sustainability

MDPI

Article

Demand Forecasting of E-Commerce Enterprises Based on Horizontal Federated Learning from the Perspective of Sustainable Development

Juntao Li [1], Tianxu Cui [1,*], Kaiwen Yang [1], Ruiping Yuan [1], Liyan He [1] and Mengtao Li [2,*]

[1] School of Information, Beijing Wuzi University, Beijing 101149, China; ljtletter@126.com (J.L.); ykw1234@163.com (K.Y.); angelholyping@163.com (R.Y.); hly199808@163.com (L.H.)
[2] School of Business Administration, Northeast University of Finance and Economics, Dalian 116025, China
* Correspondence: cuitianxubwu@163.com (T.C.); limtao@dufe.edu.cn (M.L.)

Abstract: Public health emergencies have brought great challenges to the stability of the e-commerce supply chain. Demand forecasting is a key driver for the sound development of e-commerce enterprises. To prevent the potential privacy leakage of e-commerce enterprises in the process of demand forecasting using multi-party data, and to improve the accuracy of demand forecasting models, we propose an e-commerce enterprise demand forecasting method based on Horizontal Federated Learning and ConvLSTM, from the perspective of sustainable development. First, in view of the shortcomings of traditional RNN and LSTM demand forecasting models, which cannot handle multi-dimensional time-series problems, we propose a demand forecasting model based on ConvLSTM. Secondly, to address the problem that data cannot be directly shared and exchanged between e-commerce enterprises of the same type, the goal of demand information sharing modeling is realized indirectly through Horizontal Federated Learning. Experimental results on a large number of real data sets show that, compared with benchmark experiments, our proposed method can improve the accuracy of e-commerce enterprise demand forecasting models while avoiding privacy data leakage, and the bullwhip effect value is closer to 1. Therefore, we effectively alleviate the bullwhip effect of the entire supply chain system in demand forecasting, and promote the sustainable development of e-commerce companies.

Keywords: horizontal federated learning; e-commerce enterprise demand forecasting; time-series analysis; LSTM

Citation: Li, J.; Cui, T.; Yang, K.; Yuan, R.; He, L.; Li, M. Demand Forecasting of E-Commerce Enterprises Based on Horizontal Federated Learning from the Perspective of Sustainable Development. *Sustainability* **2021**, *13*, 13050. https://doi.org/10.3390/su132313050

Academic Editors: Yi Zhang, Guowei Hua, Edwin Cheng and Weihua Liu

Received: 30 September 2021
Accepted: 22 November 2021
Published: 25 November 2021

1. Introduction

Since December 2019, COVID-19 has spread widely, with more than 200 countries and territories having confirmed cases within just in a few months. It has had a great impact on daily life and productivity at work. How to improve the coordination, stability, and sustainability of the supply chain system [1] is an important issue for e-commerce enterprises. Privacy and security are fundamental guarantees for the sustainable development of e-commerce companies. With the increasing activity of cross-border e-commerce transactions, Internet user information leaks occur frequently, and a large number of personal and corporate data have been leaked and abused. How to better strengthen the protection of personal information has become a top priority; thus, increasing consumer confidence in participating in e-commerce, improving the efficiency of e-commerce transactions, and establishing an open, transparent, and efficient privacy protection mechanism have become hot issues of great concern to all sectors of society.

For a long time, in the demand forecast of e-commerce companies, due to limited information and communication capabilities, the immaturity of information sharing and exchange technology, and the protection of trade secrets, it is impossible for certain e-commerce companies to conduct centralized information sharing modeling. Each node

is more akin to an information island. Even different departments in the same enterprise still have the problem of asymmetry of demand information, and, as such, the accuracy of demand forecasting models cannot be improved. If the situation goes on in this way, the inventory waste problem of e-commerce companies will become more and more serious.This may lead to a bullwhip effect through the entire supply chain system of e-commerce companies, which can seriously affect the sustainable development of e-commerce companies.

Data are the carriers of information. In the era of big data, only when data between different data sources can be shared and circulated openly can the data island be broken and its potential value be brought into play. However, privacy security is extremely important for countries, enterprises, and individuals. With the development and progress of society, governments, and organizations in various countries and regions have become increasingly aware of the importance of data privacy protection. A series of policies and regulations have been introduced to protect the security of private data; for example, the General Data Protection Regulation (GDPR) [2] officially implemented by the European Union in May 2018 has put forward stringent requirements for data security. This is currently the most comprehensive and widely used privacy protection law in the world.

Federated learning is a distributed machine learning framework [3], which aims to address the problems of user privacy and data islands in the process of machine learning. It can perform machine learning model training and inference work without the data coming out locally. Since Google first proposed Federated Learning in 2016, it has led to considerable progress in algorithms, frameworks, privacy, and incentive mechanisms.

To prevent the potential privacy leakage of e-commerce enterprises in the process of demand forecasting using multi-party data, as well as improve the accuracy of demand forecasting models, we propose an e-commerce enterprise demand forecasting method based on Horizontal Federated Learning and ConvLSTM, from the perspective of sustainable development. First, in view of the shortcomings of traditional RNN and LSTM demand forecasting models, which cannot handle multi-dimensional time-series problems, we propose a demand forecasting model based on ConvLSTM. It can extend the one-dimensional input LSTM to the multi-dimensional input. Through the expansion of FC-LSTM [4], a convolution structure is added during the state transition, such that the network can capture the spatial characteristics of the data while dealing with timing issues. Secondly, to address the problem that data cannot be directly shared and exchanged between e-commerce enterprises of the same type, centralized machine learning modeling is carried out. The introduction of the framework of Federated Learning allows e-commerce enterprises to achieve the goal of demand information sharing modeling indirectly, through Horizontal Federated Learning, under the premise that private data is not available locally. We take the sales data set of 1000 products of JD from 1 January 2016 to 31 December 2017 as the research object, and divide the data into six parts according to the warehouse code to simulate the scenario of Horizontal Federated Learning by multiple e-commerce enterprises. In addition, we assume that the six participants are honest, the server is honest but curious, the communication in each round is smooth, and there is no data packet loss. Experimental results on a large number of real data sets show that, compared with benchmark experiments, our proposed method can improve the accuracy of e-commerce enterprise demand forecasting models while avoiding privacy data leakage, and the bullwhip effect value is closer to 1. Therefore, we effectively alleviate the bullwhip effect of the entire supply chain system in demand forecasting, and promote the sustainable development of e-commerce companies.

Our main contributions are as follows:

- Aiming at the shortcomings of traditional RNN and LSTM demand forecasting models that cannot handle the problem of multiple feature sequences, we propose a demand forecasting model based on ConvLSTM. It can extend the one-dimensional input LSTM to a multi-dimensional input model. Through the expansion of FC-LSTM,

a convolution structure is added during state transition, such that the network can capture the spatial characteristics of the data while dealing with timing issues.

- According to the problem of centralized machine learning modeling, regarding the inability to directly share and exchange demand forecast-related data between e-commerce enterprises, we introduce the framework of Federated Learning, such that e-commerce companies of the same type can indirectly achieve the goal of demand information sharing modeling through Horizontal Federated Learning, under the premise that private data is not available locally, thus avoiding the leakage of private data.
- Experimental results on a large number of real data sets show that, compared with benchmark experiments, our proposed method improves the accuracy of the e-commerce enterprise demand prediction model while avoiding the leakage of private data, and the bullwhip effect value is even greater—close to the target value of 1—effectively alleviating the bullwhip effect of the entire supply chain system on demand forecasting.

The remainder of this paper is arranged as follows: We summarize the related literature in the Section 2. Method and results are introduced in the Section 3. In the Section 4, we conclude the paper and detail lines for future research. To the best of our knowledge, this paper is the first to apply the theory of Federated Learning to the demand prediction scenario of e-commerce enterprises, which not only considers the accuracy of the demand prediction model, but also the potential privacy data leakage problem of e-commerce enterprises in the process of information sharing. From the perspective of sustainable development, we propose a new demand forecasting method, in order to ensure that data providers in the supply chain are treated fairly, and to encourage more e-commerce companies to participate in the supply chain system to promote the establishment of a sustainable supply chain.

2. Literature Review

We summarize the related literature from the following three aspects: Federated Learning, E-commerce enterprise demand forecasting, and LSTM.

2.1. Federated Learning

In 2016, the Google team [5] first proposed the concept of Federated Learning based on Shokri. In Federated Learning, participants store all training data locally, train the model locally, and then upload the trained model updates to the cloud. Other participants download the updates to their mobile devices to improve the training model's performance, using the method of Federated Learning to avoid potential privacy leakage problems in the process of centralized machine learning. Yang et al. [6], among others, have divided Federated Learning into three categories, according to the data distribution of the participants, namely Horizontal Federated Learning (HFL), Vertical Federated Learning (VFL), and Transfer Federated Learning (TFL). The typical application scenario of Horizontal Federated Learning is the end-cloud service architecture, which is mainly aimed at users or enterprises with large amounts of homogeneous data, as shown in Figure 1. Under HFL, the data sets owned by each participant have similar characteristics, but different users are involved [3]. For example, two e-commerce companies from different regions have different customer groups, such that their customer intersection is very small; that is, their data sets have different sample IDs. However, the feature space of the data set related to the demand forecast of a single product is similar; that is, their business models are very similar. Therefore, these two e-commerce companies can work together to conduct Horizontal Federated Learning, in order to build a better demand forecasting model.

