant 2019YFB2101700; in part by the Major Scientific and Technological Innovation Project of Shandong Province under Grant 2020CXGC010115; in part by the Natural Science Foundation of Shandong Province under Grant ZR2020KF011 and Grant ZR2020MF056; in part by the Key Research and Development Project of Sichuan Province under Grant 2021YFSY0012; and in part by the Tianjin Intelligent Manufacturing Special Fund Project 20211097 and Project 20201159. The Associate Editor for this article was B. B. Gupta. *(Corresponding author: Hao Wang.)*

Zhi Li, Hao Wang, and Wenying Zhang are with the School of Information Science and Engineering, Shandong Normal University, Jinan 250399, China (e-mail: lizhisdnu@163.com; wanghao@sdnu.edu.cn; zhangwenying@sdnu.edu.cn).

Guangquan Xu is with the School of Big Data, Qingdao Huanghai University, Qingdao 266555, China, and also with the Tianjin Key Laboratory of Advanced Networking (TANK), College of Intelligence and Computing, Tianjin University, Tianjin 300072, China (e-mail: losin@tju.edu.cn).

Alireza Jolfaei is with the College of Science and Engineering, Flinders University, Adelaide, SA 5042, Australia (e-mail: alireza.jolfaei@flinders.edu.au).

Xi Zheng is with the Department of Computing, Macquarie University, Sydney, NSW 2109, Australia (e-mail: james.zheng@mq.edu.au).

Chunhua Su is with the Division of Computer Science, The University of Aizu, Aizuwakamatsu 965-8580, Japan (e-mail: chsu@u-aizu.ac.jp).

Digital Object Identifier 10.1109/TITS.2022.3215325

I. Introduction

TODAY, in the field of intelligent transportation systems (ITS), a growing number of researchers have achieved excellent results by applying deep learning [1], reinforcement learning [2] and transfer learning [3] methods. The achievements of intelligent transportation systems rely heavily on large amounts of labeled traffic data as training samples. However, because traffic data are time-sensitive, labeled traffic data is very insufficient, especially the traffic data that need to be labeled by professionals. The available labeled data are not always sufficient for the training task, which raises an interesting question: how to use a limited amount of labeled traffic data to build a reliable model to predict the unlabeled traffic data.

Since the driving behavior and traffic characteristics may vary from different locations or at different times and there are differences in traffic data collected by various types of sensors, the distribution of labeled datasets may be different from that of unlabeled datasets. Therefore traditional machine learning methods are no longer available. Transfer learning [4], which focuses on transferring knowledge across domains, is a promising machine learning method to address this problem. It uses the existing knowledge of the source domain to solve problems in different but related domains, and aims to transfer the existing knowledge to solve learning problems in the target domain with little or even no labeled data [5]. The more the source domain data and the target domain data have in common, the easier transfer learning will be, and vice versa. In the recent years, transfer learning has achieved excellent results in image processing, text processing, traffic forecasting [6], vehicle recognition [3], [7], road users detection [8] and other application scenarios.

Although transfer learning has a good performance in solving the problem of insufficient labeled traffic data [6], [7], there are still many obstacles in practical application. Especially when source domain data (labeled traffic data) and target domain data (unlabeled traffic data) are respectively held by different entities (organizations or companies), for data security and privacy-preserving considerations, it is difficult

for these data to be gathered together for transfer learning. In addition, from a legal viewpoint, companies are also not allowed to reveal any private data of their customers to other companies.

Therefore, more research has been devoted to privacy-preserving transfer learning in recent years. In 2018, Guo et al. [9] first proposed a method that simultaneously transfers knowledge from multiple external datasets and provides ε- differential privacy (DP) [10] guarantees. However, they do not offer formal security proof. In 2019, Gao et al. [11] proposed an end-to-end privacy-preserving multi-party transfer learning approach with two variants based on Partially Homomorphic Encryption (PHE) [12] and Secret Sharing techniques (SS). In 2020, Jin et al. [13] first proposed a neural network training framework based on Fully Homomorphic Encryption (FHE) [14]. They achieve privacy preservation of target domain data during transfer learning by using FHE. In the same year, Zhang et al. [15] proposed a secure version of the widely used maximum mean difference (SMMD) algorithm based on homomorphic encryption. However, all of these frameworks suffer from high computational complexity and rounds complexity. Therefore, it is hardly appropriate for the time-sensitive traffic data with a large sample size and high sample dimension. In addition, in an environment with high internet latency, the main factor that influences the computational efficiency of the protocol is the number of interaction rounds rather than the computational complexity. Thus the existing works are challenging to perform well in environments with high network latency.

To address this problem, we propose a series of secure computing subprotocols and a privacy-preserving distributed transfer learning protocol (PPDTL). These protocols enable two mutually distrustful parties to perform transfer learning without revealing their private data, where one party holds the source domain data, and the other party holds the target domain data. Our protocols are based on the transfer component analysis (TCA) algorithm [16] which is a representative transfer learning method and easy to implement. Compared with other existing privacy-preserving transfer learning works, PPDTL has low computation complexity (in the online phase) and the interaction rounds are constant. Therefore our protocols are especially appropriate for time-sensitive traffic data with high sample data dimension, which we also demonstrate through experiments. In addition, in an environment with high internet latency, the main factor that influences the computational efficiency of the protocol is the number of interaction rounds rather than the computational complexity. Compared to similar works, our protocol is a constant-round interaction protocol and thus has an excellent performance in the high internet latency environment. A comparison of our protocol with existing works is shown in TableI.

Further more, the final result obtained by our protocol is in the form of secret shares. This result can be obtained by the specified party or later used as the input to other secret-sharing-based secure two-party computation protocols for model training, such as SecureML [17], QUOTIENT [18] etc. Therefore, our protocol applies to more scenarios.

A. Contributions

The contribution of this paper can be summarized as follows:

1) We design an efficient privacy-preserving distributed transfer learning protocol, which can be performed in constant rounds and is well applicable to traffic data.
2) We apply our PPDTL protocol in the intelligent transportation field to solve the scarcity of available traffic data.
3) To the best of our knowledge, this is the first privacy-preserving transfer learning work to provide formal security proof.

B. Organizations

The rest of the paper is organized as follows. We introduce the related works and preliminaries in sections II & III. In section IV, we propose the privacy-preserving distributed transfer learning (PPDTL) protocol. Then, we give the formal security proof in section V. Finally, we apply our PPDTL protocol to the field of intelligent transportation in sections VI and give the experiments in section VII.

II. Related Work

A. Transfer Learning

Given a source domain D_s and a learning task T_s, as well as a target domain D_t and a learning task T_t, the goal of transfer learning is to capture the knowledge in the source domain D_s and the learning task T_s to help T_t achieve a better result, where $D_s \neq D_t$ or $T_s \neq T_t$. In the recent years, transfer learning has achieved excellent results in image processing, text processing, traffic forecasting [6], vehicle recognition [3], [7], road users detection [8] and other application scenarios. Transfer learning can be divided into data-based transfer learning and model-based transfer learning according to the approach [21]. Data-based transfer learning [22] focuses on transferring knowledge by adjusting the weights of sample data and transforming data features. Model-based transfer learning [23] focuses on transferring knowledge by fitting the structure or parameters of the source domain model. In this paper, we are mainly concerned with the domain adaptation and privacy preservation problems in transfer learning.

1) Domain Adaptation: Domain adaptation can be regarded as a special transfer learning scenario aiming to transfer knowledge in different but related domains [5], [24]. The essential computing problem in domain adaptation is minimizing the distribution distance between source and target domain data while preserving the significant properties of the original data (e.g., statistical properties or side information [25]). Most of the previous feature-based domain adaptation methods reduce the distance of the distribution between domains, such as Bünau et al. [26] proposed the Stationary Subspace Analysis (SSA). However, they focus on identifying subspace and do not consider the preservation of attributes such as data variance in subspace. Pan et al. [27] proposed a new method for dimensionality reduction called maximum mean discrepancy embedding (MMDE). The aim of MMDE is to learn a common latent space of source and target domains in which the

TABLE I
COMPARISONS

Works	Privacy preserving	Formal security proof	Constant round	Appropriate for traffic data
Krishnakumari et al. [19]	×	−	×	√
Mohammed et al. [8]	×	−	×	√
Eric L et al. [6]	×	−	×	√
Guo et al. [9]	√	×	×	×
Gao et al. [11]	√	×	×	×
Jin et al. [13]	√	×	×	×
Liu et al. [20]	√	×	×	×
Zhang et al. [15]	√	×	×	×
Our work	√	√	√	√

distribution distance between the source and target domains is reduced and the variance of the data is preserved as much as possible. However, there are two major limitations of MMDE. 1) MMDE cannot support out-of-sample patterns [28]; 2) MMDE requires solving a semi-definite program (SDP), which is very costly to compute. In 2010, Pan et al. [16] proposed the method Transfer Component Analysis (TCA) that avoids solving the SDP problem and simplifies the computational process. In addition, TCA supports generalizing to out-of-sample patterns. Later, Long et al. [29] proposed the joint distribution adaptation (JDA) to introduce further conditional MMD based on TCA so that JDA has higher precision than TCA. However, although JDA is more precise than TCA in the plaintext setting, the JDA algorithm requires a high number of iterations and interactions, so it is hard to execute the JDA algorithm in the ciphertext setting.

2) Privacy-Preserving Transfer Learning: With more and more application scenarios of transfer learning, the requirement for privacy-preserving in transfer learning has become more and more eager. Many privacy-preserving transfer learning works have appeared in recent years. In 2018, Guo et al. [9] first introduced privacy-preserving transfer learning with differential privacy, which achieves data privacy by introducing noises. However, this method will reduce the accuracy of the trained model. Then, Gao et al. [11] proposed a privacy-preserving transfer learning framework based on partial homomorphic encryption and secret sharing, but the number of interaction rounds is too excessive. In 2020, Jin et al. [13] proposed a privacy-preserving model-based transfer learning framework using fully homomorphic encryption. Liu et al. [20] designed a new framework called Federated Transfer Learning (FTL) based on partial homomorphic encryption and secret sharing techniques. However, it requires training models separately for the source and target domains, and the number of interaction rounds and computational complexity is expensive. Ma et al. [30] constructed a secure and verifiable bidirectional transfer learning scheme, VerifyTL, which supports knowledge transfer from the target domain to the source domain and is suitable for collaborative learning in multiple data-poor data domains. However, VerifyTL only applies to CNN model training and does not generalize to other machine learning models.

B. Intelligent Transportation Systems

The evolution of Intelligent Transportation Systems (ITS) started in the 1970s. It incorporates advanced technologies such as electronic sensor technology, data transmission technology, and intelligent control technology into transportation systems [31], which aims to provide better services to drivers and passengers in transportation systems [32]. With the development of big data and the Internet of Things, many devices generate large amounts of traffic data every day, such as many smartphones, driving recorders, GPS, cameras and detectors. To better utilize these data, machine learning, transfer learning and other artificial intelligence techniques become the focused research objects of intelligent transportation systems. At the same time, the rapid development of ITS also brings new security problems. Once the system is attacked by cyber security, it will seriously threaten the security and privacy of all road users. The remainder of this chapter details the three aspects of ITS.

1) Machine Learning in Intelligent Transportation Systems: Machine learning theory is widely used in the field of ITS. Linear regression, decision tree, support vector machine and neural network are the most commonly used models in ITS. Linear regression has an excellent performance in ITS due to its simplicity and robustness, such as traffic flow prediction [33], traffic speed estimation [34], and traffic travel route evaluation [35]. Decision tree is widely used in various ITS scenarios such as traffic accident detection [36], accident severity analysis [37], and travel pattern selection [38] due to its advantages of few hyperparameters, high interpretability, and robustness. Support vector machines have been successfully applied to travel time prediction [39], transit arrival time prediction [40], and traffic accident detection [41]. K-means is the most popular unsupervised learning tool and has been widely used in road traffic planning [42], travel time prediction [43]. When the quantity of labeled data is large enough, deep learning models perform better than traditional machine learning models. They have been widely used in ITSs. e.g., vehicle detection, traffic flow density [44], [45], etc.

2) Transfer Learning in Intelligent Transportation Systems: The earliest work introducing transfer learning into the field of transportation can be traced back to 2017. Zhao [7] proposed a method for vehicle logo recognition using transfer learning. In his work, a large number of labeled images of handwritten vehicle logos are used as source domain data. In contrast, a small number of labeled images of real vehicle logos are used as target domain data. Using transfer learning, a CNN classification model can be trained to recognize real vehicle logos. In 2018, Krishnakumari et al. [19] proposed a method for traffic state prediction using transfer learning.

