Network Traffic Prediction Based on LSTM and Transfer Learning

XIANBIN WAN, HUI LIU, HAO XU, AND XINCHANG ZHANG, (Senior Member, IEEE)

1Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China
2College of Computer Science and Technology, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250306, China

Corresponding author: Xianbin Wan (15069118031@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 92067108, in part by the Shandong Provincial Natural Science Foundation of China under Grant ZR2020MF057, and in part by the Basic Research Promotion Plan of Qilu University of Technology (Shandong Academy of Sciences) under Grant 2021JC03001.

ABSTRACT The increasing amount of traffic in recent years has led to increasingly complex network problems. To be able to improve overall network performance and increase network utilization, it is valuable to take measures to capture future trends in network traffic. In traditional machine learning, to guarantee the accuracy and high reliability of the models obtained through training, there are two basic assumptions: (1) the training samples used for learning and the new test samples satisfy the condition of independent identical distribution; and (2) there must be enough training samples to learn a good model. However, time-series data are not easily accessible in real life, and even after putting in a lot of time and effort to collect them, the data may be unavailable due to confidentiality. In this paper, a neural network model based on long and short-term memory (LSTM) and transfer learning is proposed to address the problem of small sample size in network traffic prediction. Knowledge in the source domain is transferred to the target domain using transfer learning, and a prediction model with good performance is constructed with a small amount of target domain data. The results show that the performance of the transfer learning model improves by more than 40% over the direct training model when using the same samples for predicting 10,000 rows of data, resulting in better performance of the network traffic prediction task.

INDEX TERMS LSTM, network traffic prediction, transfer learning.

I. INTRODUCTION

Due to the rapid development of society, the network is getting more and more traffic. According to the latest Visual Networking Index (VNI) report [1] by Cisco, in 2022, more traffic will flow through the global network than in all 32 years combined from the first year of the Internet to the end of 2016. Global traffic will more than triple, and by 2022, traffic flowing through the worldwide network will reach 4.8 terabytes (ZB) per year or 396 exabytes per month. In 2017, the annual run rate of global traffic was 1.5 ZB per year or 122 exabytes per month. The increase in network traffic makes the network situation more and more complex. Therefore, a large number of solutions (e.g., [2], [3]) have been proposed to optimize the network traffic.

Analyzing traffic data can improve network quality, enhance network security, and prevent congestion. Future traffic data can be obtained through network traffic prediction. It plays an essential role in network management, network design, short and long-term resource allocation, traffic (rerouting), anomaly detection, and other network areas. Accurate traffic prediction can smooth out delay-sensitive traffic, perform dynamic allocation of bandwidth services, achieve congestion control on the network, and enhance the overall user experience. To be able to improve overall network performance and enhance network utilization, it is valuable to take steps to capture the future trend of network traffic.

In traffic prediction, scholars at home and abroad have conducted extensive research for a long time and put forward many effective methods. The main models include multivariate linear AR model based on time points,
segmented autoregressive sliding average (ARMA) model [4], differential autoregressive moving average (ARIMA) model [5], differential autoregressive summation sliding average (FARIMA) model [6], etc. In addition, some scholars apply nonlinear theory to network traffic prediction and propose prediction models based on support vector machines (SVM) [7], gray models (GM) [8], Gaussian processes (GP) [9], and neural networks (NN) [10]. For example, the gray model is based on support vector machine compensation, the Gaussian process hybrid prediction model is based on Gaussian distribution and the traffic prediction model is based on long and short-term memory (LSTM) neural network.

Although the prediction effects of the above models are more satisfactory, there are still shortcomings. With the increase in network complexity, the distribution characteristics of network traffic have exceeded the traditional sense of Poisson distribution or Markov distribution, so it is difficult to ensure the accuracy of the linear model prediction. Increasingly mature machine learning-based traffic prediction methods have received great attention, and many traffic prediction models based on vector machines and artificial neural networks have emerged to greatly improve the prediction of complex traffic at present. For traditional machine learning models based on vector machines and artificial neural networks, to guarantee the accuracy and high reliability of the models obtained by training, there are two basic assumptions: (1) the training samples used for learning and the new test samples satisfy the condition of independent identical distribution; (2) there must be enough training samples to learn a good model. However, in practical applications, these two conditions are often not satisfied. Many fields eager to use machine learning do not have enough data to train a model. In this context, transfer learning was born. Transfer Learning is a term used in machine learning to refer to the effect of one type of learning on another type of learning, or the effect of an acquired experience on the completion of other activities. It can transfer existing knowledge to solve the problem of having only a small amount of labeled data in the target region [11].