For the Horizontal Federated Learning scenario, Reza et al. [7] have proposed a collaborative deep learning training mechanism based on privacy protection. Under this mechanism, all participants independently train the model and only share a subset of the gradient parameters to be updated. In 2017, Mcmahan et al. [8] proposed a Horizontal Federated Learning framework that can be used to solve the model update problem in mobile

communication terminal scenarios. Under this framework, users use mobile communication terminals to perform model parameter update training locally and then upload the learned model parameters to a public intermediate cloud platform, such that different end-users can jointly train a global model without the original data being local. Kallista et al. [9] introduced a safe gradient aggregation method that can protect the privacy of user gradients under the framework of Federated Learning. Yoshinori et al. [10] have used homomorphic encryption technology to encrypt the model parameter aggregation process, in order to ensure that the central server does not cause privacy leakage. Virginia et al. [11] realized a Federated Learning system in a multi-task mode, which enables multiple participants to complete tasks independently, and proposed the communication overhead and error tolerance issues that the multi-task training model focuses on. Mcmahan et al. [8,12] constructed a secure client-server architecture for a Federated Learning system, which allows the client to build a local model by themself and build a global federated model by interacting with the server. The method of model construction ensures that there will be no leakage of user privacy data during the entire process. Konečný et al. [5] improved the training efficiency of the central model by increasing the communication overhead. Lin et al.[13] proposed a compression method called deep gradient compression to solve the problem of communication bandwidth overhead in large-scale distributed training.

To sum up, the research on Federated Learning mainly focuses on privacy protection [14–17] and incentive mechanism [18–21]. At present, the application of Federated Learning in market segments is also gradually developing [22–24]. However, there are few examples of literature on the application of Federated Learning to forecasting [25–29]; especially, the application of Federated Learning to the sustainable development research of e-commerce enterprise demand forecasting has not yet been found.

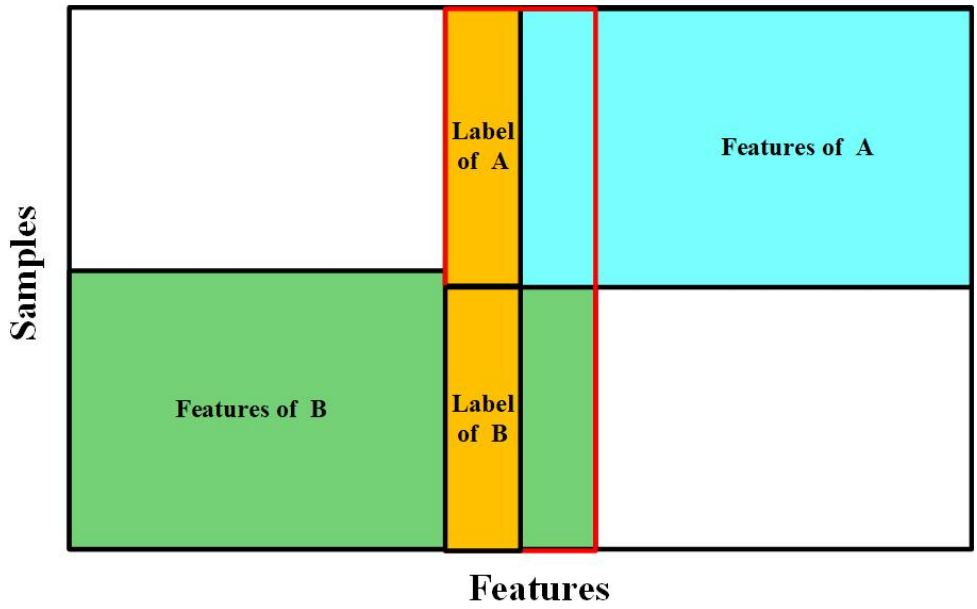

Figure 1. Schematic diagram of Horizontal Federated Learning.

2.2. E-commerce Enterprise Demand Forecasting

Demand forecasting, as an important basis for the production and operation of e-commerce enterprises, has gradually become the core force driving the development of the entire supply chain. An accurate demand forecast plays a vital role in ensuring the healthy operations of an e-commerce enterprise. At present, there are three main methods for researching demand forecasting: traditional statistics, machine learning, and combined forecasting.

Demand forecasting models based on traditional statistics mainly include regression forecasting [30,31], exponential smoothing [32–34], ARIMA model [35,36], moving aver-

age [37], and combined forecasting. The demand forecasting model based on traditional statistics is mainly used in the situation where the volume of early sales data is small. Luo et al. [38] considered the problem of low sales forecast accuracy of a large online supermarket, and analyzed historical sales data based on the ARIMA model to predict products. Demand can be analyzed to dynamically obtain the optimal ordering strategy for various commodities. Traditional statistical models are mainly used for linear data forecasting, but it is difficult to make accurate and effective forecasts for problems with large sales volatility and many influencing factors.

With the continuous maturity of related theories, such as big data and machine learning, some scholars have proposed a demand forecasting algorithm based on machine learning. These mainly include neural network [39–42], Markov [43], and grey prediction [44] models. Compared with demand forecasting models based on classical statistics, the demand forecasting model based on machine learning not only takes into account the consumer factors, but also takes into account influencing factors, such as seasons and promotions, in order to improve the forecasting accuracy. For example, Hu [45] proposed a grey prediction model based on a genetic algorithm to further improve the prediction accuracy of the G(1,1) model. Meanwhile, Liu et al. [46] used artificial bee colonies, which optimizes the fitting of polynomial parameters in the life-cycle demand forecast model to achieve high-precision forecasting effects. Adamowski [47] and Wang [48] have used neural network algorithms to build forecasting models. Noori [49] constructed an online five-day oxygen demand prediction model through the support vector calculation method, in order to reduce the uncertainty in the prediction process. Cao et al. [50] introduced an immune particle swarm algorithm to improve support vector machine of the parameter selection in the prediction model. Chalmeta et al. [51] used big data demand forecasting to analyze and identify the impact of supply chain sustainability on the past and predict its impact on the future.

Due to the numerous influencing factors and the complex and irregular characteristics of historical data, in actual forecasting, the forecast accuracy of a single forecasting model is generally low, such that research on combined forecasting models has gradually become a trend [52]. Thomas et al. [53] combined regression analysis and time-series methods to predict the bid price index and concluded that the prediction result of the combined prediction model was better than the prediction results of the two models. Chen et al. [54] proposed a new non-parametric estimation method for time-varying forecast combination weights. Franses et al. [55] used a Bayesian predictive combination algorithm to predict the number of COVID-19 cases. This algorithm is suitable for the prediction of three non-nested diffusion models of S-shaped processes, such as virus diffusion. Cerqueira et al. [56] used the idea of model compression to solve the high computational cost and lack of transparency in time-series forecasting tasks. Chen et al. [57] proposed a new time-series forecasting method that introduced a non-linear combination of error decomposition and predictor, and constructed two-hybrid systems and two artificial intelligence (AI) models for non-linear modeling and combination. Choi et al. [58] designed a new customized power demand prediction algorithm based on the LSTM deep learning method for the recent power demand patterns. Comparative experiments were carried out in three aspects: short-term, long-term, and seasonal prediction experiments.

To sum up, after the traditional stages of statistical forecasting, machine learning and combined forecasting, the accuracy of demand forecasting model has been significantly improved. However, when using multi-party data for forecasting, it still faces the problems of data island and data privacy.

2.3. LSTM

An LSTM neural network is an improved version of the RNN neural network, which mainly solves the problem of gradient disappearance, allowing the network to remember the content for a longer time and make the network more reliable [59,60]. In recent research, for time-series data with multiple seasonal patterns, Bandara et al. [61] proposed Long

Short-Term Memory Multi-Seasonal Net (LSTM-MSNet), which is a unified predictor based on decomposition frame. Xu et al. [62] used daily data of the Shanghai Stock Exchange Index and the Dow Jones Index as the research objects, and they used RNNs and LSTMs to build models, respectively, in order to compare the pros and cons of LSTM in time-series forecasting. Chimmula et al. [63] introduced a long short-term memory (LSTM) network, which is a deep learning method for predicting future COVID-19 cases, using the results of the LSTM network to predict the possible end-point of the outbreak. Abbasimehr et al. [64] proposed a demand forecasting method based on a multi-layer LSTM network. It has the ability to capture non-linear patterns in time-series data, while taking into account the inherent characteristics of non-stationary time-series data. Recently, Lu et al. [65] used the preliminary data of Haikou online car-hailing orders provided by Didi Travel's GAIA Initiative to predict the short-term demand for online car-hailing services. The model is suitable for short-term forecasting network car-hailing demand forecasting with temporal and spatial characteristic information. Agga et al. [66] studied the short-term self-use photovoltaic power generation forecast based on hybrid CNN-LSTM and ConvLSTM models.

To sum up, LSTM demand forecasting model considering spatio-temporal characteristic information has become a research trend, and it is more widely used in practical production and life. Therefore, this paper uses conv LSTM network structure to model when forecasting the demand of e-commerce enterprises.

3. Method and Results

3.1. Preliminary Knowledge and Definition

3.1.1. Typical Horizontal Federated Learning

The main application scenario of Horizontal Federated Learning is that there is significant overlap in participant features but few in the participant samples. Horizontal Federated Learning uses the data of different participants to establish a valuable Federated Learning model under the premise of ensuring participant privacy through a decentralized and distributed modeling method. In a Horizontal Federated Learning system [6], suppose that there are k participants with the same data structure collaboratively learning a machine learning model through a parameter or cloud server. The data set owned by each participant is D_k, $\mathrm{k} \in \mathrm{K} = \{1, 2, \cdots, \mathrm{K}\}$. A basic assumption is that the participants are honest and the server is honest but curious. Therefore, no participant is allowed to leak information to the server. As shown in Figure 2, a typical Horizontal Federated Learning process mainly consists of the following five steps:

Step 1: The server sends a global model M^t to all participants (data providers).

Step 2: Participant k uses the local data set D_k to train locally to obtain the local model M_k^t and then uses cryptographic techniques, such as homomorphic encryption [6], differential privacy, or secret sharing, to encrypt the model. Then, they send the encrypted result M_k^t to the server.

Step 3: The server safely aggregates the encryption model M_k^t from participant k to obtain a new global model M^{t+1} without any learning information about the participants.

Step 4: The server sends the summarized result to participant k again, according to a pre-determined rule.

Step 5: Participants use the decrypted gradient to update their respective local models.