They first transform the traffic states into meaningful images while preserving the properties and spatial information of traffic data. Then, they use the transformed traffic state images to train the CNN model for traffic state prediction. In the same year, Wang et al. [3] introduced online transfer learning to vehicle detection. Different sensors often collect vehicle data, and the data distribution of the training set may be different from that of the prediction set, so they use the new samples generated in the online phase to retrain the model trained in the offline phase. Thus, the retrained model can be used to predict vehicle detection accurately. In pavement detection, Zhang et al. [46] and Seongdeok et al. [47] proposed new methods for pavement crack detection with transfer learning. In 2020, Mohammed et al. [8] introduced transfer learning into road detection and proposed a method for detecting non-motorized road users with high accuracy. In the same year, Eric L et al. [6] introduced transfer learning and online learning into traffic prediction. They transfer model parameters from an existing traffic prediction model to a new model and fine-turn the new model through online learning. Using transfer learning, they save the computational overhead of training models and solve the problem of unreliable models when training data is insufficient. Although transfer learning has many advantages, the privacy-preserving issue may become the biggest obstacle to its promotion in practical applications. The method we proposed in this paper can solve this problem. Even if different entities hold the source domain data and the target domain data, our PPDTL protocol can perform transfer learning without revealing the private data of both parties.

3) Privacy Preservation in Intelligent Transportation Systems: Privacy protection is one of the most important issues in the Intelligent Transportation System (ITS) application. The development of ITS depends on a large amount of traffic data contributed by users or the massive data collected by road detection devices, cameras, etc. However, users are unwilling to disclose their sensitive private information, such as trip tracking, identity information, etc. In addition, if attackers break into the ITS and upload false information, it is very possible to cause a severe traffic accident. Cuong et al. [48] proposed a scheme to protect the identity information of users. In this scheme, users communicate with each other using pseudonyms, while a trusted authority (TA) is set up to authenticate pseudonyms and real identities. To avoid the information leakage of trajectories of users, Liu et al. [49] proposed a scheme of Identifying and Eliminating Violating privacy Subtrajectories (IEVS), which uses three anonymization schemes i.e., trajectory splitting, location suppression, and sensitive value. In ITS, it is often required to collect the aggregated information of users, such as average travel time, best route, etc. To prevent the leakage of their sensitive information during the aggregation process, Catalin et al. [50] propose a schema for privacy-preserving aggregation based on symmetric cryptography.

III. Preliminaries

A. Notations

In this paper, the bold lowercase letters represent vectors, e.g., $\mathbf{x}$, the vector followed by $[i]$ denotes the i th element of the vector, e.g., $\mathbf{x}[i]$, the bold capital letters represent matrices, e.g., $\mathbf{A}$, and the matrix is followed by $[i]$ to indicate the ith column of the matrix, e.g., $\mathbf{A}[i]$. These notations that appear in this article are described in detail in Table II

TABLE II
NOTATION AND MEANING

Notation	Meaning
Z_p	Prime field, where p is a large prime number.
P^n	The n-dimensional linear space.
$P^{n,n}$	The $n \times n$-dimensional linear space.
$\langle\cdot\rangle_i$	Arithmetic shares about a secret value belonging to the party i.
$\langle\cdot\rangle$	All arithmetic shares on a secret value, for example $\langle\cdot\rangle = \{\langle\cdot\rangle_1, \langle\cdot\rangle_2\}$.
$\langle\cdot\rangle_i^Y$	Yao share belonging to the party i about a secret value.
n_s	Number of samples in the source domain.
n_t	Number of samples in the target domain.
$\mathbf{x}_{s_i}$	The ith sample of the source domain.
y_{s_i}	Sample label corresponded to $\mathbf{x}_{\mathbf{s_i}}$
$\mathbf{x}_{t_i}$	The ith sample of the target domain.
μ	Trade-off parameters.
$\mathbf{W}$	Transformation matrix.
$\mathbf{f}$	Column vector consisted of the coefficients of the matrix characteristic polynomial.
$\mathbf{C}_\lambda$	Eigenvectors corresponded to the eigenvalues λ.
$\mathbf{X}_S$	Matrix consisted of all source domain data, where the ith column vector is the $\mathbf{x}_{s_i}$.
$\mathbf{X}_T$	Matrix consisted of all target domain data, where the ith column vector is $\mathbf{x}_{t_i}$.
S_i^Π	Simulator used to simulate the view of the parties i in the protocol Π.
$\mathbf{S}_i^\Pi$	Simulated view of the party i in the protocol Π.
$\mathbf{R}_i^\Pi$	Real view of the party i during the real execution of the protocol Π.

B. Transfer Component Analysis

The core algorithm used in our privacy-preserving distributed transfer learning (PPDTL) is the transfer component analysis (TCA) algorithm [16]. We first introduce the traditional TCA algorithm in this section and then describe how to construct the corresponding secure computation protocol in section IV.

1) TCA Algorithm: $D_S = \{(x_{s_1}, y_{s_1}), (x_{s_2}, y_{s_2}),\ldots, (x_{s_{n_s}}, y_{s_{n_s}})\}$, $D_T = \{x_{t_1}, x_{t_2}, \ldots, x_{t_n}\}$. Let $D'_S = \{x_{s_1}, x_{s_2}, \ldots, x_{s_{n_s}}\}$. The $\mathcal{P}(D'_S)$ is the marginal distribution of the source domain data, and the $\mathcal{Q}(D_T)$ is the marginal distribution of the target domain data. Normally, the $\mathcal{P}(D'_S)$ and the $\mathcal{Q}(D_T)$ are not equal, so it is not possible to directly use the source domain data and the labels to train the model and use the model to predict the label Y_T corresponding to the target domain data D_T. This is a typical domain adaptation problem [5].

The transfer component analysis (TCA) algorithm [16] can be used to solve this domain adaptation problem. Intuitively,

the TCA algorithm transforms data in two domains originally distributed differently into new data with close distribution by a transformation matrix. The main idea of TCA is to find a map ϕ which can make the marginal distribution of the source and target domain data close after the mapping, i.e., $\mathcal{P}(\phi(D'_S)) \approx \mathcal{Q}(\phi(D_T))$, and try to preserve their respective internal properties. Thus the goal of TCA is twofold: (1) To minimize the MMD distance between $\phi(D'_S)$ and $\phi(D_T)$, and (2) To preserve their respective internal properties to the maximum extent possible. According to the demonstration of Pan et al. [16], the computation tasks of TCA are: Firstly, the central matrix $\mathbf{H}$, the kernel matrix $\mathbf{K}$ and the matrix $\mathbf{L}$ are computed based on the data in the source and target data domains. Then, $(\mathbf{KLK} + \mu\mathbf{I})^{-1}\mathbf{KHK}$ and the first m eigenvectors of this matrix are computed, where the μ is the trade-off parameter. Finally, the transformation matrix $\mathbf{W}$ is constructed using these first m eigenvectors. The workflow is shown in Algorithm 1.

Algorithm 1 TCA Algorithm

Input: Source domain data set $D_S = \{\mathbf{x_{s_i}}, \mathbf{y_{s_i}} | i \in \{1, 2, \ldots, n_s\}\}$, and target domain data set $D_T = \{\mathbf{x_{t_i}} | i \in \{1, 2, \ldots, n_t\}\}$.

Output: The transformation matrix $\mathbf{W}$.

Process:

1: Compute kernel matrix $\mathbf{K}$ from $\{\mathbf{x_{s_i}}\}$, $\{\mathbf{x_{t_i}}\}$, matrix $\mathbf{L}$ and centering matrix $\mathbf{H}$;

2: Compute the matrix $(\mathbf{KLK} + \mu\mathbf{I})^{-1}\mathbf{KHK}$ and it's the m leading eigenvectors to construct the transformation matrix $\mathbf{W}$.

3: **return** $\mathbf{W}$;

The matrix $\mathbf{W}$ is the output of the TCA algorithm, which is called the transformation matrix, and $\mathbf{H}$ is a $(n_s + n_t)$ -dimensional square centering matrix. $\mathbf{H} = \mathbf{I_{n_s+n_t}} - \frac{1}{n_s+n_t} \cdot \mathbf{e} \cdot \mathbf{e}^T$, where $\mathbf{e}$ is an $(n_s + n_t)$-dimensional column vector which all elements are the number 1.

The kernel matrix $\mathbf{K}$ consists of four kernel matrices $\mathbf{K_{SS}}$, $\mathbf{K_{ST}}$, $\mathbf{K_{TS}}$, and $\mathbf{K_{TT}}$. $\mathbf{K_{SS}}$ and $\mathbf{K_{TT}}$ are the kernel matrices of the source and target domains, $\mathbf{K_{ST}}$ and $\mathbf{K_{TS}}$ are the cross-domain kernel matrices, and the kernel functions adopted in this paper are the linear kernels. The specific structure of the kernel matrix is as follows:

$$\mathbf{K_{SS}} = \begin{pmatrix} x_{s_1}^T x_{s_1} & x_{s_1}^T x_{s_2} & \cdots & x_{s_1}^T x_{s_{n_s}} \\ x_{s_2}^T x_{s_1} & x_{s_2}^T x_{s_2} & \cdots & x_{s_2}^T x_{s_{n_s}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{s_{n_s}}^T x_{s_1} & x_{s_{n_s}}^T x_{s_2} & \cdots & x_{s_{n_s}}^T x_{s_{n_s}} \end{pmatrix}, \quad (1)$$

$$\mathbf{K_{ST}} = \begin{pmatrix} x_{s_1}^T x_{t_1} & x_{s_1}^T x_{t_2} & \cdots & x_{s_1}^T x_{t_{n_t}} \\ x_{s_2}^T x_{t_1} & x_{s_2}^T x_{t_2} & \cdots & x_{s_2}^T x_{t_{n_t}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{s_{n_s}}^T x_{t_1} & x_{s_{n_s}}^T x_{t_2} & \cdots & x_{s_{n_s}}^T x_{t_{n_t}} \end{pmatrix}, \quad (2)$$

$$\mathbf{K_{TS}} = \begin{pmatrix} x_{t_1}^T x_{s_1} & x_{t_1}^T x_{s_2} & \cdots & x_{t_1}^T x_{s_m} \\ x_{t_2}^T x_{s_1} & x_{t_2}^T x_{s_2} & \cdots & x_{t_2}^T x_{s_m} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t_n}^T x_{s_1} & x_{t_n}^T x_{s_2} & \cdots & x_{t_n}^T x_{s_m} \end{pmatrix}, \quad (3)$$

$$\mathbf{K_{TT}} = \begin{pmatrix} x_{t_1}^T x_{t_1} & x_{t_1}^T x_{t_2} & \cdots & x_{t_1}^T x_{t_{n_t}} \\ x_{t_2}^T x_{t_1} & x_{t_2}^T x_{t_2} & \cdots & x_{t_2}^T x_{t_{n_t}} \\ \vdots & \vdots & \ddots & \vdots \\ x_{t_{n_t}}^T x_{t_1} & x_{t_{n_t}}^T x_{t_2} & \cdots & x_{t_{n_t}}^T x_{t_{n_t}} \end{pmatrix}, \quad (4)$$

$$\mathbf{K} = \begin{pmatrix} \mathbf{K_{SS}} & \mathbf{K_{ST}} \\ \mathbf{K_{TS}} & \mathbf{K_{TT}} \end{pmatrix}. \quad (5)$$

The $\mathbf{L}$ matrix is used to evaluate the MMD distance together with the $\mathbf{K}$ matrix, whose structure is:

$$L_{ij} = \begin{cases} \frac{1}{n_s^2} & \text{if } i \le n_s, j \le n_t; \\ \frac{1}{n_t^2} & \text{if } n_s \le i \le n_s + n_t, n_s \le j \le n_s + n_t; \\ -\frac{1}{n_s n_t} & \text{otherwise .} \end{cases} \quad (6)$$

In summary, ϕ maps the data from both domains together into a high-dimensional reproducing kernel Hilbert space in which we minimize the data distance between the source domain and the target domain while maximizing the preservation of their respective internal properties, and finally, to obtain a new feature representation in low dimensions, TCA uses a dimensionality reduction method to downscale the mapped data into a low-dimensional subspace. With the new feature representation in this subspace, we can apply standard machine learning methods to train a classifier or regression model in the source domain that can be used in the target domain.

2) Validation of TCA: To validate the TCA algorithm, we performed experiments on the LISA Traffic Light Dataset [51], [52]. We take the sample data in the dayTrain folder of this dataset as the source domain data (training data) and then take the sample data in daySequence1, daySequence2, and daySequence3 as the target domain data (prediction data), respectively. In our experiments, we first extract the features of the sample data, then use KNN algorithm to classify the target domain data. The experimental results are shown in Fig.1, where the blue color represents the accuracy of KNN classification directly on the target domain data, and the orange color represents the accuracy of KNN classification after using the TCA algorithm, which shows that the TCA algorithm improves the accuracy by about 8%-10%.

C. Cryptographic Primitive

1) Secure Multi-Party Computation: Secure multi-party computation (SMPC) is a type of distributed computation that considers a scenario in which two or more parties with

private inputs want to jointly compute a function without revealing their private inputs. Usually, the parties perform the computation task by jointly executing SMPC protocols. The basic requirement of SMPC protocols is to satisfy correctness and privacy, i.e., it is required that even if there is an adversary attack, the honest party can obtain the correct result. The malicious party and the adversary cannot obtain any additional useful information during the computation. Other security requirements are fairness, input independence and output accessibility, etc.