In this paper, we propose a network traffic prediction method based on LSTM neural network and transfer learning. The method uses the idea of transfer learning to save the knowledge acquired during the execution of the source task in the source domain. When the knowledge in the target domain is insufficient to complete the target task, the saved knowledge is applied to complete the target task. The specific implementation is to transfer the parameters in the network traffic prediction model with sufficient source domain data training to the network traffic prediction model without sufficient target domain data training and then train with less target domain data, and finally get the network traffic prediction model with more accurate prediction.

The network prediction task involved in this paper is a single indicator time series prediction task, i.e., given the historical change of a certain indicator, predict its change in the future period. The acquisition of traffic sequence data requires a lot of time and effort, and even then, the acquired traffic sequence data may not be usable due to the inclusion of privacy. In traditional network traffic prediction methods, neural network models can use network layers to extract features from sufficient data and show good performance in performing prediction tasks. However, when there is insufficient data, the neural network model is unable to create attributes that are not present in the data. If the neural network model obtains training data that is not representative, it models the unique attributes in these training data as general attributes, which is often referred to as the overfitting problem. The overfitting problem will result in a neural network model that predicts the training data more accurately but has a higher error rate for other data and poor generalization performance. The method proposed in this paper uses the idea of transfer learning to transfer the parameters of a network traffic prediction model, which has been trained in other domains, to the original LSTM model. The constructed LSTM model is trained using the pre-processed target domain data. The method proposed in this paper results in more accurate network traffic prediction and better generalization of the network model when using the same size of data.

Considering the previous studies, the key contributions of this work can be summarized as follows:

1. Building a network traffic prediction architecture based on LSTM and transfer learning.
2. By adding transfer learning, a neural network model can be trained using a small amount of data. Our method is able to produce more accurate predictions than the method without transfer learning.
3. Transfer learning has been applied to solve classification problems, usually in combination with CNN neural networks. The combination of transfer learning and LSTM proposed in this paper extends the application area of transfer learning and, at the same time, proposes a new method to solve the prediction problem.

The paper is organized as follows. Section II briefly summarises LSTM and transfer learning and why transfer learning should be used in network traffic prediction. Section III describes the network traffic prediction architecture based on LSTM and transfer learning that will be used in this paper. Section IV presents performance results from specific test scenarios, and conclusions are presented in Section V.

II. RELATED WORK

As mentioned in the previous section, the use of linear and nonlinear models for network traffic prediction has been extensively studied in the literature, mainly by constructing fine-grained neural network models and then training them using sufficient amounts of data. In contrast to the above literature, we will use a small amount of data to construct well-performing network traffic prediction models that address the problem of data not being easily available in the network domain. In this section, we present the background of
network traffic prediction research based on LSTM networks, and why we use LSTM networks and transfer learning for network traffic prediction research.

A. LSTM

Long short-term memory (LSTM) is a modified recurrent neural network that is suitable for processing and predicting important events with very long intervals and delays in time series. The LSTM network contains LSTM blocks, which may be described as intelligent network units because they can remember values of indefinite duration, and there are “gates” in the blocks that can determine whether the “input” needs to be remembered and whether it can be output to the “output”. The structure is shown in Fig. 1.

The hidden layer of the recurrent neural network has only one state, “h”, which is very sensitive to short-term input. For the LSTM network, three “gates” are set: the forget gate, which is responsible for controlling the continued preservation of the long-term state “c”; the input gate, which is responsible for controlling the input of the immediate state to the long-term state “c”; and the output gate, which is responsible for controlling whether to use the long-term state “c” as the output of the current LSTM.

Network traffic prediction is based on data prediction, and models that perform well in the field of data prediction will also be applicable in the field of network traffic. To guarantee high-accuracy vessel trajectory prediction, [12] proposes an AIS data-driven trajectory prediction framework, whose main component is a long short-term memory network. The vessel traffic conflict situation modeling, generated using the dynamic AIS data and social force concept, is embedded into the LSTM network. [13] proposed a spatio-temporal multi-graph convolutional network (STMGCN) based vessel trajectory prediction framework using the mobile edge computing (MEC) paradigm. It is mainly composed of three different graphs, which are, respectively, reconstructed according to the social force, the time to the closest point of approach (TCPA), and the size of surrounding vessels. These three graphs are then jointly embedded into the prediction framework by introducing the spatio-temporal multi-graph convolutional layer (STMGCL).