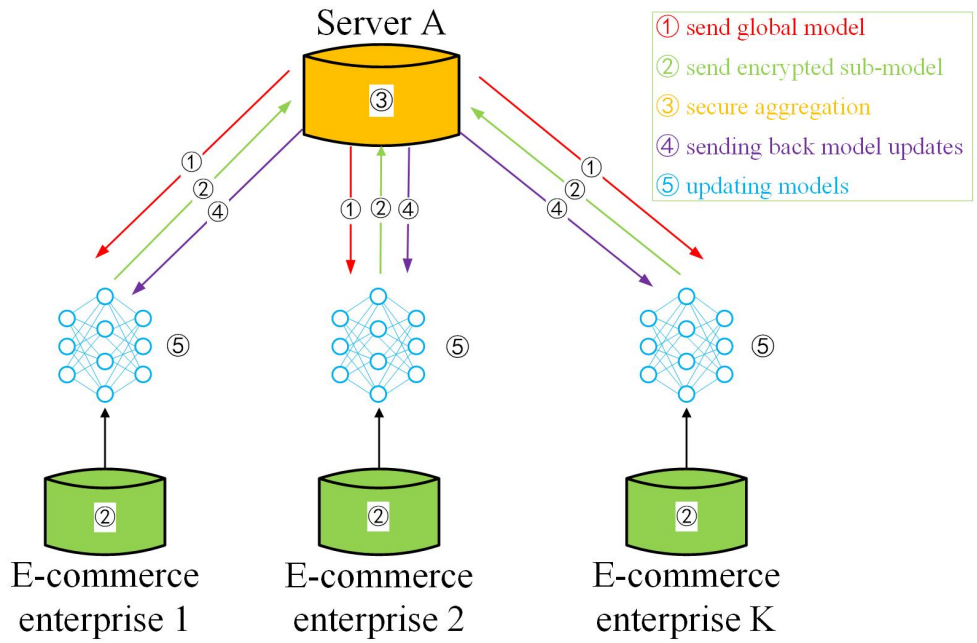

Figure 2. Horizontal Federated Learning process.

Through the above iterative process, the loss function converges or the iteration round requirements (set in advance) are reached, such that the entire training process is finally completed. This architecture is independent of specific machine learning algorithms (e.g., decision trees, LSTM, DNN), and all participants share the final model parameters. The model aggregation in step 3 is carried out as shown below. First, the server calculates the gradient of each participant k:

$$\Delta_k^t = M_k^t - M^t. \tag{1}$$

Then, the server obtains the weighted average of the gradients of all participants:

$$\Delta^t = \sum_{k=1}^{K} \frac{|D_k|}{\sum_{k=1}^{K} |D_k|} \cdot \Delta_k^t, \tag{2}$$

where $|D_k|$ is the size of the training data D_k. Finally, the server calculates a new global model, according to the gradient descent:

$$M^{t+1} = M^t + \Delta^t. \tag{3}$$

It is worth noting that, as we are in the entire Horizontal Federated Learning process, the local data is not uploaded to the server, nor does it reach any participants. Furthermore, the gradient generated by the local training also uses the necessary cryptographic encryption technology in the process of transmission, such that the model will not be reversed, and the entire training process is safe and reliable.

3.1.2. Neural Network Based on Time-Series

(I) Recurrent Neural Network (RNN)

The key of Recurrent Neural Network (RNN) is looping; that is, in the process of information transmission, part of the information is retained in the hidden neuron in each cycle and will flow through the next neuron, together with the new information, finally being transmitted to the subsequent output results [67]. Although the input layer, the hidden layer, and the output layer of the traditional artificial neural network are fully connected, the nodes in each layer are not connected. In an RNN network, the nodes in

the hidden layer are connected; that is, the information at the current time point and the information at the previous time point, together, act on the next time point [68]. This is also the basis of principle that an RNN has a memory function. As shown in Figure 3, the middle $h^{l(r,c)}$ represents the neuron of the c feature dimension at the r^{th} time step of the l^{th} hidden layer. The calculation method is

$$h^{l(r,c)} = f\left(z^{l(r,c)}\right) = f\left(w^{l}_{hx} \cdot h^{l-1(r,c)} + w^{l}_{hh} \cdot h^{l(r-1,c)} + b^{l(r,c)}\right), \quad (4)$$

where w_{hx} represents the weight coefficient along the network layer direction, ω_{hh} represents the weight coefficient along the time step direction, and $f(\cdot)$ is the activation function.

Compared with other types of data, time-series data consists of a sequence of elements. To perform feature extraction on sequence data, two factors—features and time sequence—need to be considered. Specifically, for each neuron, its original output $z^{l(r,c)}$ is determined by the neurons $h^{l-1(r,c)}$ at the same time step in the previous hidden layer, and the neuron $h^{l(r-1,c)}$ in the previous time step in the same hidden layer. Therefore, sample feature extraction is performed in the horizontal (same time step) direction, and time sequence analysis is performed in the vertical (same hidden layer) direction. Then, the original output is activated to achieve the purpose of de-linearization. Thereby, the model can handle the non-linear separability situation, the activation output is expressed as $h^{l(r,c)} = f\left(z^{l(r,c)}\right)$, where $f(\cdot)$ is the activation function.

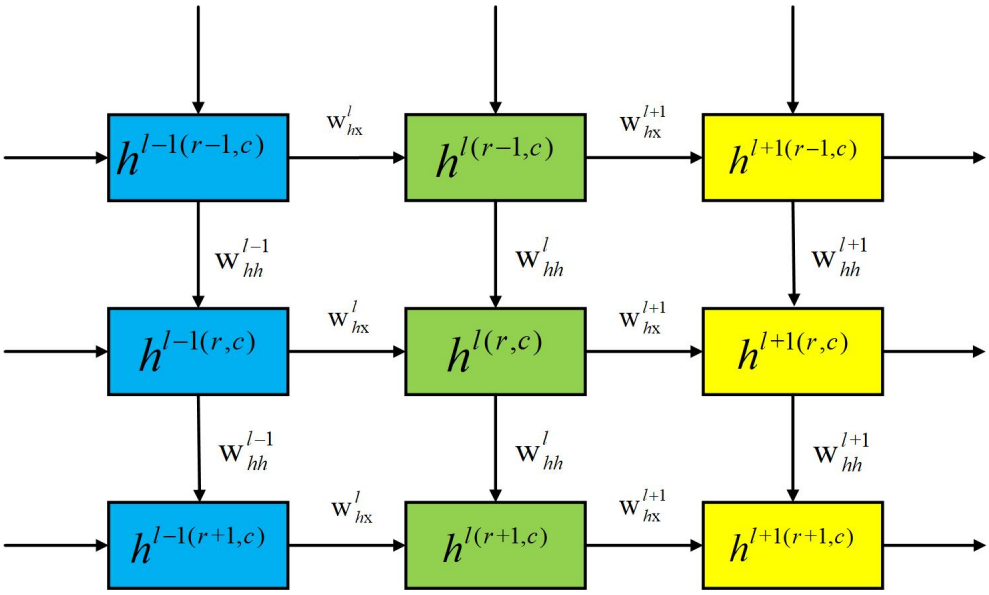

Figure 3. Schematic diagram of recurrent neural network structure.

(II) Long Short-Term Mermory Network (LSTM)

Long Short-Term Mermory Network (LSTM) is a special kind of RNN, which addresses the problem of gradient dispersion caused by long-term spans by introducing logic gate units to realize the memory mechanism and forgetting mechanism [68], therefore resolving the long-term dependence problem [67]. In the process of analyzing time-series data, the different characteristics in the time step direction have different influence on the final output decision. Moreover, as the depth of the hidden layer of the model increases, gradient transfer becomes more difficult, and the model becomes more difficult to train. The feature extraction of sequence data is carried out in two directions, namely the hidden layer direction and the sequence direction. Therefore, the back-propagation error transfer is also carried out in two directions. Especially in the timing direction, back-propagation along the time step can cause gradient dispersion. Its essence is that, over a long time span, the model focuses more on the characteristics of relatively short time steps, and it is easy to ignore the characteristics of far time steps. Eventually, the parameters cannot be updated

and the model training fails. To solve this problem, the long short-term memory network (LSTM) was proposed.

The biggest difference between LSTM and RNN is that LSTM sets a unit state $u^{l(r,c)}$ in each time step (taking the l^{th} hidden layer, the r^{th} time step, and the c^{th} feature dimension of the unit state of a neuron as an example), which helps to obtain a more comprehensive timing state when each neuron is activated, in order to provide a decision basis for the memory mechanism and the forgetting mechanism.

The specific logic gate unit structure of LSTM is shown in Figure 4. First, we take $h^{l-1(r,c)}$ (the neuron at the same time step in the previously hidden layer outputs) and $h^{l(r-1,c)}$ (the neuron at the previous time step in the current layer) as the input of the current layer, and we then carry out weight transformation and obtain the original output $z^{l(r,c)}$. Then, the original output is decomposed into four components, $z_F^{l(r,c)}$, $z_I^{l(r,c)}$, $z_G^{l(r,c)}$, $z_O^{l(r,c)}$, which are de-linearized through the activation function to obtain the four logic gate unit components of forget gate F, input gate I, quasi-cell state G, and output gate O, respectively. Finally, according to Equation (5), the forget gate F and the unit state $u^{l(r-1,c)}$ of the previous time step are multiplied element-wise, and the input gate I and the quasi-cell state G are multiplied element-wise. The two parts of the product are added by element to obtain the unit state $u^{l(r,c)}$ at the current time step; the unit state is activated, according to Equation (6), and element-wise multiplication is performed with the output gate O Product, in order to obtain the final neuron output $h^{l(r,c)}$ at the current time step (time r) of the hidden layer (layer l).