To formally describe the security definition of MPC, let x, y be the privacy inputs of the parties, and n be the length of x, y, and let the deterministic function $f = (f_1, f_2)$ be the function to be computed by the protocol Π, where f_1, f_2 are the outputs of the parties P_1, P_2, respectively. The real view of the protocol Π is noted as $\mathbf{View}_i^{\Pi}(x, y, n), i \in \{1, 2\}$, which contains $(w, r, m_1, m_2, \ldots, m_t)$, where $w \in \{x, y\}$ is the input of the party P_i, m_j, $j \in \{1, t\}$ is the jth message that the party P_i received. Since our protocol uses the semi-honest adversary model [53], we only recall the definition of security in the semi-honest adversary model.

Definition 1: In the semi-honest adversary model. We say that the protocol Π securely computes f, if there exist two polynomial algorithms S_1, S_2 satisfying the following equation.

$$\left\{S_1(1^n, x, f_1(x, y))\right\}_{x,y,n} \stackrel{c}{\equiv} \left\{View_1^{\Pi}(x, y, n)\right\}_{x,y,n}. \quad (7)$$

$$\left\{S_2(1^n, y, f_2(x, y))\right\}_{x,y,n} \stackrel{c}{\equiv} \left\{View_2^{\Pi}(x, y, n)\right\}_{x,y,n}. \quad (8)$$

2) Yao's Protocol: Yao's protocol [54] is a secure two-party computation protocol proposed by Yao in 1986. It is known that for any function f, there exists a boolean circuit C corresponding to it. Yao's protocol implements a general secure two-party computation for arbitrary functions using cryptographic techniques such as garbled circuits and oblivious transfer (OT). Through Yao's protocol, two parties can securely perform the computation of any boolean circuit without revealing any private input information.

3) Secret Sharing: The basic idea of the secret sharing technique is to split a secret value into several pieces and distribute them to different parties. Only if more than a threshold value t parties contribute their shares together the secret value could be revealed. Otherwise, the parties cannot reveal any useful information about the secret value. Our protocol uses two types of secret sharing: Arithmetic sharing and Yao sharing.

a) Arithmetic sharing: The secret value is equal to the sum of all shares, i.e., for a secret value x, we have $x \equiv \langle x\rangle_1 + \langle x\rangle_2 \ mod\ p$, where $x, \langle x\rangle_1, \langle x\rangle_2 \in Z_p$. Both the addition and constant multiplication of arithmetic secret sharing can be computed locally by the parties, specifically, $\langle x + y\rangle$ can be obtained computing $\langle x + y\rangle_i = \langle x\rangle_i + \langle y\rangle_i \ mod\ p$ locally, and $\langle cx\rangle$ (c is a public constant) can be obtained by computing $\langle cx\rangle_i \equiv c\langle x\rangle_i \ mod\ p$ locally. The multiplication of arithmetic secret shares needs to be computed interactively by the parties, and the most common technique used today is the beaver triples [55], which allows the parties to precompute in the offline phase, thus reducing the round and computational complexity of secure multiplication in the online phase.

b) Yao sharing: Yao secret sharing can be considered as a special way of secret sharing [56]. The most significant advantage of Yao secret sharing over the arithmetic secret sharing is that it can compute arbitrary functions. We assume that P_1 is the generator of the garbled circuit, P_2 is the evaluator of the circuit, x is the secret value on wire w in the garbled circuit, the share of P_1 is the wire label $\langle x\rangle_1^Y = k_w^0$ on the wire corresponding to 0, and the share of p_2 is the wire label corresponding to x, i.e., $\langle x\rangle_2^Y = k_w^x$. Only if we know both shares of x can we reconstruct x. Since any function can be represented by a circuit, Yao secret sharing can compute any function, i.e., for any function $f(x, y)$, the circuit generator P_1 constructs the corresponding garbled circuit GC, and the circuit evaluator P_2 can evaluate the garbled circuit layer by layer using the shares he holds, and finally reveals the output of the circuit together with P_1.

c) Sharing types Conversion: Demmler et al. [56] propose the ABY framework, which enables efficient conversion between three different types of sharing, arithmetic sharing (A), Boolean sharing (B), and Yao sharing (Y), in a two-party secure computing scenario. The conversion protocols include $A2B, B2A, A2Y, Y2A, B2Y, Y2B$. (Note that The arithmetic secret sharing is written as $\langle x\rangle_i^A$, but most of the secret sharing used in this paper is arithmetic secret sharing, so we use the shorthand form $\langle x\rangle_i$ to replace $\langle x\rangle_i^A$). In this paper, we only ues A2Y and Y2A.

D. Secure Linear Algebra Operations

In our main protocol, the following secure computation protocols for are linear algebra operations used:

1) Secure Multiplication: In the secure multiplication protocol, two parties can securely compute the multiplication without revealing any private information. That is, P_1 and P_2 take $(\langle x\rangle_1, \langle y\rangle_1)$ and $(\langle x\rangle_2, \langle y\rangle_2)$ as input and get the output $\langle x \cdot y\rangle_1$ and $\langle x \cdot y\rangle_2$ respectively. In this paper, we uses the protocol Π_{MUL} proposed by Beaver [55]. In Π_{MUL}

2) Secure Matrix Multiplication: In the secure matrix multiplication protocol, two parties can securely compute the matrix multiplication without revealing any private information. That is, P_1 and P_2 take $(\langle \mathbf{A}\rangle_1, \langle \mathbf{B}\rangle_1)$ and $(\langle \mathbf{A}\rangle_2, \langle \mathbf{B}\rangle_2)$ as input and get the output $\langle \mathbf{A} \cdot \mathbf{B}\rangle_1$ and $\langle \mathbf{A} \cdot \mathbf{B}\rangle_2$ respectively.

The constant-round secure matrix multiplication protocol Π_{SMM} can be obtained by two parties parallel calling Π_{MUL}. Therefore the security of Π_{SMM} is guaranteed by that of Π_{MUL}. The details shown in the Fig. 2. Since the element multiplication in Π_{SMM} can be computed in parallel, the number of interaction rounds in Π_{SMM} is independent of the size of the matrix.

3) Secure Matrix Inversion: In this paper, we uses the protocol Π_{SMI} proposed by Cramer et al. [57]. We suppose that $\mathbf{A}$ is an invertible matrix and the parties hold shares $\langle \mathbf{A}\rangle_1$ and $\langle \mathbf{A}\rangle_2$ respectively. The main steps of the security matrix inversion are as follows: first, the parties randomly select a matrix $\langle \mathbf{R}\rangle_i$ as the share of $\mathbf{R}$ in the linear space $P^{n\times n}$, and second, use the security matrix multiplication protocol to

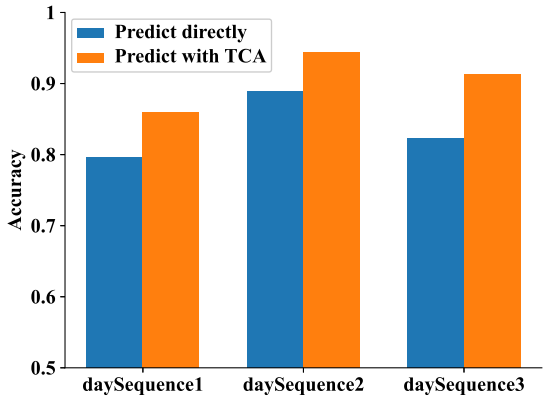


Fig. 1. Performance of TCA algorithm.

Secure Matrix Multiplication Π_{SMM}

- **Inputs:**
  -P_1 inputs $\langle \mathbf{A} \rangle_1, \langle \mathbf{B} \rangle_1$;
  -P_2 inputs $\langle \mathbf{A} \rangle_2, \langle \mathbf{B} \rangle_2$.
  Where $\mathbf{A}$ is an $m*n$ matrix and $\mathbf{B}$ is an $n*k$ matrix.
- **Protocol:**
  1) Both parties jointly invoke Π_{MUL} in parallel to compute $\langle \mathbf{A}[i][j] \rangle * \langle \mathbf{B}[j][h] \rangle$, $i \in \{0, m-1\}$, $j \in \{0, n-1\}$, $h \in \{0, k-1\}$;
  2) Both parties compute the $\langle \sum_{j=0}^{n-1} \mathbf{A}[i][j] * \mathbf{B}[j][h] \rangle$, $i \in \{0, m-1\}$, $h \in \{0, k-1\}$ locally;
  3) Let $\langle (\mathbf{A} \cdot \mathbf{B})[i][k] \rangle = \langle \sum_{j=0}^{n-1} \mathbf{A}[i][j] * \mathbf{B}[j][h] \rangle$, $i \in \{0, m-1\}$,$h \in \{0, k-1\}$.
- **Outputs:**
  -P_S outputs $\langle \mathbf{A} \cdot \mathbf{B} \rangle_1$;
  -P_T outputs $\langle \mathbf{A} \cdot \mathbf{B} \rangle_2$.

Fig. 2. The constructions of Π_{SMM}.

compute the share of $\mathbf{R} \cdot \mathbf{A}$ and reveal it. Third, find the inverse of $\mathbf{R} \cdot \mathbf{A}$. Finally, note that $(\mathbf{R} \cdot \mathbf{A})^{-1}$ can be considered as a constant, so both sides perform the multiplication of $\mathbf{R} \cdot \mathbf{A}^{-1}$ and $\langle \mathbf{R} \rangle$ locally to obtain $\langle \mathbf{A}^{-1} \rangle$.

4) Secure Matrix Characteristic Polynomial: In the secure matrix characteristic polynomial protocol, two parties can securely compute the characteristic polynomial of the secret matrix without revealing any private information. In this paper, we uses the protocol Π_{SMCP} proposed by Cramer et al. [57]. We suppose that both parties hold shares of the matrix $\mathbf{A}$ and $\mathbf{f}$ which is a column vector consisting of the characteristic polynomial coefficients of $\mathbf{A}$. Then the main steps for computing $\langle \mathbf{f} \rangle$ are as follows: First let $\mathbf{M}$ be a $(n+1) \times (n+1)$ Vandermonde matrix, where the ith row of $\mathbf{M}$ is $(1, z_i, z_i^2, \ldots, z_i^n)$, and z_i is a randomly selected element in P^n. The $\mathbf{M}$ is the public matrix. Then, each party locally computes the inverse matrix $\mathbf{M^{-1}}$ of the Vandermonde matrix and the $\langle z_0\mathbf{I} - A \rangle, \langle z_1\mathbf{I} - A \rangle, \ldots, \langle z_n\mathbf{I} - A \rangle$. Then, using the method for computing the determinant of the matrix proposed by Cramer et al. to compute $(\langle |z_0\mathbf{I} - A| \rangle, \langle |z_1\mathbf{I} - A| \rangle, \ldots, \langle |z_n\mathbf{I} - A| \rangle)$. Finally, both parties compute

Functionality $\mathcal{F}_{PPDTL}$

Functionality $\mathcal{F}_{PPDTL}$ works with parties P_S and P_T, as follows:

- Upon receiving $\langle \mathbf{X}_S \rangle_S$, $\langle \mathbf{X}_S \rangle_T$ from party P_S, and $\langle \mathbf{X}_T \rangle_S$, $\langle \mathbf{X}_T \rangle_T$ from P_T, in addition to the common input $\mathbf{H}, \mathbf{L}$ and μ.
  1) Compute $\mathbf{X}_S = \langle \mathbf{X}_S \rangle_S + \langle \mathbf{X}_S \rangle_T$, and $\mathbf{X}_T = \langle \mathbf{X}_T \rangle_S + \langle \mathbf{X}_T \rangle_S$.
  2) Invoke the TCA algorithm to compute the kernel matrix $\mathbf{K}$ and the transformation matrix $\mathbf{W}$.
  3) Compute $\mathbf{WK}$, and choose a random matrix $\mathbf{R}$, where $\mathbf{WK}$ is the new data after TCA transformation.
  4) Set $\langle \mathbf{WK} \rangle_S = \mathbf{WK\text{-}R}$ and $\langle \mathbf{WK} \rangle_T = \mathbf{R}$.
  5) Sent $\langle \mathbf{WK} \rangle_S$ to P_S and $\langle \mathbf{WK} \rangle_T$ to P_T.

Fig. 3. The privacy-preserving distributed transfer Learning functionality $\mathcal{F}_{PPDTL}$.

$\langle \mathbf{f} \rangle = \mathbf{M}^{-1}(\langle |z_0\mathbf{I} - A| \rangle, \langle |z_1\mathbf{I} - A| \rangle, \ldots, \langle |z_n\mathbf{I} - A| \rangle)$ locally and output it.

5) Secure Sub-Space Membership Decisions: In this paper, we uses the protocol Π_{SSMD} proposed by Cramer et al. [57]. We suppose that both parties hold a share of the n-dimensional matrix $\mathbf{A}$ and a share of the n-dimensional vector $\mathbf{y}$ (where $\mathbf{y}$ can be a zero vector). Both parties execute Π_{SSMD} together with $\langle \mathbf{A} \rangle$ and $\langle \mathbf{y} \rangle$ as inputs. Then both parties get a share of bits b. When the system of equations has no solution $b = 0$ and vice versa $b = 1$.