In general, network traffic prediction is a well-researched area, and LSTM-based prediction models have a wide range of applications. In [14], the authors designed an LSTM neural network-based traffic prediction system using mobile services at LTE base stations as the research object. Ramakrishnan and Soni [15] proposed several recurrent neural network (RNN) structures (standard RNN, long short-term memory (LSTM) network, and gate recurrent unit (GRU)) to solve the network traffic prediction problem. The performance of these models was analyzed for three important problems in network traffic prediction: traffic prediction, packet protocol prediction, and packet distribution prediction. Recent results were obtained on traffic prediction problems on public datasets such as GEANT and Abilene networks. To enhance the robustness of the real-time network traffic prediction model [16], Lu and Yang modified the loss function of the LSTM network. Unlike the traditional LSTM model, the model was continuously updated with the arrival of new traffic. The experimental results showed that the model has better prediction accuracy compared with models constructed by support vector regression and bp neural networks.

It is clear from the literature that LSTM neural networks are widely used for network traffic prediction. However, the performance of LSTM neural network models is also limited by the amount of training data, and they cannot perform the prediction task well when the training data is too small. Therefore, transfer learning, which requires only a small amount of data, is crucial for network traffic prediction.

B. TRANSFER LEARNING

Weiss et al. [17] had given a unified definition of transfer learning.

Definition (Transfer learning): Given a source domain $D_S$ and learning task $T_S$, a target domain $D_T$ and learning task $T_T$, transfer learning aims to help improve the learning of the target predictive function $f_T(x)$ in $D_T$ using the knowledge in $D_S$ and $T_S$, where $D_S \neq D_T$, or $T_S \neq T_T$.

Transfer learning can be divided into the following three categories.

- **Inductive transfer learning**: Whether the source domain is the same as the target domain or not, the source task is different from the target task.
- **Transductive transfer learning**: The source task is the same as the target task, but the source domain target domain is different.
- **Unsupervised transfer learning**: The source task is relevant to the target task regardless of whether the source and target domains are the same.

Pan and Yang [18] had given a unified definition of transductive transfer learning.
Definition for Transductive Transfer Learning: Given a source domain $D_S$ and a corresponding learning task $T_S$, a target domain $D_T$ and a corresponding learning task $T_T$, transductive transfer learning aims to improve the learning of the target predictive function $f_T(\cdot)$ in $D_T$ using the knowledge in $D_S$ and $T_S$, where $D_S \neq D_T$ and $T_S \neq T_T$. In addition, some unlabeled target domain data must be available at training time.

The difference between traditional learning and transfer learning is shown in Fig. 2. In the study of this paper, network traffic prediction in different network environments, i.e., $D_S \neq D_T$, network traffic has the same data dimension and the same feature space, i.e., $T_S = T_T$. From the definitions, we can see that transductive transfer learning is the best choice where the source domain is different from the target domain, but the source task is the same as the target task.

Transfer learning of neural networks first trains a base network on a source dataset and then transfers the learned features (weights of the network) to a second network, trained on the target dataset. This idea has been shown to improve the generalization ability of deep neural networks in many computer vision tasks, such as image recognition and object localization. However, unlike the image recognition problem, transfer learning techniques have not been thoroughly investigated in time series classification tasks. Based on this, Fawaz et al. in [19] construct deep convolutional neural networks to solve the time series classification problem.

In [20], Kashiparekh et al. proposed a deep convolutional neural network trained on a different univariate time series classification task. Once trained, the model can be easily adapted to the new time series classification target task by performing a small amount of fine-tuning using labeled instances of the target task. The authors observe a significant improvement in classification accuracy and computational efficiency when using a pre-trained deep convolutional neural network as a starting point for subsequent task-specific fine-tuning compared to existing state-of-the-art time series classification methods. [21] investigated whether the application of transfer learning to the electroencephalogram time series classification problem could conveniently replace the feature engineering involved with direct data visualization. The model achieved more than 80% classification accuracy, but the trained neural network exhibited overfitting characteristics. The authors suggest that alternative data visualization techniques and modifications of transfer learning methods may yield better results for multichannel neural time series data.

The above literature describes how to use transfer learning in time series data to transfer knowledge from one domain (i.e., source domain) to another domain (i.e., target domain) so that the target domain can achieve better learning results. Usually, the source domain has sufficient data volume and the target domain has less data volume, and transfer learning needs to take the knowledge learned in the case of sufficient data volume and transfer it to the new environment with small data volume.

The transfer is widespread in the learning of various knowledge, skills, and social norms. Transfer learning focuses on storing a solution model for an existing problem and using it for other different but related problems. The literature on network traffic prediction based on small samples is very sparse, mainly due to the difficulty of obtaining data. We aim to address this shortcoming by tackling this prediction problem with transfer learning and LSTM neural networks.