The operational relationship between logic gates is as follows:

$$u^{l(r,c)} = F^{l(r,c)} \otimes u^{l(r-1,c)} + I^{l(r,c)} \otimes G^{l(r,c)}, \tag{5}$$

$$h^{l(r,c)} = O^{l(r,c)} \otimes \tanh\left(u^{l(r,c)}\right). \tag{6}$$

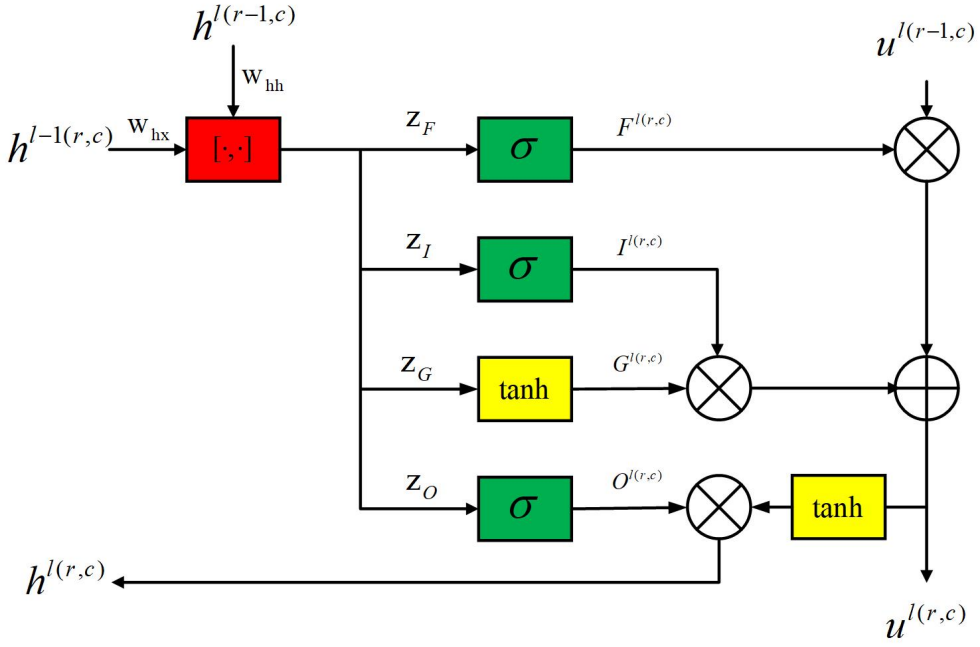

Figure 4. Schematic diagram of long short-term memory network structure.

For convenience, we summarize the defined symbols and their meanings in Table 1 below.

Table 1. List of mathematical symbols.

Symbol	Meaning
k	Participant Index
D_k	Data set owned by participants
M_k^t	Participant k's local model in round t
M^t	Global model for round t
M^{t+1}	Global model for round t+1
Δ_k^t	The gradient change of participant k in round t
Δ^t	The gradient change of the global model in the t round
$h^{l(r,c)}$	The neuron of the c^{th} dimension in the r^{th} time step for the l^{th} layer
$u^{l(r,c)}$	The state of the c^{th} dimension in the r^{th} time step for the l^{th} layer
F;I;G;O	Forget gate; Input gate; quasi-unit state; Output gate
w_{hx}	Weight coefficient along the direction of the network layer
w_{hh}	Weight coefficient along the time step
$\otimes$	Multiply by element
$\oplus$	Add by element
σ	Sigmoid activation function
tanh	Hyperbolic tangent activation function

3.2. Design of E-Commerce Enterprise Demand Forecasting Method Based on HF-ConvLSTM

The typical HF-ConvLSTM-based e-commerce enterprise demand forecasting method framework is depicted in Figure 5, including participant determination, data set deployment, model initialization, model training, model evaluation, model launching, and online reasoning. In this paper, we focus on the algorithms for model training and model evaluation.

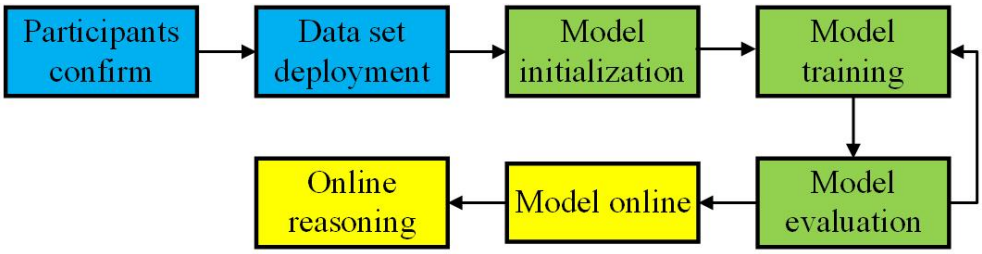

Figure 5. The framework of e-commerce enterprise demand forecasting method based on HF-ConvLSTM.

3.2.1. E-Commerce Enterprise Demand Forecasting Model Based on ConvLSTM

Both RNN and LSTM are suitable for processing one-dimensional time-series data; that is, the dimension scale C of the data sample feature defined above is 1. However, in real e-commerce enterprise demand forecasting scenarios, time-series data are often composed of multiple features. ConvLSTM is a network model suitable for feature extraction from multi-feature time-series data. It introduces a convolution kernel on the basis of the original LSTM for local feature extraction of two-dimensional data and acts on the calculation process of the original output in a single neuron. In addition, in order to obtain the timing status information more deeply, the unit status of the previous time step and the current time step are called in the decomposition process of the original output, and the corresponding weights are assigned to calculate the original output component required by the logic gate unit.

As shown in Figure 6, in the calculation of the original output, the weighted transformation is replaced by the local feature extraction using the convolution kernel; that is,

$$z^{l(r,c)} = \left(\sum_p \sum_q kernel^{(p,q)} \cdot h^{l-1(r+p,c+q)} \right) + w_{hh} \cdot h^{l(r-1,c)} + b, \tag{7}$$

where $p \times q$ is the size of the convolution kernel.

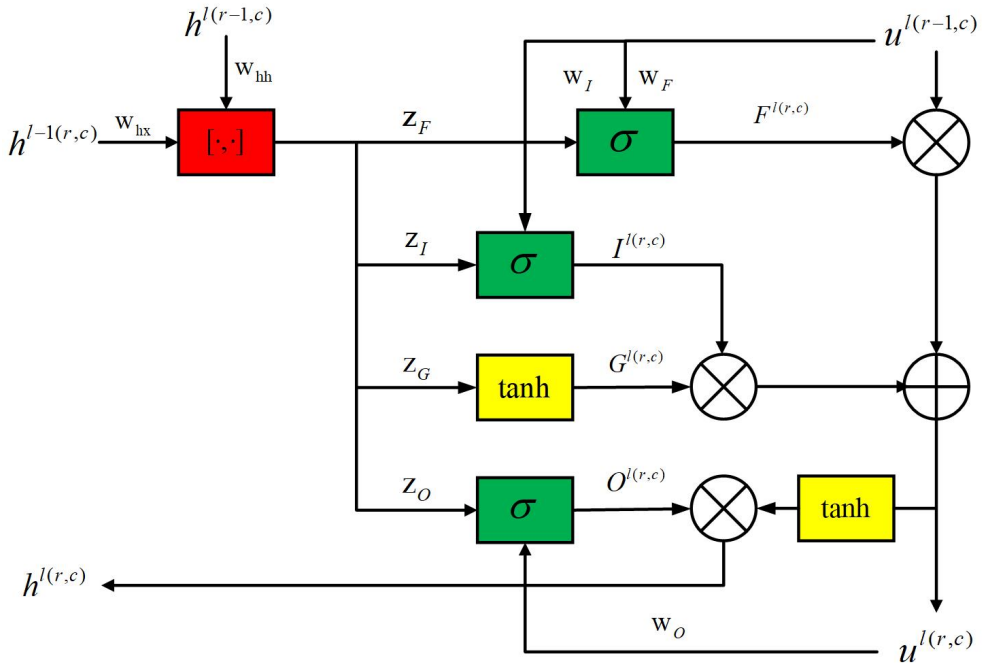

Figure 6. Schematic diagram of ConvLSTM structure.

In terms of decomposition of the original output, the original output is decomposed into the components needed by the four logic gate units. ConvLSTM adds the consideration of the unit state to the direct decomposition method of traditional LSTM. Specifically, according to Equations (8) and (9), the influence of the previous time step unit state is introduced into the forget gate component and the input gate component, respectively, and the quasi-unit state component remains unchanged, according to Equation (10), and the influence of the current time step unit state is introduced in the output gate component, according to Equation (11).

$$z_F \leftarrow z_F + w_F \cdot u^{l(r-1,c)}, \tag{8}$$

$$z_I \leftarrow z_I + w_I \cdot u^{l(r-1,c)}, \tag{9}$$

$$z_G \leftarrow z_G, \tag{10}$$

$$z_O \leftarrow z_O + w_O \cdot u^{l(r,c)}, \tag{11}$$

where w_F, w_I, and w_O are the weights of the unit states in each component. The rest of the network model structure is the same as that in the traditional LSTM.