6) Secure General Linear Systems: In this paper, we uses the protocol Π_{GLS} proposed by Cramer et al. [57]. We suppose that both parties hold a share of the n-dimensional matrix $\mathbf{A}$ and a share of the n-dimensional vector $\mathbf{y}$. (Where $\mathbf{y}$ can be a zero vector and in this paper we only consider the case where $\mathbf{y}$ is a zero vector.) Both parties execute the Π_{GLS} together with $\langle \mathbf{A} \rangle$ and $\langle \mathbf{y} \rangle$ as inputs. Then both parties obtain the kernel $Ker(\mathbf{A})$ of $\mathbf{A}$ and the share of the special solution $\mathbf{x}$.

IV. Privacy-Preserving Distributed Transfer Learning

A. An Overview of PPDTL Protocol

Based on the secret sharing technique, we propose a privacy-preserving protocol Π_{PPDTL} for distributed transfer learning (PPDTL). It is a standard secure two-party computing protocol in the semi-honest model under the ideal/real-model paradigm [53] and can solve the privacy data leakage problem while running the traditional TCA algorithm. The formal definition of security is provided as Definition 1, and the functionality of PPDLT ($\mathcal{F}_{PPDLT}$) is shown in Fig. 3.

Our privacy-preserving distributed transfer learning protocol Π_{PPDTL} securely computes the $\mathcal{F}_{PPDTL}$ without revealing the private data of each party. Using the original private data as their input, the parties get the share of the transformation matrix $\mathbf{W}$ from Π_{PPDTL} protocol, and then the share of the new data that transformed by the TCA algorithm is computed

securely by using the share $\langle \mathbf{W} \rangle$ and the original private data. Notice that under plaintext, the TCA algorithm ends after computing the transformation matrix $\mathbf{W}$, whereas our protocol needs to compute the new data after the transformation. The reason is that under the plaintext, with $\mathbf{W}$ and the kernel matrix $\mathbf{K}$, the user can compute the new feature representation locally without interaction, but in our protocol, both parties can only hold the share of the kernel matrix, not the plaintext of kernel matrix, so it is meaningless to just compute $\mathbf{W}$. Therefore, our protocol requires additional steps to compute the transformed new data securely.

Our main computation task is to securely compute the first m eigenvectors of the square matrix $(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}$, and this task can be divided into the following sub-tasks:

- the product of two matrices;
- the sum of two matrices;
- the product of a constants and a matrix;
- the inverse of a invertible matrix;
- the characteristic polynomials of a square matrix;
- the eigenvalues of a square matrix;
- the eigenvectors of a square matrix.

In order to perform these computation tasks securely, our protocol uses secret sharing techniques, which require users to split their private data into shares and distribute them to other parties. Leveraging the arithmetic secret sharing property, the parties can compute the shares of the matrix addition and the shares of the constant multiplication locally. For the computation tasks of multiplication, inversion and characteristic polynomials of matrices, we use the technique of Cramer et al. [57], which achieves secure computation in a constant number of rounds. For the computation task of evaluating the eigenvalues of a square matrix, we use Yao's garbled circuit [54]. For computing the eigenvectors of a square matrix, we design a new sub-protocol Π_{SMEV} that can securely compute the eigenvectors of a square matrix in a constant number of rounds.

To make it easier to understand, we divide the main protocol into three modules: a. the secure characteristic value sub-protocol Π_{SCV} (as shown in Fig. 5), which securely computes the matrix $(\mathbf{KLK}+\mu\mathbf{I})^{-1}KHK$ and its top m most significant characteristic values; b, the secure transformation matrix sub-protocol Π_{STM} (as shown in Fig. 8.), which securely computes the output matrix $\mathbf{W}$ of TCA algorithm; c, the secure matrix multiplication sub-protocol Π_{SMM} (as shown in Fig. 2.), which securely computes the product of two matrices.

In the main protocol Π_{PPDTL}, two parties first set the arithmetic secret sharing shares for their privacy inputs. Then both parties distribute the shares to each other and invoke the three sub-protocols Π_{SCV}, Π_{STM}, Π_{SMM} in sequence to obtain the shares of new data. The details of Π_{PPDTL} are described in section IV-C.

B. The Basic Sub-Protocols

1) The Secure Characteristic Value Computing Protocol: In this paper, we desire a secure characteristic value computing protocol Π_{SCV}, which securely computes the $\mathcal{F}_{SCV}$, i.e., it securely computes shares of the top m characteristic values of the matrix $(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}$ and sorts them.

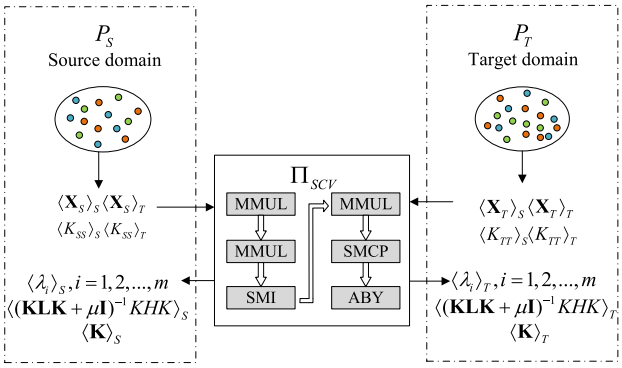


Fig. 4. The structure of Π_{SCV}.

Note that the matrix $\mathbf{X}_S$ consists of all x_{s_i} which denotes the source domain dataset, the matrix $\mathbf{X}_T$ consists of all x_{t_i} which denotes the target domain dataset, and $\mathbf{K}_{ST} = \mathbf{X}_S^T \mathbf{X}_T$. The details of Π_{SCV} are described in Fig. 5, and the structure of the Π_{SCV} is shown in Fig. 4, where P_S holds the source domain data and P_T holds the target domain data. In the execution of Π_{SCV}, the parties need to invoke the sub-protocols Π_{SMM}, Π_{SMI}, Π_{SMM}, Π_{SMCP}, Π_{ABY} in the order as the Fig. 4 shows.

2) The Secure Matrix Eigenvector Computing Protocol: In this paper, we desire a protocol Π_{STM}, intuitively, both parties take the shares of square matrices $\mathbf{A}$ and an eigenvalue λ of $\mathbf{A}$ as input and first mutually invoke the sub-protocol Π_{GLS} to compute the set of linear solutions $\lambda\mathbf{I}-\mathbf{A}$ of the matrix $\lambda\mathbf{I}-\mathbf{A}$, (i.e. $\langle \mathbf{u}_0, \mathbf{u}_1, \ldots, \mathbf{u}_n \rangle$). It is notable that these n vectors certainly include the fundamental solution system of $(\lambda\mathbf{I}-\mathbf{A})x = 0$, so we invoke the $n-1$ sub-protocols of Π_{SSMD} in parallel to determine whether the vector $\mathbf{u}_i$ can be represented linearly by the first $i-1$ vectors $\mathbf{u}_1$, $\mathbf{u}_2$,..., $\mathbf{u}_{i-1}$. That is, decide ($\mathbf{u}_1$, $\mathbf{u}_2$,..., $\mathbf{u}_{i-1}$)$x = \mathbf{u}_i$ has a solution or not. If it has a solution, $\mathbf{u}_i$ is discarded; on the contrary, $\mathbf{u}_i$ is reserved. Because if the above equation has a solution, it means that $\mathbf{u}_i$ is linearly dependent on the previous vectors, and only the vectors that are linearly independent of the other vectors can compose the fundamental solutions. The details of Π_{SMEV} are described in Fig. 6.

3) The Secure Transformation Matrix Computing Protocol: The protocol Π_{STM} invokes m Π_{SMEV} protocols in parallel to receive the shares of k ($k \geq m$) eigenvectors. Then based on the order of the corresponding eigenvalues, both parties reserve the first m shares of these k eigenvectors and combine these m shares into a transformation matrix $\langle \mathbf{W} \rangle$ and output it. The details of Π_{STM} is described in Fig. 8, and the structure of the Π_{STM} is shown in Fig. 7. In the execution of Π_{STM}, the parties need to invoke m Π_{SMEV} in parallel as Fig. 7 shows.

C. The Details of PPDTL Protocol

Next, we introduce the detail of privacy-preserving distributed transfer learning protocol Π_{PPDTL}.

In presence of a semi-honest adversary, the Π_{PPDTL} securely computes the $\mathcal{F}_{PPDTL}$. We assume that the parties

The Secure Characteristic Value sub-protocol Π_{SCV}

- **Inputs:**
  -common inputs: $\mathbf{H}$, $\mathbf{L}$;
  -P_S inputs $\langle \mathbf{X}_S \rangle$, $\langle K_{SS} \rangle$;
  -P_T inputs $\langle \mathbf{X}_T \rangle$, $\langle K_{TT} \rangle$.
- **Protocol:**
  1) $\mathcal{S}$ sends $\langle \mathbf{X}_S \rangle_T$ and $\langle K_{SS} \rangle_T$ to $\mathcal{T}$, $\mathcal{T}$ also sends $\langle \mathbf{X}_T \rangle_S$ and $\langle K_{TT} \rangle_S$ to $\mathcal{S}$. Then both parties invoke **matrix multiplication sub-protocol** with $\langle \mathbf{X}_T \rangle$ and $\langle \mathbf{X}_S \rangle$ as input. Finally $\mathcal{S}$ output $\langle \mathbf{K}_{ST} \rangle_S$, $\mathcal{T}$ output $\langle \mathbf{K}_{ST} \rangle_T$.
  2) Each party computes $\langle \mathbf{K} \rangle$ and the shares of $\mathbf{KL}$ and $\mathbf{KH}$ locally, and then takes $\langle \mathbf{KL} \rangle, \langle \mathbf{K} \rangle$, and $\langle \mathbf{KH} \rangle$ as input to call **matrix multiplication sub-protocol**. $\mathcal{S}$ outputs $\langle \mathbf{KLK} \rangle_S$ and $\langle \mathbf{KHK} \rangle_S$, $\mathcal{T}$ outputs $\langle \mathbf{KLK} \rangle_T$ and $\langle \mathbf{KHK} \rangle_T$.
  3) Each party computes their own share of $\langle \mathbf{KLK} + \mu\mathbf{I} \rangle_S$ or $\langle \mathbf{KLK} + \mu\mathbf{I} \rangle_T$ locally and takes it as input to mutually invoke **matrix inverse sub-protocol**, then $\mathcal{S}$ outputs $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \rangle_S$, $\mathcal{T}$ output $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \rangle_T$
  4) Both parties take $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \rangle$ and $\langle \mathbf{KHK} \rangle$ as input and mutually invoke **matrix multiplication sub-protocol**. Then $\mathcal{S}$ outputs $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \mathbf{KHK} \rangle_S$ and $\mathcal{T}$ outputs $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \mathbf{KHK} \rangle_T$
  5) Both parties take $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} KHK \rangle_S$ and $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} \mathbf{KHK} \rangle_T$ as input and mutually invoke **feature polynomial coefficient sub-protocol**. Then $\mathcal{S}$ output $\langle \mathbf{f} \rangle_S$and $\mathcal{T}$ output $\langle \mathbf{f} \rangle_T$
  6) Both parties take $\langle \mathbf{f} \rangle_S$ and $\langle \mathbf{f} \rangle_T$ as input and mutually invoke **A2Y sub-protocol**, then they mutually invoke **Yao protocol** to get the Yao share of top m characteristic values of matrix $(\mathbf{KLK} + \mu\mathbf{I})^{-1} KHK$. Finally both parties invoke **Y2A sub-protocol** to obtain the arithmetic share of the top m characteristic values of $(\mathbf{KLK} + \mu\mathbf{I})^{-1} KHK$.
- **Outputs:**
  -P_S outputs $\langle \lambda_i \rangle_S, i = 1, 2, ..., m, \langle \mathbf{K} \rangle$, and $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} KHK \rangle_S$;
  -P_T outputs $\langle \lambda_i \rangle_T, i = 1, 2, ..., m, \langle \mathbf{K} \rangle$, and $\langle (\mathbf{KLK} + \mu\mathbf{I})^{-1} KHK \rangle_T$.

Fig. 5. The details of Π_{SCV}.

have precomputed a sufficient number of beaver triples, and computed the corresponding kernel matrices $\mathbf{K_{SS}}$ and $\mathbf{K_{TT}}$ based on their own data, and that both parties know the public value center matrix $\mathbf{H}$, and the matrix $\mathbf{L}$.