III. NETWORK TRAFFIC PREDICTION ARCHITECTURE BASED ON LSTM AND TRANSFER LEARNING

In this section, a network traffic prediction architecture based on LSTM and transfer learning is built and displayed in Fig. 3. The architecture is divided into a data processing module, a model building module, and a parameter transfer module. The data processing module processes data into time series data more suitable for neural network models to capture features, which includes processing outliers, complementing missing values, scaling data and raw time series to build supervised data. The model building module will construct the LSTM neural network model and use the processed data for training. The parameter transfer module transfers the parameters of the neural network model performing the source task to the neural network model performing the target task. The following sections describe how the data is preprocessed, how the model is built, and how the parameters are transferred.

A. DATA PREPROCESSING

1) PROCESS OUTLIER

An outlier is an observation that deviates too much from other observations, is far from the general level of the series, may be generated by a different series, and is often a very large or very small value. Due to the complex network environment of the industrial Internet, outliers may be generated due to errors in the data acquisition process or may be caused by unreliable network equipment itself and unreliable network transmission. In the general data collection process, outliers appear more frequently, often making it difficult to build the data model later. Therefore, outliers in the data set need to be processed to identify and remove the outliers or use other
values to replace the outliers, obtain a stable data set, and better construct the data model.

The logic of the percentile algorithm is to sort the factor values in ascending or descending order and to process the factor values whose ranking percentile is higher than the set percentage or lower than the set percentage, similar to the practice of “removing the highest scores and the lowest scores” in some competitions. The set percentages need to be analyzed on a case-by-case basis. Due to the uncertainty of the percentages, this paper decided to use the median absolute deviation algorithm for the outliers.

The median absolute deviation (MAD) algorithm is to determine whether each element is an outlier by determining whether its deviation from the median value is within a reasonable range.

1. Calculate the median value of all elements: $X_{\text{median}}$.
2. Calculate the absolute deviation of all elements from the median, a single element is denoted as $X_i$: $\text{bias} = |X_i - X_{\text{median}}|$.
3. Obtain the median value of the absolute deviation: $\text{MAD} = \text{bias}_{\text{median}}$.
4. Determine the parameter $n$, then all the data can be adjusted as (1).

$$X'_i = \begin{cases} X_{\text{median}} + n\text{MAD} & X_i > X_{\text{median}} + n\text{MAD}; \\ X_{\text{median}} - n\text{MAD} & X_i < X_{\text{median}} - n\text{MAD}; \\ X_i & X_{\text{median}} - n\text{MAD} < X_i < X_{\text{median}} + n\text{MAD}. \end{cases}$$

2) COMPLEMENTARY MISSING VALUES

There are many reasons for missing values. Broadly speaking, information is temporarily unavailable; data is not recorded, omitted, or lost due to human factors, which is the main reason for missing data; data is lost due to the failure of data collection equipment, storage media, or transmission media failure; the cost of acquiring such information is too high; the real-time performance of the system requires a high level of performance, that is, it is required to make judgments or decisions quickly before getting such information. The presence of missing values will cause the system to lose a large amount of useful information, making the certainty exhibited in the system weaker and the uncertainty component present in the system more prominent. Data containing null values will cause the data analysis process to fall into chaos and lead to unreliable outputs.

To avoid the problems caused by missing values, the missing values are often removed to obtain the complete data set. Alternatively, other approaches are used for completing, such as Mean/Mode Completer and K-means clustering. The Mean/Mode Completer method divides the attributes in the initial dataset into numerical and non-numerical attributes to be processed separately. If the null value is numeric, the missing attribute is filled based on the average of the values of the attribute in all other objects; if the null value is non-numeric, the missing attribute is filled with the value that has the highest number of values in all other objects (i.e., the value that occurs most frequently) based on the statistical principle of plurality.

Another method similar to it is the Conditional Mean Completer method. In this method, the value used for averaging is not taken from all the objects in the data set, but from those that have the same decision attribute value as that object. The basic starting point of these two methods of data averaging is the same, to supplement the missing attribute values with the maximum probability possible to take the values, only differing a little in the specific method. Compared with the other methods, it uses the majority of the information from the existing data to infer the missing values. The dataset used in this paper is a network traffic dataset, so we use a more
implementable approach. k nearest distance method is to first determine the K nearest samples with missing data based on Euclidean distance or correlation analysis and then weight the average of these K values to estimate the missing data for that sample. In this method, k “neighbors” are first selected based on some distance measure, and their average values are used to interpolate the missing data. The distance measures vary depending on the type of data: 1. Continuous data: The most commonly used distance measures are Euclidean distance, Manhattan distance, and cosine distance. 2. Categorical data: Hamming distance is more commonly used in this case. For all values of categorical attributes, if the values of two data points are different, the distance between them is added by one. The Hamming distance is actually the same as the number of different values taken between attributes.