We use the classic back-propagation algorithm to solve the parameters of the ConvLSTM model. The specific solution process is as follows:

(I) The error loss of the neuron of the c^{th} feature dimension in the r^{th} time step for the l^{th} hidden layer is superimposed with the error of the same time step in the latter layer and the error of the latter time step in the same layer, which is

$$\delta^{l(r,c)} = \frac{\partial Loss}{\partial h^{l(r,c)}} = \frac{\partial Loss^{l+1(r,c)}}{\partial h^{l(r,c)}} + \frac{\partial Loss^{l(r+1,c)}}{\partial h^{l(r,c)}} = \delta^{l+1(r,c)} + \delta^{l(r+1,c)}. \tag{12}$$

(II) The back-propagation error of the output gate is

$$\delta_O^{l(r,c)} = \frac{\partial Loss}{\partial h^{l(r,c)}} \frac{\partial h^{l(r,c)}}{\partial O^{l(r,c)}} = \delta^{l(r,c)} \otimes \tanh\left(u^{l(r,c)}\right). \tag{13}$$

(III) The unit state back-propagation error of the current time step is composed of the back-propagation error of both the current time step and the next time step, namely:

$$\delta_u^{l(r,c)} = \frac{\partial Loss}{\partial h^{l(r,c)}} \frac{\partial h^{l(r,c)}}{\partial u^{l(r,c)}} + \delta_u^{l(r+1,c)} = \delta^{l(r,c)} \otimes O^{l(r,c)} \otimes \tanh'\left(u^{l(r,c)}\right) + \delta_u^{l(r+1,c)}. \quad (14)$$

(IV) The input gate back-propagation error is

$$\delta_I^{l(r,c)} = \frac{\partial \text{ Loss}}{\partial u^{l(r,c)}} \cdot \frac{\partial u^{l(r,c)}}{\partial I^{l(r,c)}} = \delta_u^{l(r,c)} \otimes G^{l(r,c)}. \quad (15)$$

(V) The back-propagation error of the forget gate is

$$\delta_F^{l(r,c)} = \frac{\partial Loss}{\partial u^{l(r,c)}} \cdot \frac{\partial u^{l(r,c)}}{\partial F^{l(r,c)}} = \delta_u^{l(r,c)} \otimes u^{l(r-1,c)}. \quad (16)$$

(VI) The back-propagation error of the quasi-unit state is

$$\delta_G^{l(r,c)} = \frac{\partial \text{ Loss}}{\partial u^{l(r,c)}} \frac{\partial u^{l(r,c)}}{\partial G^{l(r,c)}} = \delta_u^{l(r,c).} \quad (17)$$

(VII) The back-propagation error of the unit state at the previous time step is

$$\delta_u^{l(r-1,c)} = \frac{\partial Loss}{\partial u^{l(r,c)}} \cdot \frac{\partial u^{l(r,c)}}{\partial u^{l(r-1,c)}} = \delta_u^{l(r,c)} \otimes F^{l(r,c).} \quad (18)$$

(VIII) The back-propagation error of the four components of the original output can be expressed by the above calculation results, namely:

$$\begin{cases} \delta_{z_F}^{l(r,c)} = \frac{\partial Loss}{\partial F^{l(r,c)}} \frac{\partial F^{l(r,c)}}{\partial z_F} = \delta_F^{l(r,c)} \otimes \sigma'(z_F) \\ \delta_{z_I}^{l(r,c)} = \frac{\partial Loss}{\partial I^{l(r,c)}} \frac{\partial I^{l(r,c)}}{\partial z_I} = \delta_I^{l(r,c)} \otimes \sigma'(z_I) \\ \delta_{z_G}^{l(r,c)} = \frac{\partial Loss}{\partial G^{l(r,c)}} \frac{\partial G^{l(r,c)}}{\partial z_G} = \delta_G^{l(r,c)} \otimes \tanh'(z_G) \\ \delta_{z_O}^{l(r,c)} = \frac{\partial Loss}{\partial O^{l(r,c)}} \cdot \frac{\partial O^{l(r,c)}}{\partial z_O} = \delta_O^{l(r,c)} \otimes \sigma'(z_O) \end{cases}. \quad (19)$$

Then, the back-propagation error of the original output is obtained through the combination of the reverse error of its four components, namely:

$$\delta_z^{l(r,c)} = \left[\delta_{z_F}^{l(r,c)}, \delta_{z_I}^{l(r,c)}, \delta_{z_G}^{l(r,c)}, \delta_{z_O}^{l(r,c)}\right]. \quad (20)$$

(IX) The model parameter gradient can be expressed by the equation with the original output reverse error, namely:

$$\begin{cases} \nabla_{w_{hx}} = \frac{\partial \text{ Loss}}{\partial z^{l(r,c)}} \cdot \frac{\partial z^{l(r,c)}}{\partial w_{hx}} = \delta_z^{l(r,c)} \cdot h^{l-1(r,c)} \\ \nabla_{w_{hh}} = \frac{\partial \text{ Loss}}{\partial z^{l(r,c)}} \cdot \frac{\partial z^{l(r,c)}}{\partial w_{hh}} = \delta_z^{l(r,c)} \cdot h^{l(r-1,c)} \\ \nabla_b = \frac{\partial Loss}{\partial z^{l(r,c)}} \frac{\partial z^{l(r,c)}}{\partial b} = \delta_z^{l(r,c)} \end{cases}. \quad (21)$$

3.2.2. E-Commerce Enterprise Demand Forecasting Model Based on HF-ConvLSTM

In a real Horizontal Federated Learning environment, in addition to designing the machine learning model described in Section 3.2.1, we also need to design an optimization algorithm to optimize the solution process of the Federated Learning system. Mcmahan et al. [8] first adopted the federated average algorithm (FedAvg) in the Federated Learning optimization problem. This algorithm can be used for a non-convex objective function in deep neural networks (DNN). It is based on the stochastic gradient descent algorithm SGD Evolved. This method first initializes the weights of the neural network model on a trusted

third party (central server) and then assigns the weights to the participants to train the local model. After a certain number of iterations (also called global pooling), it stops.

Generally, in a Horizontal Federated Learning system, suppose that there are K participants participating in the training of the demand prediction model, the total sample set of all participants is D, the sample size is $|\mathrm{D}|$, and the data set owned by the k^{th} participant is D_k, with sample size $|D_k|$. The objective function to be optimized is $\min_{w \in R^d} f(w)$, where

$$f(w) = \frac{1}{|D|} \sum_{i=1}^{|D|} f_i(w), \tag{22}$$

with $f_i(w) = \mathrm{Loss}(x_i, y_i; w)$ being the loss result obtained by predicting the individual cases (x_i, y_i) by the model parameter w; and x_i and y_i, respectively, representing the i^{th} data point of the k^{th} participant and its label. For the k^{th} participant,

$$F_k(w) = \frac{1}{|D_k|} \sum_{i=1}^{|D_k|} f_i(w). \tag{23}$$

Then, the overall loss function of the Horizontal Federated Learning model is

$$f(w) = \sum_{k=1}^{K} \frac{|D_k|}{|D|} F_k(w). \tag{24}$$

The gradient of the k^{th} participant is $g_k = \nabla F_k(w_t)$, the learning rate is η, and the new parameter obtained in the t^{th} iteration is

$$w^{t+1} \leftarrow w^t - \eta \sum_{k=1}^{K} \frac{|D_k|}{|D|} g_k. \tag{25}$$

The local update of each participant k is

$$w_k^{t+1} \leftarrow w_k^t - \eta \nabla F_k\left(w^k\right). \tag{26}$$

Then, combining this with the ConvLSTM-based e-commerce enterprise demand prediction model proposed in Section 3.2.1, we can get the algorithm pseudo-code of the HF-ConvLSTM-based e-commerce enterprise demand prediction model, as shown in Algorithm 1, where w_{hx_k}, w_{hh_k}, and b_k all represent the model parameters of the k^{th} client; and w_{hx}, w_{hh}, and b represent the parameters of the global model.

In Algorithm 1, we show the backpropagation algorithm based on the federated average ConvLSTM model. The algorithm is divided into two parts: Server and client. The server part is shown in lines 1–13. Specifically, in line 2, we initialize the global model w_{hx}^0, w_{hh}^0, b^0, and broadcast it to all participants participating in the horizontal federated demand prediction model. For the current iteration $t(t = 1, 2, ...T)$, in lines 4–7, for each participant k, we calculate the parameter and update it, according to the code of the client part, and return the encrypted intermediate result to the server. In line 8, we aggregate the intermediate results from different participants to obtain the global model for the next round. In lines 9–11, we set the termination condition of the algorithm iteration; that is, either the loss function converges, or the maximum iteration round is reached. In line 12, we send the aggregated model parameters to all participants.

In lines 14–31, we detail the operations performed by the client. Specifically, in line 15, we receive the encryption model parameters from the server and perform decryption operations to obtain the decrypted intermediate results w_{hx}^t, w_{hh}^t, b^t and then set them to the initial values of the operation performed by the client in line 16. In each local iteration, from 1 to the number of iterations S, we randomly divide the local data set D_k into batches $\frac{|D_k|}{\text{batches}}$, where batches denotes the number of batches. In lines 18–27, we elaborate the

operations performed on each batch. After calculation, we get $w_{hx\_k}^{\text{batches,s}}$, $w_{hh\_k}^{\text{batches,s}}$, $b_k^{\text{batches,s}}$. In line 29, we use the above results, and again use the stochastic gradient descent method to get $w_{hx\_k}^{\text{batches, }S}$, $w_{hh\_k}^{\text{batches, }S}$, $b_k^{\text{batches, }S}$. In line 30, we set the latest intermediate result as the updated parameters of the client's local model for the input of the next iteration. In line 31, we perform an additive homomorphic encryption operation on the locally updated parameters, and we then send it and the related loss function to the server.

Algorithm 1 Back-Propagation of ConvLSTM based on Federated Averaging.

Input: $x_i^{(r,c)}; y_i^{(r)}; |D_k|; i = 1, 2, \ldots, |D|; r = 1, 2, \ldots, R; c = 1, 2, \ldots, C; k = 1, 2, \ldots, K$.
Output: w_{hx}, w_{hh}, b.