In the protocol Π_{PPDTL}, both parties take their private data and kernel matrices as inputs and call the three modules Π_{SCV}, Π_{STM}, Π_{SMM} in sequence to compute the shares of the new data $\mathbf{W}^T \mathbf{K}_S$, and each party receives only its share of

Secure Matrix Eigenvector sub-protocol Π_{SMEV}

- **Inputs:**
  -P_S inputs $\langle \mathbf{A} \rangle_S$, $\langle \lambda \rangle_S$;
  -P_T inputs $\langle \mathbf{A} \rangle_T$, $\langle \lambda \rangle_T$.
- **Protocol:**
  1) Both parties locally compute $\langle \lambda\mathbf{I} - \mathbf{A} \rangle$.
  2) Both parties take $\langle \lambda\mathbf{I} - \mathbf{A} \rangle$, $\mathbf{0}$ as input and mutually invoke General Linear Systems Protocol Π_{GLS}, then both parties obtain $\langle Ker(\mathbf{A}) \rangle$, i.e. $\langle \mathbf{u}_0, \mathbf{u}_1, ..., \mathbf{u}_n \rangle$
  3) Both parties take $\langle Ker(\mathbf{A}) \rangle$ as input , and determine whether the vector $\mathbf{u}_i$ can be linearly represented by the first $i-1$ vectors $\mathbf{u}_1$, $\mathbf{u}_2$,... , $\mathbf{u}_{i-1}$ through invoking $n-1$ Π_{SSMD} in parallel. Let $b_1 = 0$ and let the output of the ith Π_{SSMD} be $\langle b_{i+1} \rangle$
  4) Both parties reveal b_i,and construct the matrix $\langle \mathbf{C} \rangle$. The column vectors of the matrix $\langle \mathbf{C} \rangle$ are all $\langle \mathbf{u}_i \rangle$ corresponding to $b_i = 0$.
- **Outputs:**
  -P_S outputs $\langle \mathbf{C} \rangle_S$;
  -P_T outputs $\langle \mathbf{C} \rangle_T$.

Fig. 6. The details of Π_{SMEV}.

each data during the whole computation phase. Neither party learns any additional information except for its own input and output. The details are shown in Fig. 9, and the structure of the main protocol as shown in Fig. 10, where P_S holds the source domain data and P_S holds the source domain data. In the execution of Π_{PPDTL}, the parties need to invoke Π_{SCV}, Π_{STM}, Π_{SMM} in sequence as the Fig. 10 shows.

It is worth noting that the main protocol sequence performs three sub-protocols that can be completed in a constant number of rounds, so the main protocol is a constant-round protocol. Furthermore, the computation, communication and interaction complexity of the main protocol is independent of the feature number of samples. Thus this protocol is particularly suitable for the data extracted features by deep learning.

V. Security Proof

In this section, we provide a security proof sketch of PPDTL protocol in the (Π_{SCV}, Π_{STM})-hybrid model [53]. Due to the limited space, the formal security proofs of these sub-protocols and PPDTL protocol are given in the full version.

Theorem 1: In presence of a semi-honest adversary, the PPDTL protocol securely computes the $\mathcal{F}_{PPDTL}$ in the (Π_{SCV}, Π_{STM}, Π_{SMM})-hybrid model.

A. Correctness Analysis

Because the Π_{PPDTL} consists entirely of Π_{SCV}, Π_{STM}, and Π_{SMM}, the correctness of the Π_{PPDTL} is guaranteed by these sub-protocols. The correctness of these above sub-protocols have been proved in the full version. Thus Π_{PPDTL} correctly computes the $\mathcal{F}_{PPDTL}$

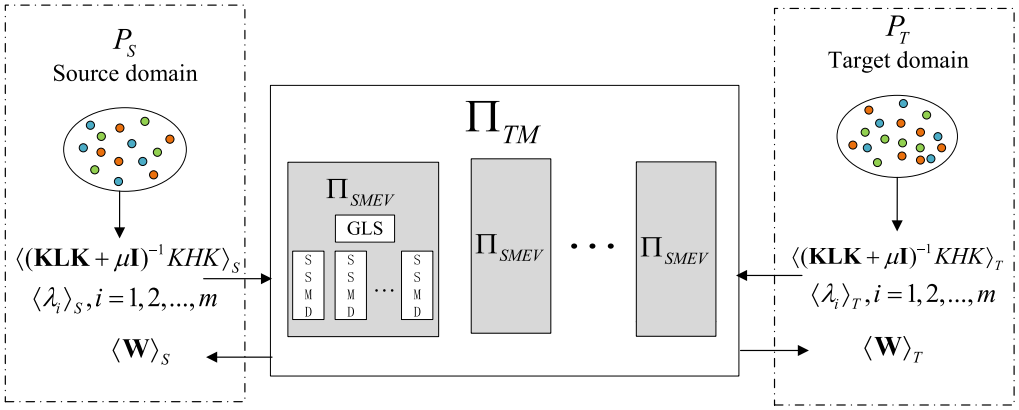


Fig. 7. The structure of Π_{STM}.

Transformation Matrix sub-protocol Π_{STM}

- **Inputs:**
  -P_S inputs $\langle\lambda_i\rangle_S, i = 1,2,...,m, \langle(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}\rangle_S$;
  -P_T inputs $\langle\lambda_i\rangle_T, i = 1,2,...,m, \langle(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}\rangle_T$.
- **Protocol:**
  1) Both parties take the shares $\langle(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}\rangle$ and $\langle\lambda_i\rangle, i = 1,2,...,m$ as input, and mutually invoke m Π_{SMEV} in parallel to get the share of $k((k \geq m))$ eigenvectors i.e. $\langle\mathbf{V}_{\lambda_{ij}}\rangle, j = 1,2,...,k$. The $\mathbf{V}_{\lambda_{ij}}$ means that the jth eigenvector of the eigenvalue λ_i
  2) Both sides preserve the shares of the first m eigenvectors in order, and combine these shares into a matrix $\langle\mathbf{W}\rangle$
- **Outputs:**
  -P_S outputs $\langle\mathbf{W}\rangle_S$;
  -P_T outputs $\langle\mathbf{W}\rangle_T$.

Fig. 8. The details of Π_{STM}.

Privacy-Preserving Distributed Transfer Learning protocol Π_{PPDTL}

- **Inputs:**
  -P_S inputs $\mathbf{X}_S$;
  -P_T inputs $\mathbf{X}_S$.
- **Initialize:**
  -$\mathcal{S}$ set the shares$\{\langle\mathbf{X}_S\rangle_S, \langle\mathbf{X}_S\rangle_T\}$, $\langle K_{SS}\rangle_S$, $\langle K_{SS}\rangle_T$. $\mathcal{T}$ set the shares$\{\langle\mathbf{X}_T\rangle_S, \langle\mathbf{X}_T\rangle_T\}$, $\langle K_{TT}\rangle_T$, $\langle K_{TT}\rangle_S$.
- **Protocol:**
  1) Both parties take $\langle\{x_{s_i}\rangle_S | i \in \{1,2,...,n\}\}$, K_{SS}, $\mathbf{H}$, and $\mathbf{L}$ as their inputs, and mutually invoke Π_{SCV}, then output $\langle\lambda_i\rangle, i = 1,2,...,m$, $\langle\mathbf{K}\rangle$ and $\langle(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}\rangle$;
  2) Both parties take the $\langle\lambda_i\rangle, i = 1,2,...,m$ and $\langle(\mathbf{KLK}+\mu\mathbf{I})^{-1}\mathbf{KHK}\rangle$ as input and mutually invoke Π_{STM} to obtain the matrix share $\langle\mathbf{W}\rangle$;
  3) Both parties take $\langle\mathbf{W}\rangle$ and the $\langle\mathbf{K}\rangle$ obtained in the execution of Π_{SCV} as their inputs and mutually invoke Π_{SMM} to obtain the output $\langle\mathbf{W}^T\mathbf{K}_S\rangle$.
- **Outputs:**
  -P_S outputs $\langle\mathbf{W}^T\mathbf{K}_S\rangle_S$;
  -P_T outputs $\langle\mathbf{W}^T\mathbf{K}_S\rangle_T$.

Fig. 9. The details of Π_{PPDTL}.

B. Security Analysis

Proof sketch: Our security proof follows the ideal/real simulation paradigm [53]. To prove the security of PPDTL in the (Π_{SCV}, Π_{STM})-hybrid model, we need to analyze it in two cases, i.e., the security of Π_{PPDTL} in the case of corrupted P_S and in the case of the corrupted P_T. In each case, we first start the proof by constructing a simulator for Π_{PPDTL}. Next, we prove that the distribution of the output of the simulator is indistinguishable from that of the view of real execution. Thus, we prove the security of Π_{PPDTL} in the (Π_{SCV}, Π_{STM})-hybrid model.

We analyze the security of Π_{PPDTL} in the following two cases.

Case 1: the P_S was corrupted.

We first construct a simulator $S_S^{\Pi_{PPDTL}}$, and take the $\mathbf{X}_S$, $\langle\mathbf{W}^T\mathbf{K}\rangle_S$ as its input. The simulator $S_S^{\Pi_{PPDTL}}$ works as follows:

1. The simulator $S_S^{\Pi_{PPDTL}}$ initializes the data to get the share $\langle\mathbf{X}_S\rangle_S$ and $\langle\mathbf{X}_S\rangle_T$.

2. The simulator $S_S^{\Pi_{PPDTL}}$ invokes the simulator $S_S^{\Pi_{SCV}}$ which have been constructed in the full version with the $\langle\mathbf{X}_S\rangle$, two random martrix $\mathbf{R_1}, \mathbf{R_2}$ and a set of random numbers $\lambda_i', i = 1,2,\ldots,m$ as. input, and obtains the output of $S_S^{\Pi_{SCV}}$.

3. The simulator $S_S^{\Pi_{PPDTL}}$ invokes the simulator $S_S^{\Pi_{STM}}$ which have been constructed in the full version with $\lambda_i', i = 1, 2, \ldots, m$ and two random martrix $\mathbf{R_2}, \mathbf{R_3}$ as input, and obtains the output of $S_S^{\Pi_{STM}}$.

4. The simulator $S_S^{\Pi_{PPDTL}}$ invokes the simulator $S_S^{\Pi_{SMM}}$ with two random martrix $\mathbf{R_1}$, $\mathbf{R_3}$, and the $\langle \mathbf{W}^T\mathbf{K}\rangle_S$ as input, and obtains the output of $S_S^{\Pi_{SMM}}$

We now prove that the distribution of the output of $S_S^{\Pi_{PPDTL}}$ is statistically indistinguishable from the distribution of the real view of Π_{PPDTL}.

The real view of the protocol for P_S is noted as:

$$\begin{aligned}&\mathbf{View}_S^{\Pi_{PPDTL}}(\mathbf{X}_S, \mathbf{X}_T)\\ &= \Big\{\langle(\mathbf{X}_S, \langle\mathbf{W}^T\mathbf{K}\rangle_S, \mathbf{R}_S^{SCV}\big(\langle\mathbf{X}_S\rangle, \langle\mathbf{X}_T\rangle\big),\\ &\quad \mathbf{R}_S^{STM}\big(\langle\lambda_i\rangle, i = 1, 2, \ldots, m,\\ &\quad \langle(\mathbf{KLK} + \mu\mathbf{I})^{-1}\mathbf{KHK}\rangle\big),\\ &\quad \mathbf{R}_S^{SMM}\big(\langle\mathbf{W}\rangle, \langle\mathbf{K}\rangle\big), \Big\}. \end{aligned} \tag{9}$$

The output of the simulator $\mathbf{S}_S^{\Pi_{PPDTL}}$ is noted as:

$$\begin{aligned}&\mathbf{S}_S^{\Pi_{PPDTL}}(\mathbf{X}_S, \langle\mathbf{W}^T\mathbf{K}\rangle_S)\\ &= \Big\{\langle(\mathbf{X}_S, \mathbf{W}^T\mathbf{K}_S, \mathbf{S}_S^{SCV}\big(\langle\mathbf{X}_S\rangle_S, \langle\mathbf{X}_T\rangle_S,\\ &\quad \lambda_i', i = 1, 2, \ldots, m, \mathbf{R_1}, \mathbf{R_2}\big),\\ &\quad \mathbf{S}_S^{STM}\big(\lambda_i', i = 1, 2, \ldots, m, \mathbf{R_2}, \mathbf{R_3}\big),\\ &\quad \mathbf{S}_S^{SMM}\big(\mathbf{R_1}, \mathbf{R_3}, \langle\mathbf{W}^T\mathbf{K}\rangle_S\big)\Big\}. \end{aligned} \tag{10}$$

Our goal is to prove that:

$$\mathbf{View}_S^{\Pi_{PPDTL}}(\mathbf{X}_S, \mathbf{X}_T) \stackrel{c}{\equiv} \mathbf{S}_S^{\Pi_{PPDTL}}(\mathbf{X}_S, \langle\mathbf{W}^T\mathbf{K}\rangle_S). \tag{11}$$

Since simulators $S_S^{\Pi_{SCV}}$,$S_S^{\Pi_{STM}}$ and $S_S^{\Pi_{SMM}}$ have been constructed in the full version, we have that:

$$\begin{aligned}\mathbf{S}_S^{SCV} &\stackrel{c}{\equiv} \mathbf{R}_S^{SCV}.\\ \mathbf{S}_S^{STM} &\stackrel{c}{\equiv} \mathbf{R}_S^{STM}.\\ \mathbf{S}_S^{SMM} &\stackrel{c}{\equiv} \mathbf{R}_S^{SMM}.\end{aligned} \tag{12}$$

So we conclude that Π_{PPDTL} securely computes the $\mathcal{F}_{PPDTL}$ in the case that P_S is corrupted.

Case 2: P_T is corrupted.

The proof procedure for this case is exactly the same as that for the first case, so we omit it here.

At this point, we have completed the proof of security for PPDTL protocol.