3) DATA SCALING
Data scaling, in statistics, means that the original data are transformed by a certain mathematical transformation in a certain way to put the data into a small specific interval, such as 0 to 1 or $-1$ to 1. The purpose is to eliminate the differences in characteristics, order of magnitude, and other characteristic attributes between different samples and transform them into a dimensionless relative value, with the resulting values of each characteristic quantity being in the same order of magnitude. There are many methods of data scaling, such as Min-Max Normalization, Min-Max Normalization, also known as the extreme difference method, is the simplest way to deal with the magnitude problem, which is to scale the value of a column in the data set to between 0 and 1. It is calculated as (2). A single element is denoted as $X$, the minimum value in the dataset is denoted as $X_{min}$, and the maximum value in the data set is denoted as $X_{max}$.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}. \quad (2)$$

This is a linear transformation of the original data. The Min-Max normalization method preserves the interrelationship between the original data, but if after normalization, the new input data exceeds the range of values of the original data, i.e., it is not in the original interval $[X_{min}, X_{max}]$, an out-of-bounds error will be generated. Therefore, this method is suitable for cases where the range of values of the original data has been determined.

Mean normalization is similar to Min-Max normalization, with the difference that the best value in the numerator is replaced by the mean value $u$. It can be calculated using (3).

$$X' = \frac{X - u}{X_{max} - X_{min}}. \quad (3)$$

This method scales the data to the interval $[-1,1]$ with an average value of 0. In this paper, the data are scaled to between $[0, 1]$ using the extreme difference method.

4) RAW TIME SERIES TO CONSTRUCT SUPERVISED DATA
Supervised learning is a problem with an input variable ($X$) and an output variable ($Y$), and an algorithm can be used to learn the mapping function $y = f(x)$ from $x$ to $y$. The goal of the algorithm is to approximate the true mapping relationship well enough so that when new input data ($X$) is available, the output variable ($Y$) of that data can be predicted. A supervised learning problem is obtained by shifting the time series forward by a one-time step.

5) DATASET
There are two datasets used for the experiments in this paper: the “int” traffic dataset and the “isp” traffic dataset. The “int” traffic dataset was collected from 09:30 on November 19, 2004, to 11:11 on January 27, 2005. As shown in Fig. 4, data were collected every five minutes. The “isp” traffic dataset was from a private ISP with centers in 11 European cities. These data correspond to a transatlantic line and were collected from June 7, 2005, 06:57 to July 31, 2005, 11:17. Data were collected every five minutes, as shown in Fig. 5.

B. MODEL BUILDING
In this paper, we use the LSTM network to construct a network traffic prediction model. The input of the neural network based on transfer learning is the network traffic of the previous time of the backbone network, and the output
result is the network traffic of the latter time. After completing the corresponding training, the network traffic of the previous time of the core network is used as the input, and the output result is the traffic of the core network at a later time. The LSTM network traffic prediction model based on transfer learning is obtained after the training is completed. The neural network model based on transfer learning is shown in Fig. 6. The input of the directly trained neural network is the network traffic of the core network at the previous time, and the output result is the network traffic of the later time, and the directly trained LSTM network traffic prediction model is obtained after the training is completed.

The forward propagation algorithm of LSTM is shown in Fig. 6. Update the output of the forgetting gate: the forgetting gate controls whether to forget the hidden cell state of the previous layer, and the input is the hidden state $h_{t-1}$ of the previous moment and the input data $x_t$ of the current moment, defining the weight $W_f$, the bias $b_f$, the weight $U_f$ and activation function $sigmoid$; and the second part consists of the hidden state $C_t$ and $tanh$ activation function.

The last step is to update the predicted output of the current moment: define the weights $V$ and bias $c$, and then define the activation function, generally the $sigmoid$ function, to get the predicted output of the current moment.

The backpropagation algorithm of LSTM is shown in Fig. 6. It defines $L$ as the loss function and updates the parameters by chaining the derivative rule to achieve conditional satisfaction. Although the structure of LSTM is quite complex, we can use it effectively with some API support.