1: **Server:**
2: Initialize model parameters $(w_{hx}^0, w_{hh}^0, b^0)$ and broadcast the original model parameters to all clients.
3: **for** each $t \in$ range (T) **do**
4: **for** k clients perform parallel computing, $k = 1, 2, \ldots, K$ **do**
5: Update model parameters locally: $\left[\left[w_{hx\_k}^{t+1}\right]\right]$, $\left[\left[w_{hh\_k}^{t+1}\right]\right]$, $\left[\left[b_k^{t+1}\right]\right]$ (see client execution steps)
6: Send the updated model parameters and related loss function Loss_k^{t+1} to the server.
7: **end for**
8: Aggregate the received model parameters: $\left[\left[w_{hx}^{t+1}\right]\right]$, $\left[\left[w_{hh}^{t+1}\right]\right]$, $\left[\left[b^{t+1}\right]\right]$.
9: **if** the loss function converges or reaches the maximum iteration round **then**
10: break
11: **end if**
12: Send the aggregated model parameter $\left[\left[w_{hx}^{t+1}\right]\right]$, $\left[\left[w_{hh}^{t+1}\right]\right]$, $\left[\left[b^{t+1}\right]\right]$ to all clients.
13: **end for**
14: **Client k:**
15: Get encryption parameters $\left[\left[w_{hx}^t\right]\right]$, $\left[\left[w_{hh}^t\right]\right]$, $\left[\left[b^t\right]\right]$ from server, and decrypt them to obtain w_{hx}^t, w_{hh}^t, b^t.
16: Let $w_{hx\_k}^{1,1} = w_{hx}^t, w_{hh\_k}^{1,1} = w_{hh}^t, b_k^{1,1} = b^t$.
17: **for** each $s \in$ range (S) **do**
18: **for** each $batch \in$ range $(batches)$ **do**
19: **for** each $l \in$ range (L) **do**
20: **for** each $r \in$ range (R) **do**
21: **for** each $c \in$ range (C) **do**
22: Use Equations (7–11) to obtain $z^{l(r,c)}, z_F, z_I, z_G, z_O, \delta_z^{l(r,c)}$.
23: Use Equations (20–21) to obtain $\nabla_{w_{hx\_k}}^{\text{batch }+1,s}$, $\nabla_{w_{hh\_k}}^{\text{batch }+1,s}$, $\nabla_{b_k}^{\text{batch }+1,s}$.
24: **end for**
25: **end for**
26: **end for**
27: **end for**
28: **end for**
29: Use SGD to obtain $w_{hx\_k}^{\text{batches,s+1}}$, $w_{hh\_k}^{\text{batches,s+1}}$, $b_k^{\text{batches,s+1}}$.
30: Get local model parameter update $w_{hx\_k}^{t+1} = w_{hx\_k}^{\text{batches, }S}, w_{hh\_k}^{t+1} = w_{hh\_k}^{\text{batches, }S}, b_k^{t+1} = b_k^{\text{batches, }S}$.
31: Perform additive homomorphic encryption on the updated model parameter $w_{hx_k}^{t+1}, w_{hh_k}^{t+1}, b_k^{t+1}$ to obtain $\left[\left[w_{hx_k}^{t+1}\right]\right]$, $\left[\left[w_{hh_k}^{t+1}\right]\right]$, $\left[\left[b_k^{t+1}\right]\right]$, and we then send the updated model parameters and related loss function Loss_k^{t+1} to the server.

3.3. Experimental Design and Analysis

To evaluate the feasibility and effectiveness of the demand forecasting algorithm (HF-ConvLSTM) for e-commerce enterprises based on Horizontal Federated Learning,

which we proposed from the perspective of sustainable development, we carried out a large number of simulation experiments using Python (with the PyTorch and PySyft packages), in order to compare the forecast error rate of HF-ConvLSTM with those of two classic time-series-based demand forecasting models under the same public data set and experimental environment settings. The experiment was run on a Windows 10 system server equipped with 128 GB main memory, 4 Intel Xeon E5-2690 v3 @ 2.60GHz(X2) CPUs with 12 cores and 24 threads, and an Nvidia GeForce GTX1060 6 GB graphics card.

3.3.1. Data Set

We took the Jingdong commodity sales data set as the research object, in order to demonstrate our experimental process. This data set comes from the JDATA platform GOC storage network intelligent inventory management preliminary contest. To protect data privacy, all data have been desensitized. The data set used in this paper is only used for academic research. The data set includes sales data of 1,000 products from 1 January 2016 to 31 December 2017 (as the sales activities during the large-scale promotion period were quite different from usual, the sales data in June and November were excluded). It consists of the following four tables: A basic commodity information table (sku_{info}.csv), with a total of 1000 samples; a commodity attribute information table (sku_{attr}.csv), with a total of 6776 samples (to make it easier to understand the meaning of commodity attributes, a computer attribute table is shown in Figure 7); a sales information table (sku_{sales}.csv), which records the daily sales data of the goods, having a total of 2,127,485 records, among which the value range of the sales discount attribute discount was (0, 10], with 10 representing no discount, and lower values representing higher discounts; and a promotion information table (sku_{prom}.csv), with a total of 862,111 records, in which the promotion form code $promotion_{type}$ attribute represents a variety of promotion methods, such as full discounts, vouchers, direct discounts, and so on. For convenience, we summarize the column names and meanings of the four tables in Table 2.

Table 2. Original data attribute and meaning.

Symbol	Meaning
$Column_{name}$	Description
$item_{skuid}$	SKU unique identification code
$item_{firstcateid}$	First-level category
$item_{secondcateid}$	Second-level category
$item_{thirdcateid}$	Three-level category
$brand_{code}$	Brand code
$Attr_{cd}$	Attribute code
$Attr_{valuecd}$	Attribute value
Dc_{id}	Warehouse code
Date	Date
$quantity$	Sales
$vendibility$	Inventory status at the end of the day
$Original_{price}$	Price of the day (0–1)
$discount$	Sales discount (0–10)
$Promotion_{type}$	Promotion form code

First of all, we drew a line chart for the total monthly sales volume of the 1000 products over two years, according in Figure 7. We can see that the overall sales volume of the products showed an upward trend overall, without significant or cyclical fluctuations. There were several obvious peaks and valleys in the process of product sales changes. Most of these peaks occurred during promotional periods, indicating that the sales promotion activities of e-commerce companies have huge effects on sales. In addition, it can be seen that there were significant decreases in product sales before and after festivals with promotional activities, which is in line with the fact that consumer demand ofen decreases after promotional periods.

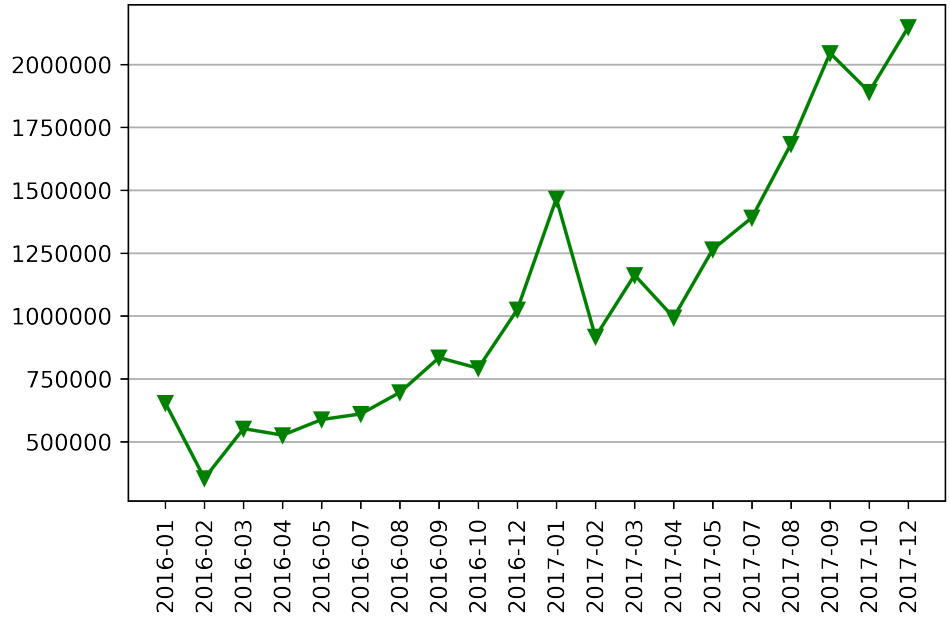

Figure 7. Linechart of total sales of 1000 products over two years.

It can be seen that dates, promotional forms, sales discounts, and prices were all factors affecting the forecasting of commodity demand. The Jingdong experimental data sets used in this paper all contain the above attribute information and, thus, can be used as a basis for commodity demand forecasting.

3.3.2. Data Pre-Processing and Data Set Segmentation

Data pre-processing was a key step in our forecasting process. The quality of the processed data directly affects the modeling effect. Taking into account the characteristics of product sales data of e-commerce companies, we carried out missing point and outlier processing, normalization, and data type conversion on the JD product sales data.

(I) Missing value processing

To deal with missing values for some incomplete or wrong data in the original data set, due to collection errors of the e-commerce companies, we used python to perform visual analysis. Figure 8 shows that, in the sales information table, only the day's price $original_{price}$ and sales discount had some data missing. As can be seen from Figures 9–11, there were no missing data. As Figure 12 shows, taking sales discount as an example, we found that 42.78% of the data had no sales discount information. Some of the sales discount information was missing due to the collection error of the e-commerce company, indicating that the sales discount information of this part of the product had no major impact on sales and, so, it can be considered that this part of the product has no discount. Thus, we filled this type of missing data with a value of 10 (the range of the sales discount was 0–10; the lower the value, the bigger the discount, and 10 means no discount).

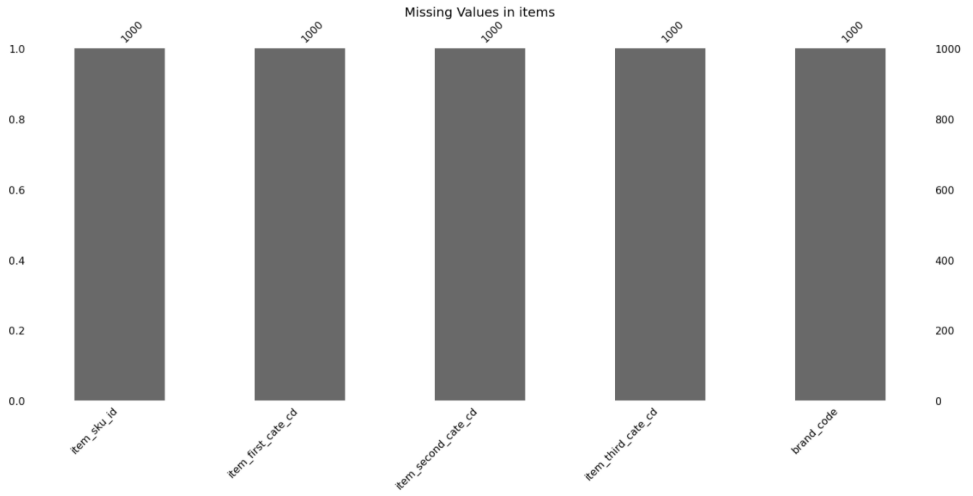

Figure 8. Missing values in the basic commodity information table.