VI. Application in Intelligent Transportation Systems

In this section, we will demonstrate how to apply our PPDTL in intelligent transportation systems.

Short-term traffic forecasting is one of the important applications in the field of intelligent transportation, especially for map service providers. Accurate short-term traffic forecasting can bring a perfect customer experience. However, the lack of labeled traffic data is a common problem for almost every map service provider. An intuitive solution is to train the model with the help of transfer learning and the labeled data (historical data) mastered by the traffic management department.

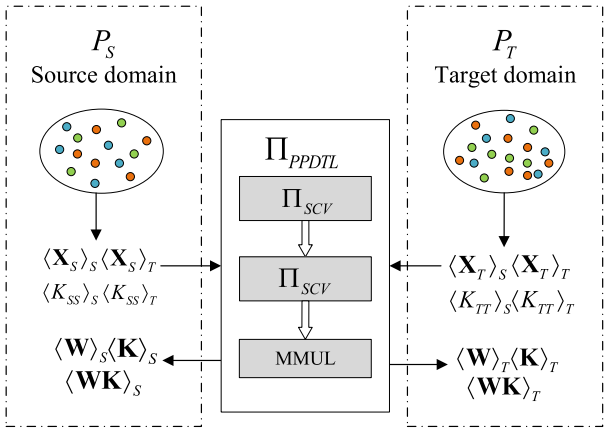


Fig. 10. The structure of the Π_{PPDTL}.

We consider a scenario where a map service provider wants to forecast the current traffic situation at a certain location. However, it does not have enough labeled traffic data to train the model. At the same time, the Traffic Data Center has a lot of labeled historical traffic data for the area. Due to the special properties of traffic data, the distribution of historical traffic data may be different from the distribution of newly collected data, so it is not useful to train the model directly using historical data. However, these labeled data can be used as the source domain data, and the unlabeled data collected by the map service provider can be used as the target domain data to transfer learning, so that the problem of insufficient available data can be solved. However, traffic data contains a large amount of sensitive information and commercial values. Therefore the direct use of traditional transfer learning suffers from the problem of leaking private data of both parties. This problem can be solved if the PPDTL which we designed is employed. In the following, we elaborate on how to deploy PPDTL in this scenario.

In the case of traffic forecasting, many methods are based on the combination of CNNs and LSTMs [58], [59]. The CNN layers extract high-level features [58], [60], [61], and the LSTMs are utilized to mine the temporal features. In our deployment, the traffic data center and the map service provider proceed as follows, and a graphical illustration is shown in Fig. 11.

1) The traffic data center and The map service provider input their own data into the CNN layer locally to extract high-dimensional features.
2) The traffic data center takes its own data after feature extraction as the source domain data, and the map service provider takes its own data after feature extraction as the target domain data. Then they jointly execute the PPDTL protocol to perform privacy-preserving transfer learning, and finally get their own outputs $\langle\mathbf{WK}\rangle_S$ and $\langle\mathbf{WK}\rangle_T$.

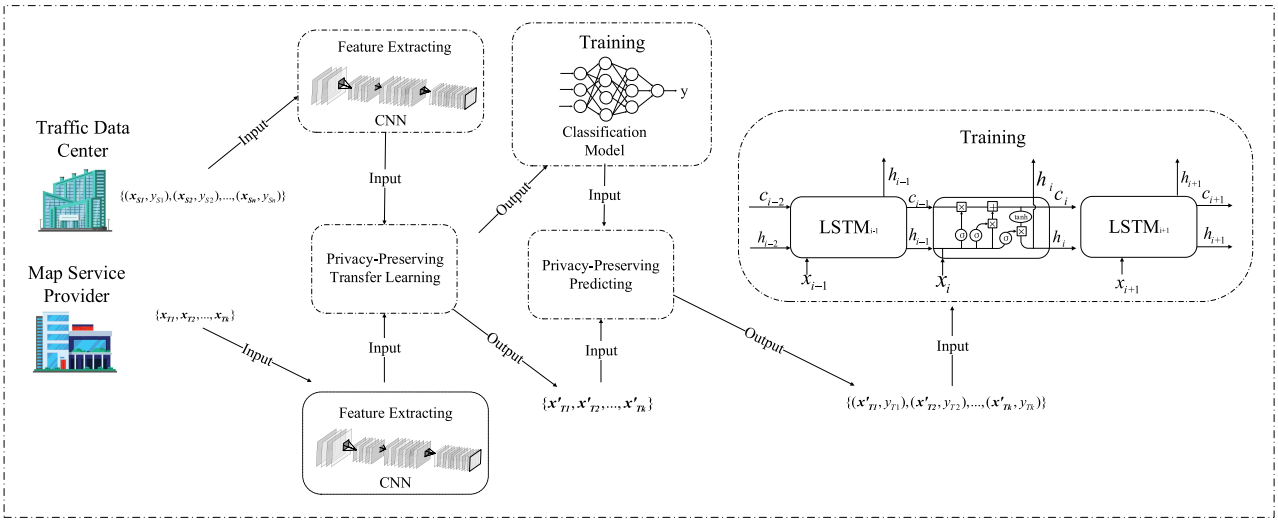


Fig. 11. Deployment in the intelligent transportation system.

3) The map service provider sends the first n columns of $\langle \mathbf{WK} \rangle_T$ to the traffic data center, and the traffic data center sends the last k columns of $\langle \mathbf{WK} \rangle_S$ to the map service provider, where n and k respectively are the number of samples in the source domain and the target domain. Since the first n columns of $\mathbf{WK}$ are the source domain data transformed by PPDTL, and the last k columns are the transformed target domain data, both parties can only reveal the transformed data that belong to themselves.
4) The traffic data center uses the data revealed from the previous step as a training set to train a classification model, which is used to label the transformed data of the map service provider.
5) The traffic data center and map service provider jointly perform the privacy-preserving prediction, where we use the MiniONN [62] approach, which can securely perform the prediction without revealing the private model parameters and the private predicted data.
6) The map service provider uses the newly labeled data as training data to perform the training of the traffic forecasting model.

In the above scenario, the map service provider collects new unlabeled traffic data every day. When it wants to use this new data, it will face the problem of labeling the data again. If transfer learning had to be performed again each time we labeled new data, the overhead would be unaffordable. Fortunately, our PPDTL supports the out-of-sample extension, which can efficiently process new samples. When the new traffic data $\mathbf{x_{k+i}}$ appears, both parties just need to securely compute $\mathbf{W x_{k+i}}$ to get the new data transformed by PPDTL, and this overhead is very economical.

In addition, in the scenario of traffic forecasting, it is often required to extract high-dimensional features of the data by using CNN layers. However, many transfer learning methods become very inefficient when the dimensionality of the data is too high. Fortunately, our PPDTL protocol is particularly suitable for processing high-dimensional data, which has very little impact on the performance of PPDTL.

VII. Experiment

A. Experiment Setting

We use the MP-SPDZ library [63] to simulate the protocol on the cloud server with a 64-core CPU and 256 GB RAM. Specifically, since both parties can generate sufficient beaver triples in the offline phase, we only test the running time in the online phase.

We first validate the PPDTL protocol on a real dataset: LISA Traffic Light Dataset [51], [52], which consists of continuous test and training video sequences, totaling 43,007 frames and 113,888 annotated traffic lights. Then we also study the efficiency of our protocol on synthetic datasets. Since most of the sub-protocols in this work, such as Π_{SMCP}, Π_{STM}, Π_{SMI}, are built based on the Π_{SMM} protocol. Therefore, the efficiency of the Π_{SMM} protocol determines the efficiency of the PPDTL. So we test the performance of Π_{SMM} first. Then, in the testing of the PPDTL protocol, we use 50 threads to parallelize with many steps of sub-protocols, such as step 2 and step 5 in the Π_{SCV} protocol, step 2 and step 3 in the Π_{SMEV} protocol, and step 1 in the Π_{STM} protocol, to reduce the running time.

B. Experiment Results

To validate the application of PPDTL protocol in intelligent transportation systems, we performed experiments on the LISA Traffic Light Dataset [51], [52], which consists of continuous test and training video sequences and totaling 43,007 frames and 113,888 annotated traffic lights. We take the sample data in the dayTrain folder of this dataset as the source domain data (training data) and then take the sample data in daySequence1, daySequence2, and daySequence3 as the target domain data (prediction data). In our experiments, the parties first extract image features locally, then initialize the data after feature extraction, and then execute the PPDTL protocol to

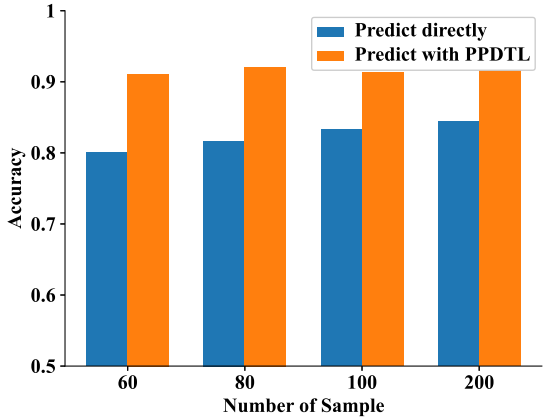


Fig. 12. Performance of PPDTL protocol.

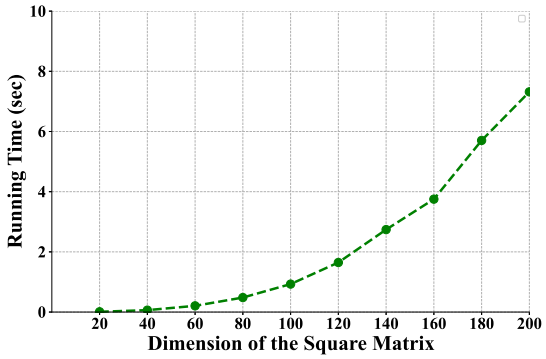


Fig. 13. Performance of Π_{SMM}.

obtain the shares of source domain data and target domain data after transfer learning for both parties. In the next step, both parties publish their shares to reveal the plaintexts of the transferred source domain data and transferred target domain data. Finally, the party that holds the target domain data uses the transferred data to perform KNN classification on the target domain data. (Note that for the sake of privacy preservation, the parties can classify the target domain data using the privacy-preserving classification method [17], [62], but this is beyond the scope of this paper.) By random sampling, we test the effects on classification accuracy by various sample sizes. The experimental results are shown in Figure 12, where the blue color represents the accuracy of KNN classification directly on the target domain data, and the orange color represents the accuracy of KNN classification after using the PPDTL protocol, which shows that the PPDTL protocol improves the accuracy by about 7%-10%. We observe that in this dataset, the greatest improvement in classification accuracy is achieved using PPDTL when the sample size is around 80, which means that excellent results can be achieved without a large amount of labeled data when using our protocol.

Through the experiment on synthetic datasets, we show the performance of Π_{SMM} with different dimensions of the square matrix (from 20 to 200) in Fig. 13. We can see that the time overhead of Π_{SMM} is quadratically related to the dimensions of the square matrix.

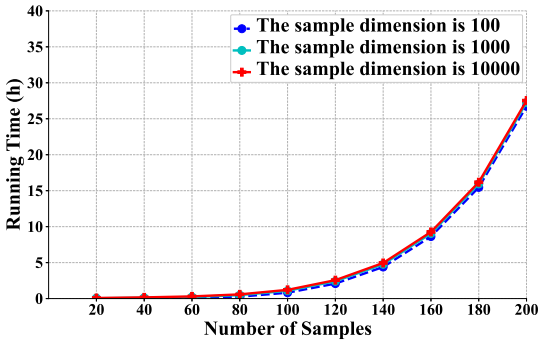


Fig. 14. Performance of Π_{PPDTL}.

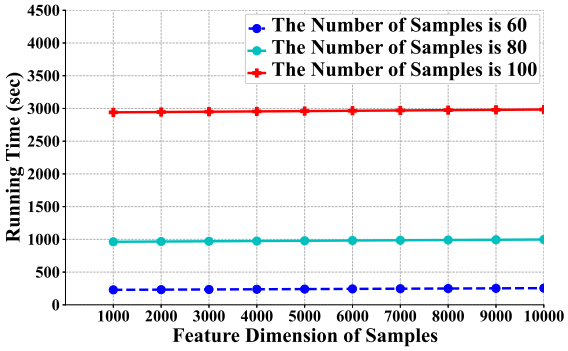


Fig. 15. Influence of performance.