The network traffic prediction model based on LSTM and transfer learning constructed in this paper uses mean squared error (MSE) as the loss function. In mathematical statistics, the mean square error refers to the expected value of the square of the difference between the estimated value of the parameter and the true value of the parameter. MSE can evaluate the degree of change in the data. The smaller the value of MSE, the better the accuracy of the prediction model in describing the experimental data. Moreover, as the error decreases, the gradient also decreases, which is beneficial to convergence, and even with a fixed learning rate, it can converge to the minimum value faster. It can be calculated by (4). The actual value is represented by $y_t$ in the equation, the predicted value is represented by $\hat{y}_t$, and the amount of data in the data set is defined using $m$. The model uses Adam as the optimizer and sets the learning rate to 0.02. The main advantage of Adam is that, after bias correction, each iteration
of the learning rate has a certain range, which makes the parameters relatively stable.

\[ \text{MSE} = \frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2. \]  

(4)

C. PARAMETER TRANSFER

1) MODEL SAVING

In the use of transfer learning, the data in the source domain is used to train the model to get a better model, but in the actual application, it is not possible to train it first and then use it, which will increase the time consumption. Therefore, it is possible to save the previously trained model and then load it when you need to use it. One way is to save the whole model and then load it directly, but this will consume more memory; the other way is to save only the parameters of the model. All we have to do is to save the dictionary and call it, then create a new model with the same structure when we need it, and import the saved parameters into the new model.

2) MODEL LOADING

A neural network is an operational model that consists of a large number of nodes and their mutual connections. Each node represents a specific output function, called the activation function. Each connection between two nodes represents a weighted value for the signal passing through the connection, called the weight, which is equivalent to the memory of an artificial neural network. Depending on the model saving method, there are some differences in the model loading method. Saving the complete model means that both the model structure and the model parameters are saved, and when loading the model, you can choose to load all of them or transfer only the model parameters to the model with the same structure as the original model. When only the model parameters are saved, you can only choose to reconstruct the model with the same structure as the original model and then transfer the parameters, and you cannot load the model directly. The method proposed in this thesis saves the complete model and chooses to load all of it when the model is loaded.

IV. EVALUATION RESULTS

This section establishes two network traffic prediction models based on LSTM, namely the transfer learning model and the direct training model. After the training is completed, the network traffic prediction models are obtained and their loss function curves are drawn. The advantages of the transfer learning model in the training process are demonstrated by comparing the loss function curve. Perform the prediction task using the two models separately, listing the results in a table. By comparing the evaluation indexes of the two models, the advantages of the transfer learning model in the prediction process are demonstrated.

| Models                      | Knowledge transfer direction     | RMSE  |
|-----------------------------|----------------------------------|-------|
| Transfer model              | backbone network domain          | 0.0432|
| Direct training model       | core network domain              | 0.0843|

A. EXPERIMENTAL SETUP

We performed ablation experiments to ensure that the parameters used were superior before building the transfer learning model and direct training the model.

1) TRANSFER LEARNING MODEL

1. An LSTM neural network model is constructed using PyTorch, with step size set to 10, i.e., the first 10 rows of the dataset are used for prediction; batch size set to 10, i.e., the number of samples processed in each batch is 10, and input size set to 1. The model optimizer is Adam, and the learning rate is set to 0.02.

2. Intercept 10,000 rows of samples from the “int” dataset in the academic backbone network domain to train it. Set the epoch to 50.

3. Transfer the model to the core network domain and train it again using 1000 rows of samples from the “isp” dataset. Set the epoch to 50.

4. Test the prediction accuracy of the model with 10,000 samples from the “isp” dataset.

2) DIRECT TRAINING MODEL

1. An LSTM neural network model is constructed using PyTorch, with step size set to 10, i.e., the first 10 rows of the dataset are used for prediction; batch size set to 10, i.e., the number of samples processed in each batch is 10, and input size set to 1. The model optimizer is Adam, and the learning rate is set to 0.02.

2. Train with samples from the 1000 rows of the “isp” dataset. Set the epoch to 50.

3. Test the prediction accuracy of the model with samples from the 10,000 row “isp” dataset.

B. EXPERIMENTAL EFFECT DEMONSTRATION

In this paper, the prediction accuracy of the transfer learning model and the direct training model after 50 epochs of training using 1000 rows of data is shown in TABLE 1, and it can be seen that the transfer learning model shows better results than the direct training model.

Root mean square error (RMSE) is the square root of the ratio of the square of the deviation of the predicted value from the true value to the number of observations. It measures the deviation between the predicted and true values and is more sensitive to outliers in the data. The calculation method of RMSE is shown in (5). The actual value is represented by \( y_i \) in the equation, the predicted value is represented by \( \hat{y}_i \), and
FIGURE 7. Loss functions of different data volumes are used for transfer learning model and direct learning model.