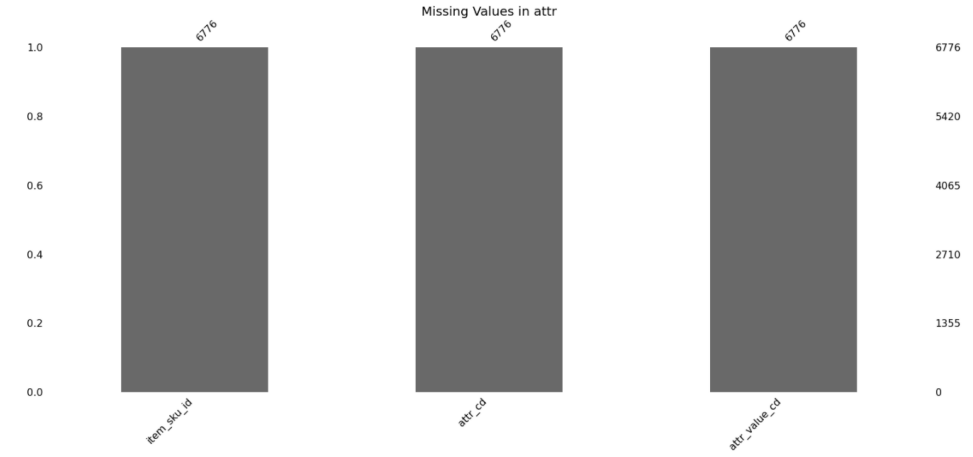

Figure 9. Missing values in the product attribute information table.

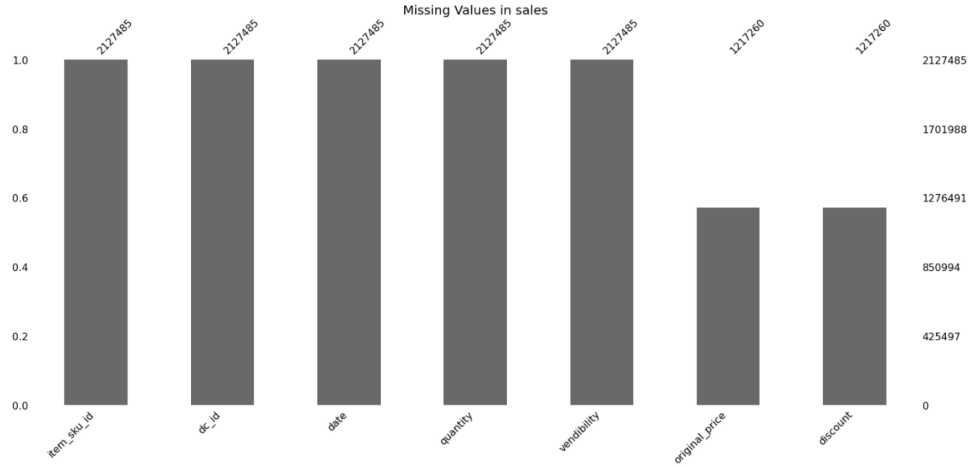

Figure 10. Missing values in the sales information table.

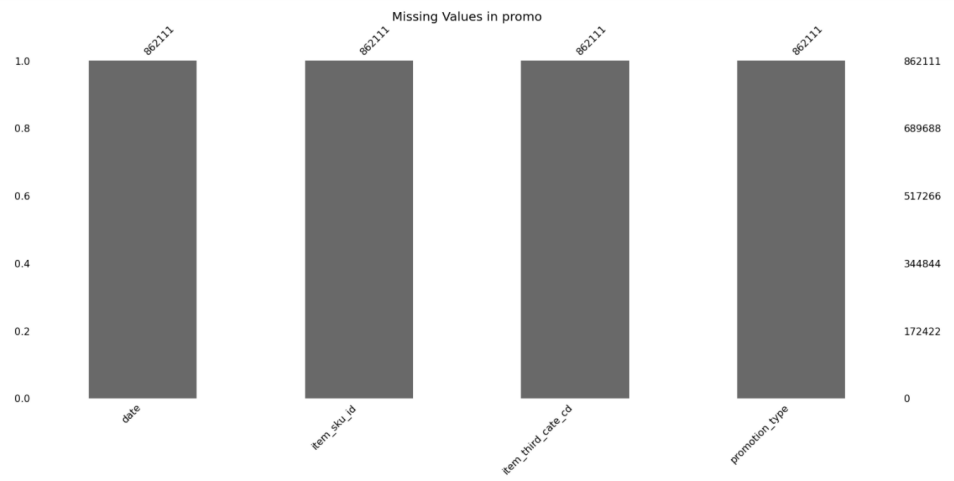

Figure 11. Missing values in the promotion information table.

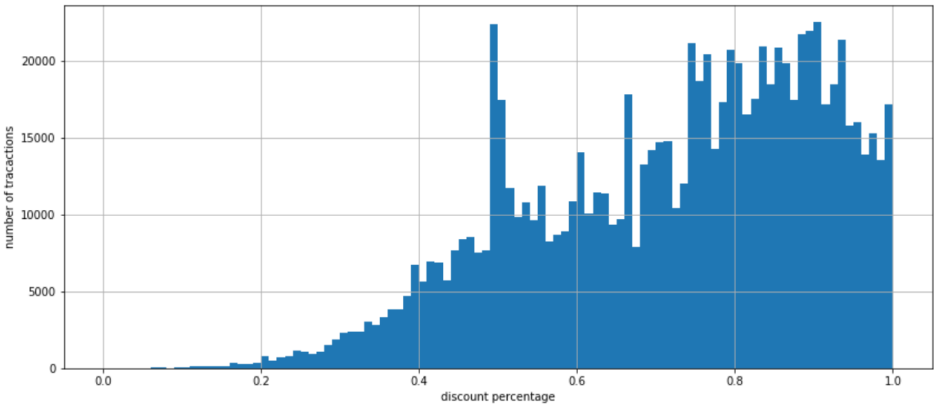

Figure 12. Missing sales discount data.

(II) Duplicate value processing

Duplicate values refer to data with all the same attribute values. In the JD product sales data set, one SKU corresponded to one record at the same point in time, so that there were no duplicates, and no processing were required.

(III) Outlier handling

Outliers are sample values that are significantly different from others. In the JD promotion information table, if the SKU unique identification code $item_{sku\ id}$ was -999, the data was considered to have an abnormal value; however, no abnormalities were found in other data. There were only 108 outliers in the $item_{sku\ id}$ attribute in this data set. Therefore, these outliers were directly deleted.

(IV) Data conversion

Generally, the different measurement units of each attribute bring about the problem of inconsistent data dimension; so, we normalized the data. First, the variables were divided into discrete and continuous types and then processed using different normalization methods.

We performed a 0–1 standardization operation for discrete attributes, such as promotion type, and changed the original data through a linear transformation, such that the final value fell into the interval $[0, 1]$. The transformation rule is shown in Equation (27):

$$x^* = \frac{x - \min x}{\max x - \min x}. \tag{27}$$

The z-score normalization operation was adopted for continuous attribute values, such as end-of-day inventory. After the operation, it follows the standard normal distribution $N(0,1)$, and its transformation rule is shown in Equation (28):

$$x^* = \frac{x - \mu}{\sigma}. \tag{28}$$

For the date (*Date*), we converted it to the Datetime data type, following %Y-%m-%d.

(V) Data connection

To facilitate training, before the demand prediction model training, we needed to connect the data in the four tables and store them in a new data summary table. According to the actual situation of this paper, we deleted the first-level category, second-level category, and other attributes in the spliced data summary table, and only retained key information related to sales. The total number of processed data was 2,127,485, and the field names are shown in Table 3 below.

Table 3. Header information of data set after data connection.

$item_{sku\ id}$	Dc_{id}	Date	vendibility	$Original_{price}$	discount	$Attr_{cd}$	$Attr_{value\ cd}$

(VI) Data set segmentation

To simulate the scenario of multi-party Horizontal Federated Learning, we divided the data set processed by the above steps into six parts, according to the warehouse code (Dc_{id}), and numbered them from 1 to 6; that is, using the participant index k ($k = 1, 2, \ldots, 6$). The data distribution of each participant, after the data set segmentation, is shown in Table 4.

Table 4. Basic information of each participant after data set segmentation.

Participants (k)	**1**	**2**	**3**	**4**	**5**	**6**
Dc_{id}	0	1	2	3	4	5
The amount of data	433,315	424,875	423,169	416,310	279,957	149,859
Number of types of Sku_{id}	1000	998	996	996	994	989

3.3.3. Comparison of Algorithms

To examine the feasibility and effectiveness of the proposed e-commerce enterprise demand forecasting method (HF-ConvLSTM) based on Horizontal Federated Learning, we compare it with classic demand forecasting algorithms based on time-series: LSTM [69,70] and BiLSTM [69].

In the first step, we separately compared the performance of ConvLSTM with LSTM and BiLSTM.

LSTM. LSTM is a variant of the RNN model. For sequences with long-term dependence, it is difficult to train the early RNN model, especially when the error propagates back along multiple time steps, it is easy for the gradient to disappear. LSTM introduces a logic gate unit with memory function to solve the training difficulties encountered in early RNN [69]. For an RNN variant, such as LSTM, which can obtain the long-term dependence of sequence data, each memory unit in its structure is differentiable. Therefore, it is ensured that the chain rule can be used for derivation and error back-propagation based on the time step, such that the learning performance on deep time-series is better than that of other RNN variants [70].