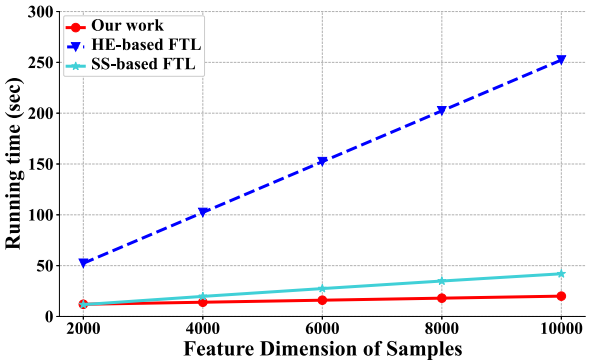
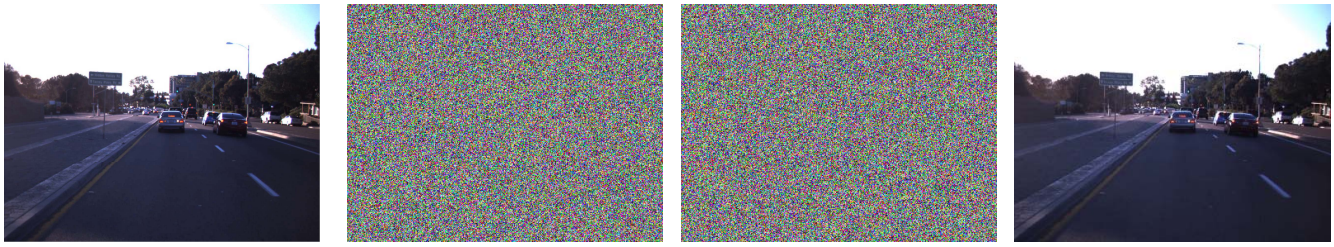


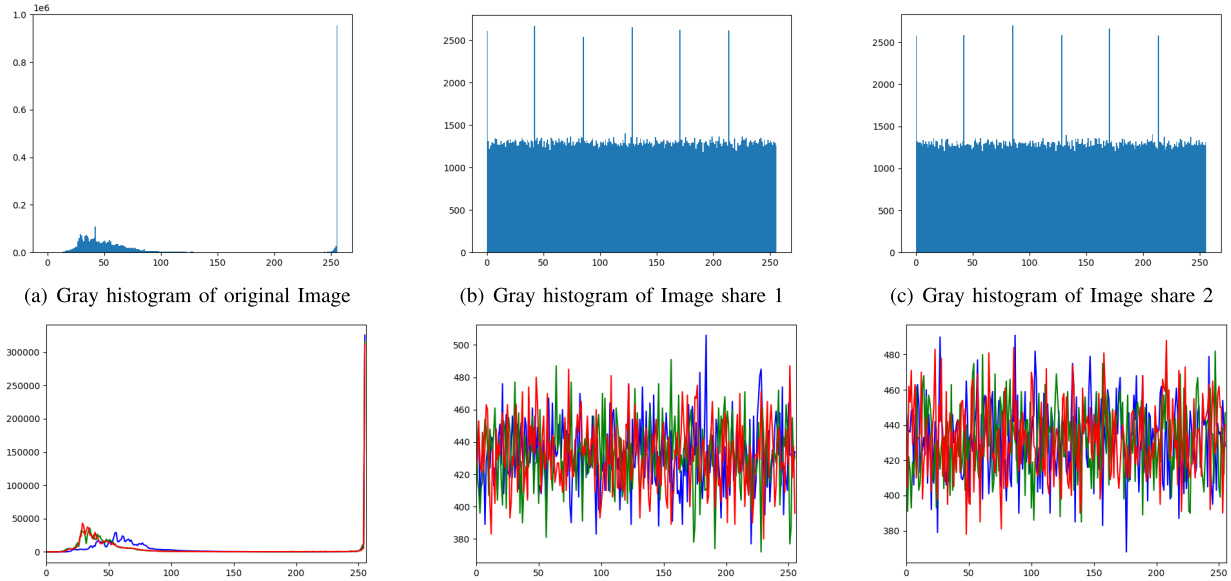
Fig. 16. Comparison with the state-of-the-art work.

In the testing of the PPDTL protocol, we show the performance of PPDTL on synthetic datasets with different sample numbers (from 20 to 200) and three different sample dimensions (100, 1000 and 10000) in Fig. 14. From the experimental results, it can be seen that the time to execute a SMM protocol takes only a few seconds, while the time to execute a PPDTL protocol takes several hours. It is because the times of invoking SMM sub-protocols in PPDTL are quadratically related to the number of samples. Although the running time of our protocol increases rapidly with the number of samples, it can be seen from Fig.12 that the greatest improvement in classification accuracy is achieved using PPDTL when the number of samples is about 80, and the time cost is small with only about 0.27 hours.



(a) Original image (b) Image share 1 (c) Image share 2 (d) Revealed image

Fig. 17. Image secret sharing.



(a) Gray histogram of original Image (b) Gray histogram of Image share 1 (c) Gray histogram of Image share 2

(d) RGB histogram of original Image (e) RGB histogram of Image share 1 (f) RGB histogram of Image share 2

Fig. 18. Image secret sharing.

From figure 14, we can see that the three color lines almost overlap, which indicates that the performance of PPDTL is almost the same in three different sample dimensions. To more clearly show the influence on PPDTL by sample dimension, in Fig. 15 we show the performance of PPDTL with different dimensions numbers (from 1000 to 10000) and three different sample numbers (60, 80, 100). We can see that the gradient of all the lines is almost zero, which indicates that the influence of the dimension numbers on the PPDTL is quite low.

By observing Fig. 14 and Fig. 15, we can conclude that the main factor influencing the performance of PPDTL is the sample numbers, and the influence of the sample dimension can be ignored, which is also easy to explain from a theoretical aspect. Because in our protocol, except the computation when generating the kernel matrix is related to the sample dimension, all other computations are only related to the scale of the kernel matrix, which is only related to the number of samples, so the number of samples directly determines the performance of the protocol. The sample dimension can only influence the computation of the kernel matrix, which is very slight. This conclusion implies that PPDTL is well suited to handle high-dimensional data.

To validate the efficiency of our protocol, we compare it with the state-of-the-art work [20] on synthetic datasets. Since [20] provides a method for training models in privacy-preserving transfer learning scenarios, to achieve the same function as [20], in our experiments, we first use the PPDTL protocol to compute the shares of the migrated source domain data and the target domain data, and then use these shares as input to invoke the privacy-preserving linear regression method of SecureML [17] for training a model. The experimental results are shown in Fig.16. It can be seen that our method outperforms the method in [20] in terms of running time. When the sample dimension is 2000, the running time of our method is roughly 4 times faster than that of HE-based FTL and similar to that of SS-based FTL. When the sample dimension is 10000, our method is roughly 12 times faster than HE-based FTL and roughly 2 times faster than SS-based FTL. In addition, our protocol outperforms [20] in terms of the number of interaction rounds. Our PPDTL protocol can be completed in a constant round while the rounds of interactions in [20] is related to the number of sample data.

Through experimental results and theoretical analysis, we conclude that our proposed protocol outperforms the existing state-of-the-art work in terms of running time and

number of Interactive rounds for traffic data with high sample dimensions.

C. Security Validation

To experimentally valid the security of the PPDTL protocol, an example of a traffic light image from the LISA Traffic Light Dataset [51], [52] is given in Fig.17 In the PPDTL protocol, the original image data are first divided into two shares by secret sharing. It can be observed that the original image becomes two disordered and meaningless share images after secret sharing. Intuitively, these two share images reveal nothing about the original image. To further prove this conclusion, we plotted histograms (Fig.18) of the share image and the original image. It is easy to see that the grayscale values of the original image are concentrated in a specific internal, while the grayscale values of the share image are uniformly distributed. This means that the adversaries fail to obtain any information about the original image through the share image. In addition, as shown in the figure, the original image can be recovered by adding the RGB three-channel values of the two share images separately.

VIII. Conclusion

The achievement of ITS applications relies heavily on a large amount of labeled traffic data. Nevertheless, in real-world scenarios, most intelligent transportation service providers often suffer from insufficient labeled data. Some existing works use transfer learning to solve this problem, however when the labeled dataset and the unlabeled dataset are held by two entities, it is still unrealistic for two mutually distrustful data owners to cooperate in transfer learning regarding data security and privacy-preserving. In this paper, we propose a secure two-party computation protocol for transfer learning, which can perform transfer learning without revealing any private data of both two mutually distrustful entities and is very appropriate for traffic data. With this protocol, we extend the application scenario of transfer learning to allow the transfer learning can be done in the scenario where the source domain data and the target domain data belong to two different data owners. Compared to existing work, our work addresses the privacy-preserving problem of transfer learning with traffic data. In addition, our protocol satisfies stronger security and has fewer interaction rounds than other privacy-preserving transfer learning protocols.

Through experiments, we find that the dimension of the sample data has little effect on the runtime of our protocol while the increase in the number of samples has a large impact, which indicates that our method is more appropriate for traffic data with large data dimension than for datasets with very large sample size. Fortunately, in our experiments, we find that the protocol achieves excellent results without a large amount of data. Furthermore, through comparison experiments, we observe that our protocol is roughly 2 to 12 times faster than the existing state-of-the-art work on datasets with high sample dimensions.

Moreover, our protocol is well extensible because the output form of our protocol is an arithmetic share, so our protocol can be used in combination with any privacy-preserving machine learning scheme based on secret sharing.

At this point, there is still an open research issue which is how to securely perform transfer learning when source domain data and participant data are held by more than two mutually distrustful entities. In the future, we will continue to research the privacy-preserving transfer learning protocols in the multi-party setting to support multiple source domain data and one target domain scenarios, which will enable the Π_{PPDTL} protocols to apply to more ITS scenarios.

References

[1] M. Usman, M. A. Jan, and A. Jolfaei, "SPEED: A deep learning assisted privacy-preserved framework for intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4376–4384, Jul. 2021.

[2] K. Lou, Y. Yang, E. Wang, Z. Liu, T. Baker, and A. K. Bashir, "Reinforcement learning based advertising strategy using crowdsensing vehicular data," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4635–4647, Jul. 2020.

[3] H. Wang, Y. Yu, Y. Cai, L. Chen, and X. Chen, "A vehicle recognition algorithm based on deep transfer learning with a multiple feature subspace distribution," *Sensors*, vol. 18, no. 12, p. 4109, Nov. 2018.

[4] J. Blitzer, R. McDonald, and F. Pereira, "Domain adaptation with structural correspondence learning," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2006, pp. 120–128.

[5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Dec. 2009.

[6] E. L. Manibardo, I. Lana, and J. D. Ser, "Transfer learning and online learning for traffic forecasting under different data availability conditions: Alternatives and pitfalls," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6.

[7] W. Zhao, "Research on the transfer learning of the vehicle logo recognition," in *Proc. AIP Conf. Proc.*, vol. 1864, 2017, Art. no. 020058.

[8] M. Elhenawy, H. I. Ashqar, M. Masoud, M. H. Almannaa, A. Rakotonirainy, and H. A. Rakha, "Deep transfer learning for vulnerable road users detection using smartphone sensors data," *Remote Sens.*, vol. 12, no. 21, p. 3508, Oct. 2020.

[9] X. Guo, Q. Yao, W. Tu, Y. Chen, W. Dai, and Q. Yang, "Privacy-preserving transfer learning for knowledge sharing," 2018, *arXiv:1811.09491*.

[10] C. Dwork, "Differential privacy: A survey of results," in *Proc. Int. Conf. Theory Appl. Models Comput.* Berlin, Germany: Springer, 2008, pp. 1–19.

[11] D. Gao, Y. Liu, A. Huang, C. Ju, H. Yu, and Q. Yang, "Privacy-preserving heterogeneous federated transfer learning," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 2552–2559.

[12] R. L. Rivest, L. Adleman, and M. L. Dertouzos, "On data banks and privacy homomorphisms," *Found. Secure Comput.*, vol. 4, no. 11, pp. 169–180, 1978.

[13] C. Jin, M. Ragab, and K. M. M. Aung, "Secure transfer learning for machine fault diagnosis under different operating conditions," in *Proc. Int. Conf. Provable Secur.* Cham, Switzerland: Springer, 2020, pp. 278–297.

[14] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Symp. Theory Comput. (STOC)*, 2009, pp. 169–178.

[15] B. Zhang, C. Chen, and L. Wang, "Privacy-preserving transfer learning via secure maximum mean discrepancy," 2020, *arXiv:2009.11680*.

[16] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2010.

[17] P. Mohassel and Y. Zhang, "SecureML: A system for scalable privacy-preserving machine learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 19–38.

[18] N. Agrawal, A. S. Shamsabadi, M. J. Kusner, and A. Gascón, "QUOTIENT: Two-party secure neural network training and prediction," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 1231–1247.

[19] P. Krishnakumari, A. Perotti, V. Pinto, O. Cats, and H. Van Lint, "Understanding network traffic states using transfer learning," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 1396–1401.

[20] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, "A secure federated transfer learning framework," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 70–82, Jul./Aug. 2020.

[21] F. Zhuang et al., "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jul. 2020.

[22] M. Sugiyama, T. Suzuki, S. Nakajima, H. Kashima, P. Buenau, and M. Kawanabe, "Direct importance estimation for covariate shift adaptation," *Ann. Inst. Statist. Math.*, vol. 60, no. 4, pp. 699–746, 2008.

[23] F. Zhuang, P. Luo, H. Xiong, Q. He, Y. Xiong, and Z. Shi, "Exploiting associations between word clusters and document classes for cross-domain text categorization," *Stat. Anal. Data Mining, ASA Data Sci. J.*, vol. 4, no. 1, pp. 100–114, 2011.