TABLE 2. Prediction accuracy of reverse transfer (1000row).

| Models               | Knowledge transfer direction        | RMSE  |
|----------------------|-------------------------------------|-------|
| Transfer model       | core network domain → backbone      | 0.06  |
| Direct training model| backbone network domain             | 0.112 |

The amount of data in the data set is defined using $m$.

\[
RMSE = \sqrt{\frac{1}{m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2}.
\] (5)

The loss function curves of the transfer learning model and the direct training model are printed in Figs. 7a and 7d, from which it can be seen that the loss function of the transfer learning model starts to decrease around the function value of 0.08 after removing the outliers, while the loss function of the direct training model starts to decrease around the function value of 0.4. This shows that the transfer learning model learns the knowledge in the training in the source domain and can be applied to the target domain to better perform the tasks in the target domain. Therefore, the transfer learning model can have a better starting point and a better ending point in the training process.

The implementation of transfer learning comes from the similarity between the source domain and the target domain. The data in $D_T$ can be learned by the knowledge in $D_S$, and similarly, the data in $D_S$ can be learned by the knowledge in $D_T$, and we list the proofs of reverse transfer in TABLE 2. In this paper, the transfer of learning knowledge from the backbone network to the core network is called forward transfer, and conversely, the transfer of learning knowledge from the core network to the backbone network is called reverse transfer.

Change the amount of training data to 100 rows and 10 rows to see the effect. The loss function curves of the transfer learning model and the direct training model are printed in Figs. 7b, 7e, 7c, and 7f. From the figure, we can see that the starting point of the loss function during training becomes higher for the transfer learning model and the direct training model after changing the amount of training data. Although the training effect of both models decreases, we can see that the training effect of the direct training model decreases more. The curve of the transfer learning model, although more unstable, has a lower starting point and acquires a lower endpoint at the end of training. The accuracy of the transfer learning model and the direct training model after the amount of training data changes is given in TABLE 3 and TABLE 4.

As can be seen from the table, after changing the amount of training data, the prediction effects of both the transfer learning model and the direct training model decrease, and the prediction effect of the direct training model decreases more significantly. When the amount of training data decreases to a certain extent, the prediction model will lose its prediction ability. Therefore, the use of the transfer model can reduce the impact of the reduction in the amount of training data on the prediction ability, so that an acceptable level of error can be obtained using fewer data.

V. CONCLUSION

In this paper, we construct a network traffic prediction architecture based on LSTM and transfer learning, apply transfer learning to continuous time series problems, and build a...
prediction model with high performance in a network traffic prediction scenario. From the forward transfer experiment and the reverse transfer experiment, we can see that the knowledge acquired from the source domain can be applied in the target domain, while the knowledge acquired from the target domain can also be applied in the source domain, so the source and target domains are similar. The results of the comparison experiments show that the transfer learning model has better starting and ending points than the direct training model in the training process with the same amount of data. Compared with the direct training model without transfer learning, the performance of the transfer learning model can be improved by more than 40% in completing the target task after training with the source domain data, which leads to the performance improvement of the network traffic prediction task.