[24] J. Ghosn and Y. Bengio, "Bias learning, knowledge sharing," *IEEE Trans. Neural Netw.*, vol. 14, no. 4, pp. 748–765, Jul. 2003.

[25] L. Song et al., "Colored maximum variance unfolding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1385–1392.

[26] P. Von Bünau, F. C. Meinecke, F. C. Király, and K.-R. Müller, "Finding stationary subspaces in multivariate time series," *Phys. Rev. Lett.*, vol. 103, no. 21, Nov. 2009, Art. no. 214101.

[27] S. J. Pan, J. T. Kwok, and Q. Yang, "Transfer learning via dimensionality reduction," in *Proc. AAAI*, vol. 8, 2008, pp. 677–682.

[28] J. A. K. Suykens, "Data visualization and dimensionality reduction using kernel maps with a reference point," *IEEE Trans. Neural Netw.*, vol. 19, no. 9, pp. 1501–1517, Sep. 2008.

[29] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2200–2207.

[30] Z. Ma et al., "VerifyTL: Secure and verifiable collaborative transfer learning," 2020, *arXiv:2005.08997*.

[31] L. Qi, "Research on intelligent transportation system technologies and applications," in *Proc. Workshop Power Electron. Intell. Transp. Syst.*, Aug. 2008, pp. 529–531.

[32] S.-H. An, B.-H. Lee, and D.-R. Shin, "A survey of intelligent transportation systems," in *Proc. 3rd Int. Conf. Comput. Intell., Commun. Syst. Netw.*, Jul. 2011, pp. 332–337.

[33] H. Sun, H. X. Liu, H. Xiao, R. R. He, and B. Ran, "Use of local linear regression model for short-term traffic forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1836, no. 1, pp. 143–150, Jan. 2003.

[34] Z. Shan, D. Zhao, and Y. Xia, "Urban road traffic speed estimation for missing probe vehicle data based on multiple linear regression model," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 118–123.

[35] N. Zenina and A. Borisov, "Regression analysis for transport trip generation evaluation," *Inf. Technol. Manage. Sci.*, vol. 16, no. 1, pp. 89–94, 2013.

[36] H. J. Payne and S. C. Tignor, "Freeway incident-detection algorithms based on decision trees with states," *Transp. Res. Rec.*, no. 682, pp. 30–37, Jan. 1978.

[37] J. Abellán, G. López, and J. De Oña, "Analysis of traffic accident severity using decision rules via decision trees," *Exp. Syst. Appl.*, vol. 40, no. 15, pp. 6047–6054, 2013.

[38] C. Xie, J. Lu, and E. Parkany, "Work travel mode choice modeling with data mining: Decision trees and neural networks," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1854, no. 1, pp. 50–61, Jan. 2003.

[39] L. Vanajakshi and L. R. Rilett, "Support vector machine technique for the short term prediction of travel time," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2007, pp. 600–605.

[40] Y. Bin, Y. Zhongzhen, and Y. Baozhen, "Bus arrival time prediction using support vector machines," *J. Intell. Transp. Syst.*, vol. 10, no. 4, pp. 151–158, 2006.

[41] J. Xiao and Y. Liu, "Traffic incident detection using multiple-kernel support vector machine," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2324, no. 1, pp. 44–52, Jan. 2012.

[42] Y. Meng and X. Liu, "Application of K-means algorithm based on ant clustering algorithm in macroscopic planning of highway transportation hub," in *Proc. 1st IEEE Int. Symp. Inf. Technol. Appl. Educ.*, Nov. 2007, pp. 483–488.

[43] R. P. Deb Nath, H.-J. Lee, N. K. Chowdhury, and J.-W. Chang, "Modified K-means clustering for travel time prediction based on historical traffic data," in *Proc. Int. Conf. Knowl.-Based Intell. Inf. Eng. Syst.* Berlin, Germany: Springer, 2010, pp. 511–521.

[44] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[45] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2014.

[46] K. Zhang, H. D. Cheng, and B. Zhang, "Unified approach to pavement crack and sealed crack detection using preclassification based on transfer learning," *J. Comput. Civil Eng.*, vol. 32, no. 2, Mar. 2018, Art. no. 04018001.

[47] S. Bang, S. Park, H. Kim, and H. Kim, "Encoder–decoder network for pixel-level road crack detection in black-box images," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 34, no. 8, pp. 713–727, Aug. 2019.

[48] C. N. H. Vinh, A. Truong, and T. T. Huu, "A privacy preserving authentication scheme in the intelligent transportation systems," in *Proc. Int. Conf. Future Data Secur. Eng.* Cham, Switzerland: Springer, 2018, pp. 103–123.

[49] X. Liu and Y. Zhu, "Privacy and utility preserving trajectory data publishing for intelligent transportation systems," *IEEE Access*, vol. 8, pp. 176454–176466, 2020.

[50] C. Gosman, C. Dobre, and F. Pop, "Privacy-preserving data aggregation in intelligent transportation systems," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, May 2017, pp. 1059–1064.

[51] M. P. Philipsen, M. B. Jensen, A. Mogelmose, T. B. Moeslund, and M. M. Trivedi, "Traffic light detection: A learning algorithm and evaluations on challenging dataset," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, Sep. 2015, pp. 2341–2345.

[52] M. B. Jensen, M. P. Philipsen, A. Møgelmose, T. B. Moeslund, and M. M. Trivedi, "Vision for looking at traffic lights: Issues, survey, and perspectives," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1800–1815, Jul. 2016.

[53] C. Hazay and Y. Lindell, *Efficient Secure Two-Party Protocols: Techniques and Constructions*. Berlin, Germany: Springer, 2010.

[54] A. C.-C. Yao, "How to generate and exchange secrets," in *Proc. 27th Annu. Symp. Found. Comput. Sci. (SFCS)*, Oct. 1986, pp. 162–167.

[55] D. Beaver, "Efficient multiparty protocols using circuit randomization," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 1991, pp. 420–432.

[56] D. Demmler, T. Schneider, and M. Zohner, "ABY—A framework for efficient mixed-protocol secure two-party computation," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15.

[57] R. Cramer and I. Damgård, "Secure distributed linear algebra in a constant number of rounds," in *Proc. Annu. Int. Cryptol. Conf.* Berlin, Germany: Springer, 2001, pp. 119–136.

[58] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, 2017.

[59] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial–temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*.

[60] H. Yao et al., "Deep multi-view spatial–temporal network for taxi demand prediction," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, pp. 1–8, 2018.

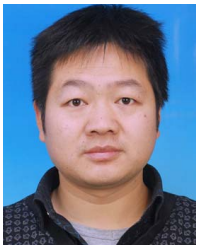
[61] H. Yao, X. Tang, H. Wei, G. Zheng, and Z. Li, "Revisiting spatial–temporal similarity: A deep learning framework for traffic prediction," in *Proc. Conf. Artif. Intell.*, vol. 33, 2019, pp. 5668–5675.

[62] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via MiniONN transformations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur., (CCS)*, B. M. Thuraisingham, D. Evans, T. Malkin, and D. Xu, Eds. Dallas, TX, USA: ACM, Oct./Nov. 2017, pp. 619–631, doi: 10.1145/3133956.3134056.

[63] M. Keller, "MP-SPDZ: A versatile framework for multi-party computation," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*. New York, NY, USA: ACM, Oct. 2020, pp. 1575–1590.

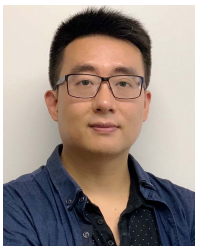
Zhi Li received the B.S. degree in computer science from Jining Medical University in 2018. He is currently pursuing the Ph.D. degree with Shandong Normal University. His research interests include secure multi-party computation and privacy-preserving computation.

Hao Wang (Member, IEEE) received the Ph.D. degree in computer science from Shandong University in 2012. He was a Visiting Fellow at the University of Wollongong from 2019 to 2020. He is currently a Full Professor and a Ph.D. Supervisor at Shandong Normal University. He has coauthored over 60 research papers in prestigious journals and conferences. His current research interests include applied cryptography, secure multi-party computation, and blockchain.



Guangquan Xu (Member, IEEE) received the Ph.D. degree from Tianjin University in March 2008. He is currently a Full Professor at the Tianjin Key Laboratory of Advanced Networking (TANK), College of Intelligence and Computing, Tianjin University, China. He is a fellow of IET and a member of the CCF. He is also the Director of the Network Security Joint Laboratory and the Network Attack & Defense Joint Laboratory. He has published more than 100 papers in reputable international journals and conferences, including IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE INTERNET OF THINGS JOURNAL, *FGCS*, *IEEE Communications Magazine*, *Information Sciences*, IEEE WIRELESS COMMUNICATIONS, IEEE TRANSACTIONS ON CYBERNETICS, *ACM TIST*, and *IEEE Network*. His research interests include cybersecurity and trust management. He served as a TPC Member for FCS2020, IEEE UIC 2018, SPNCE2019, IEEE UIC2015, and IEEE ICECCS 2014, and a Reviewer for IEEE ACCESS, *ACM TIST*, *JPDC*, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *Soft Computing*, *FGCS*, and *Computational Intelligence*.

Alireza Jolfaei (Senior Member, IEEE) is currently an Associate Professor in cybersecurity and networking with the College of Science and Engineering, Flinders University. He is also a Distinguished Speaker of the ACM on the topic of cybersecurity. His main research interests are cyber-physical systems security, where he investigates the hidden interdependencies in industrial communication protocols and aims to provide fundamentally new methods for security-aware modeling, analysis, and design of safety-critical cyber-physical systems in the presence of cyber-adversaries. He has been a chief investigator of several internal and external grants with a total amount exceeding $2,6 million. He successfully supervised eight HDR students to completion. He received the prestigious IEEE Australian Council Award for his research paper published in the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY. He is a Board Member of the IEEE Consumer Technology Society's publications and the Editor-in-Chief of the IEEE Consumer Technology Society's World Newsletter. He has served as the IEEE Australia Chair of the IEEE Technology and Engineering Management Society for Membership Development and Activities, the Secretary of the IEEE NSW Joint Chapter on Consumer Technology, Broadcast Technology, and Product Safety Engineering, the Chairman of the Computational Intelligence Society in the IEEE Victoria Section, and the Chairperson of Professional and Career Activities for the IEEE Queensland Section. He was the Founder and the Councillor of the IEEE Student Branch at Federation University and also contributed to the foundation of the IEEE Northern Territory Subsection. He has served as the Associate Editor of several IEEE journals and IEEE TRANSACTIONS, including the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and the IEEE TRANSACTIONS ON CONSUMER TECHNOLOGY. He has served as a Program Co-Chair, a Track Chair, and a Technical Program Committee Member, for major conferences, including IEEE TrustCom and IEEE ICCCN.



Xi Zheng (Member, IEEE) received the Ph.D. degree in software engineering from UT Austin in 2015. From 2005 to 2012, he was the Chief Solution Architect at Menulog, Australia. He is currently the Director of the Intelligent Systems Research Group (ITSEG.ORG), a Senior Lecturer (an aka Associate Professor USA), and the Deputy Program Leader of Software Engineering at Macquarie University, Australia. His research interests include the Internet of Things, intelligent software engineering, machine learning security, human-in-the-loop AI, and edge intelligence. He has secured more than $1 million competitive funding in Australian Research Council (Linkage and Discovery) and Data61 (CRP) projects on safety analysis, model testing and verification, and trustworthy AI on autonomous vehicles. He also won a few awards, including Deakin Industry Researcher (2016) and MQ Earlier Career Researcher (Runner-up 2020). He has a number of highly cited papers and best conference papers. He serves as a PC Member for CORE A* Conferences, including FSE (2022) and PerCom (2017–2022). He also serves as the PC Chair of IEEE CPSCom-2021, IEEE Broadnets-2022, and an Associate Editor for *Distributed Ledger Technologies*.

Chunhua Su received the B.S. degree from the Beijing Electronic and Science Institute in 2003 and the M.S. and Ph.D. degrees in computer science from the Faculty of Engineering, Kyushu University, in 2006 and 2009, respectively. He is currently working as a Senior Associate Professor with the Division of Computer Science, The University of Aizu. He has worked as a Post-Doctoral Fellow at Singapore Management University from 2009 to 2011 and a Research Scientist with the Cryptography & Security Department, Institute for Infocomm Research, Singapore, from 2011 to 2013. From 2013 to 2016, he has worked as an Assistant Professor with the School of Information Science, Japan Advanced Institute of Science and Technology. From 2016 to 2017, he worked as an Assistant Professor with the Graduate School of Engineering, Osaka University. He has published more than 100 papers in international journals and conferences. His research interests include cryptanalysis, cryptographic protocols, privacy-preserving technologies in machine learning, and the IoT security & privacy.

Wenying Zhang received the Ph.D. degree in cryptography from the Department of Information Research, Information Engineering University of PLA, Zhengzhou, Henan, China, in June 2004. From July 2004 to September 2006, she was a Post-Doctoral Fellow at the Institute of Software, Chinese Academy of Sciences, Beijing, China. She is currently a Professor and a Ph.D. Supervisor at Shandong Normal University. Her research interests include cryptography, Boolean function, and hash function analysis.