REFERENCES

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022, Cisco, San Jose, CA, USA, 2019.
[2] X. Zhang, Y. Wang, M. Yang, and G. Geng, “Toward concurrent video multicast orchestra- tion for caching-assisted mobile networks,” *IEEE Trans. Veh. Technol.*, vol. 70, no. 12, pp. 13205–13220, Dec. 2021, doi: 10.1109/TVT.2021.3119429.
[3] X. Zhang, Y. Wang, G. Geng, and J. Yu, “Delay-optimized multicast tree packing in software-defined networks,” *IEEE Trans. Services Comput.*, early access, Aug. 20, 2021, doi: 10.1109/TSC.2021.3106264.
[4] N. Sadek and A. Khothonadzad, “Multi-scale high-speed network traffic prediction using k-factor Gegenbauer ARMA model,” in *Proc. IEEE Int. Conf. Commun.*, Jun. 2004, pp. 2148–2152.
[5] H. Z. Moayedi and M. A. Masnadi-Shirazi, “Arima model for network traffic prediction and anomaly detection,” in *Proc. Int. Symp. Inf. Technol.*, vol. 4, Aug. 2008, pp. 1–6.
[6] C. G. Dethle and D. G. Wadke, “On the prediction of packet process in network traffic using FARIMA time-series model,” *J. Indian Inst. Sci.*, vol. 84, nos. 1–2, p. 31, 2013.
[7] W. Chen, Z. Shang, and Y. Chen, “A novel hybrid network traffic prediction approach based on support vector machines,” *J. Comput. Netw. Commun.*, vol. 2019, pp. 1–10, Feb. 2019.
[8] X. Xiao, H. Duan, and J. Wen, “A novel car-following inertia gray model and its application in forecasting short-term traffic flow,” *Appl. Math. Model.*, vol. 87, pp. 546–570, Nov. 2020.
[9] Y. Xu, F. Yin, W. Xu, J. Lin, and S. Cui, “Wireless traffic prediction with scalable Gaussian process: Framework, algorithms, and verification,” *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1291–1306, Jun. 2019.
[10] B. G. Gétiner, M. Sari, and O. Borat, “A neural network based traffic-flow prediction model,” *Math. Comput. Appl.*, vol. 15, no. 2, pp. 269–278, 2010.
[11] A. Sherstinsky, “Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network,” *Phys. D. Nonlinear Phenomena*, vol. 404, Mar. 2020, Art. no. 132306.
[12] R. W. Liu, M. Liang, J. Nie, W. Y. B. Lim, Y. Zhang, and M. Guizani, “Deep learning-powered vessel trajectory prediction for improving smart traffic services in maritime Internet of Things,” *IEEE Trans. Netw. Sci. Eng.*, early access, Jan. 7, 2022, doi: 10.1109/TNSRE.2022.3140529.
[13] R. W. Liu, M. Liang, J. Nie, Y. Yuan, Z. Xiong, H. Yu, and N. Guizani, “STMGCN: Mobile edge computing-powered vessel trajectory prediction using spatio-temporal multi-graph convolutional network,” *IEEE Trans. Ind. Informat.*, early access, Apr. 8, 2022, doi: 10.1109/TII.2022.3165886.
[14] H. D. Trinh, L. Giupponi, and P. Dini, “Mobile traffic prediction from raw data using LSTM networks,” in *Proc. IEEE 29th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun.* (PIMRC), Sep. 2018, pp. 1827–1832.
[15] N. Ramakrishnan and T. Soni, “Network traffic prediction using recurrent neural networks,” in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl.* (ICMLA), Dec. 2018, pp. 187–193.
[16] H. Lu and F. Yang, “Research on network traffic prediction based on long short-term memory neural network,” in *Proc. IEEE 4th Int. Conf. Comput. Commun.* (ICCC), Dec. 2018, pp. 1109–1113.
[17] K. Weiss, T. M. Khosgoftaar, and D. Wang, “A survey of transfer learning,” *J. Big Data*, vol. 3, no. 1, pp. 1–40, Dec. 2016.
[18] S. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, pp. 1345–1359, Nov. 2010.
[19] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Müller, “Transfer learning for time series classification,” in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1367–1376.
[20] K. Kashifarekh, J. Narwariya, P. Malhotra, L. Vig, and G. Shroff, “ConvTimeNet: A pre-trained deep convolutional neural network for time series classification,” in *Proc. Int. Joint Conf. Neural Inform. Process.* (IJCNN), Jul. 2019, pp. 607–612.
[21] D. Kearney, S. McLoone, and T. E. Ward, “Investigating the application of transfer learning to neural time series classification,” in *Proc. 30th Irish Signals Syst. Conf.* (ISSC), Jun. 2019, pp. 1–5.
[22] Y. Yu, X. Si, C. Hu, and Z. Jianxun, “A review of recurrent neural networks: LSTM cells and network architectures,” *Neural Comput.*, vol. 31, no. 7, pp. 1235–1270, Jul. 2019.

HUI LIU was born in Linyi, Shandong, China, in 1995. He received the B.S. degree from Weifang University. He is currently pursuing the M.S. degree with the Qilu University of Technology. His main research interests include computer networks, networked control systems, and computer network reliability.

HAO XU was born in Weifang, Shandong, China, in 1997. He received the B.S. degree from the Qilu University of Technology, where he is currently pursuing the M.S. degree. His main research interests include computer networks, networked control systems, and computer network reliability.

XINCHANG ZHANG (Senior Member, IEEE) received the M.S. degree from the Shandong University of Science and Technology, China, in 2005, and the Ph.D. degree from the Computer Network Information Center, Chinese Academy of Sciences, China, in 2010. He is currently a Professor at the Qilu University of Technology (Shandong Academy of Sciences). He has over 40 papers in research journals, such as IEEE *Journal on Selected Areas in Communications* (JSAC), *IEEE Transactions on Services Computing* (TSC), *IEEE Transactions on Vehicular Technology* (TVT), and *IEEE Communications Magazine* and international conference proceedings. His research interests include network protocols and architectures, and cloud computing. He won the Shandong (in China) Science and Technology Progress Awards, in 2013, 2018, and 2019, respectively.