BiLSTM. On the basis of LSTM, bidirectional LSTM uses past and future timing information (dimension data of the previous time step and the next time step) to determine the node output at the current time step position in the sequence in two directions. In other words, for the node information of any dimension in a sequence, the node information in the two time steps before and after it helps to predict the node information [69].

ConvLSTM. This model was proposed in Section 3.2.1, and it was theoretically analyzed and explained.

In the second step, we compared the performance of the above three algorithms in the Horizontal Federated Learning framework; that is, we compared the performance of HF-ConvLSTM with HF-LSTM and HF-BiLSTM. Due to limited space, we will not elaborate on LSTM and BiLSTM.

3.3.4. Evaluation Index

We used the average absolute error (MAE) and average absolute percentage error (MAPE) to evaluate the performance of the demand forecasting algorithm. MAE is a basic evaluation method in regression tasks, which represents the absolute error between the true value and the predicted value, and which can be used to compare the accuracy of different prediction models. MAPE represents the ratio of the absolute error to the true value. It can compare the stability of different prediction models. The calculation formulas are as follows:

$$\mathrm{MAE}(y, f) = \frac{1}{|D_k|_{test}} \sum_{i=1}^{|D_k|_{test}} |y_i - f_i|, \tag{29}$$

$$\mathrm{MAPE}(y, f) = \frac{1}{|D_k|_{\mathrm{test}}} \sum_{i=1}^{|D_k|_{test}} \frac{(y_i - f_i)^2}{y_i}, \tag{30}$$

where y_i represents the true value, f_i represents the predicted value, and $\left|D_{k\_test}\right|$ is the size of the test data set of the k^{th} participant.

In addition, to evaluate the mitigation ability of each model to the bullwhip effect (BWE), we used the ratio of the predicted value variance to the true value variance to quantify the bullwhip effect. The specific formula is defined as follows:

$$\mathrm{BWE}(y, f) = \frac{\mathrm{var}(f)}{\mathrm{var}(y)}, \tag{31}$$

where $\mathrm{var}(f)$ and $\mathrm{var}(y)$, respectively, represent the variance of the predicted value and the true value, with calculation formulae as follows:

$$\mathrm{var}(f) = \frac{1}{\left|D_{k\_test}\right|} \sum_{i=1}^{\left|D_{k\_test}\right|} \left(f_i - \bar{f}\right)^2, \tag{32}$$

$$\mathrm{var}(y) = \frac{1}{\left|D_{k\_test}\right|} \sum_{i=1}^{\left|D_{k\_test}\right|} \left(y_i - \bar{y}\right)^2, \tag{33}$$

where $\bar{f} = \frac{1}{\left|D_{k\_test}\right|} \sum_{i=1}^{\left|D_{k\_test}\right|} f_i$ and $\bar{y} = \frac{1}{\left|D_{k\_test}\right|} \sum_{i=1}^{\left|D_{k\_test}\right|} y_i$ represent the sample means of the predicted data and the real data, respectively.

In summary, Equation (31) vividly describes the uncertainty and severity of the bullwhip effect. It is difficult to completely eliminate the bullwhip effect in practice. Only when BWE = 1, is there no distortion in the transmission of demand information. The greater the distance between BWE and 1, the more obvious the bullwhip effect will be, which means that there may be extremely unreasonable inventory waste in the operation process of e-commerce enterprises, which directly restricts the sustainable development of e-commerce enterprises. In other words, the bullwhip effect value represents the sustainable development ability of e-commerce enterprises, to a certain extent.

3.3.5. Experimental Setup

This part briefly introduces the relevant parameter settings for each comparative experiment.

(I) LSTM

The LSTM layer was composed of a fully connected layer and five hidden layers containing logic units. First, feature extraction is performed through the fully connected layer, and the SGD stochastic gradient descent optimizer is used to iterate the model parameters during the backward error propagation between the hidden layers. In the gradient descent process, the learning rate is set to 0.001, the activation function is the sigmoid function, and the tanh function is set by the internal logic unit of the LSTM. In addition, to prevent overfitting, a random dropout rate of 0.1 is set, the training batch size is set to 512, and the number of iterations is 500 rounds [71].

(II) BiLSTM

The BiLSTM model is a combination of forward and backward LSTM models. By setting a two-way time step, the forward sequence is compared with the reverse sequence, in order to better learn the temporal characteristics. Other parts, such as the optimizer, learning rate, and other structure settings, were the same as LSTM. The training batch size is set to 512, and the number of iterations is 500 rounds.

(III) ConvLSTM

On the basis of the LSTM model, the convolutional layer is introduced to extract the local features of the sample. Unlike the fully connected layer, all neurons between adjacent layers are interconnected in pairs. The convolutional layer sets a certain size and number of convolutions to characterize the sample. Sliding interception is carried out, in order to learn the local distribution information of sample features. In this experiment, 16 convolution kernels with size (5, 5) are set in the ConvLSTM model, and the input features of the sample are slidingly cut under the condition of a step size of (1, 1). Then, consistent with the LSTM model, the memory mechanism and forget mechanism are realized through logic units. The training batch size is set to 512, and the number of iterations is 500 rounds.

(IV) E-commerce enterprise demand forecasting model based on Horizontal Federated Learning

In the Horizontal Federated Learning system, the federated average algorithm is used to aggregate the prediction model training performed by the participants in the system, i.e., the LSTM prediction model based on Horizontal Federated Learning (HF-LSTM), the BiLSTM prediction model based on Horizontal Federated Learning (HF-BiLSTM), and the ConvLSTM prediction model based on Horizontal Federated Learning (HF-ConvLSTM, Algorithm 1). The structure settings in each model were consistent with the settings under the conditions of the non-Federated Learning systems. On this basis, the proportion of the data set size for each participant in the overall data set size was used as the respective weight; that is, the weights were assigned according to the ratio of 433,315:424,875:423,169:416,310:279,957:149,859. We set the training batch size to 512 and the number of iterations to 500 rounds.

3.3.6. Display and Analysis of Experimental Results

The results obtained using the processed data and the abovementioned experimental environment and parameter settings are shown in Tables 5–10. For the convenience of observation, we display these results in Figures 13–18.

Figures 13–15 show the three evaluation indicators of MAE, MAPE, and BWE, respectively, in order to examine the predictive ability of the model under the condition of non-Federated Learning system. Figures 16–18 also show the MAE, MAPE, and BWE indicators, respectively, when examining the predictive ability of the models, considering the Horizontal Federated Learning system.

The MAE is used to compare the accuracies of different models. The smaller the value, the smaller the absolute error between the true value and the predicted value, and the higher the accuracy of the model prediction. It can be seen, from Figure 13, that the MAE values of the BiLSTM and ConvLSTM models were lower than those with the rest of the participant data sets training gains. In addition, the MAE value of the ConvLSTM model was lower than that value of the BiLSTM model under the same conditions. It can be seen that, when the data size was at least 424,875 (the size of participant 2's data

set), the prediction error can be significantly reduced by adding two-way time steps or convolutional layers to the traditional LSTM model. From the MAE value of the model trained using the six participant data sets, it can be seen that the performance of the convolutional layer in reducing the prediction error was better than the two-way time step.

It can be seen, from Figure 16, that, under the conditions of the Horizontal Federated Learning system, whether the participants were paired for Federated Learning or all participants participated in Federated Learning at the same time, the prediction errors of the HF-BiLSTM model and the HF-ConvLSTM model were lower than those of the HF-LSTM model. After comparing with the experimental results in Figure 13, it was found that the MAE values of the model based on Horizontal Federated Learning conditions were all lower than the smallest value in the federated participant combination. It can be seen that the demand prediction method based on Horizontal Federated Learning can significantly reduce the prediction error of the model, compared with the prediction error of the model obtained by independent training.

The MAPE is used to compare the stability of different prediction models. The smaller the value, the smaller the proportion of the absolute error square in the true value, the better the stability, and the smaller the fluctuation. It can be seen, from Figure 14, that the MAPE values of the model trained under the data sets of participants 1, 2, 3, and 4 were all higher than those of the models trained under the data set of participant No. 5, and the MAPE value of the model trained under the data set of all six participants. The data sets of participants 1–4 were at least 50% larger than the data sets of participants 5 or 6. It can be found that, when the data set is relatively large, the model predicts that the volatility will increase. In this case, the convolutional layer or the bidirectional time step can still weaken the change, to a certain extent, where the improved ability of the convolutional layer is higher than that of the bidirectional time step.

It can be seen, from Figure 17, that, under the conditions of the Horizontal Federated Learning system, whether the participants were paired for Federated Learning or all participants participated in Federated Learning at the same time, the prediction error fluctuations of the HF-BiLSTM model and the HF-ConvLSTM model were lower than that of the HF-LSTM model. By comparison with the experimental results in Figure 14, it can be seen that the MAPE values of the models based on the Horizontal Federated Learning condition were lower than the smallest value in the combination of federated participants. Thus, the demand forecasting method based on Horizontal Federated Learning can significantly reduce the forecast error of the model.

The BWE is used to evaluate the mitigation ability of the model to the bullwhip effect, i.e., promotion of the ability for sustainable development. The smaller the distance from 1, the more the bullwhip effect has been improved and the more the sustainability of the enterprise has been improved. It can be seen, from Figure 15, that the BWE values fluctuated around 1, and the experimental result data can be obtained successfully. Among the three models of LSTM, BiLSTM, and ConvLSTM, the BWE value of the ConvLSTM model was the closest to 1. This indicates that the bullwhip effect can be effectively alleviated (and, thus, the sustainability of e-commerce enterprises improved) by introducing the convolution operation into the LSTM model.

It can be seen, from Figure 18, that, under the condition of the Horizontal Federated Learning system, whether the participants were combined in pairs for Federated Learning or all participants participated in Federated Learning at the same time, the distance between the BWE value of each model and 1 was lower than that with the federated participant combination. Therefore, the Horizontal Federated Learning method improves the mitigation ability of the model to the bullwhip effect, on the premise of ensuring data privacy, and promotes the sustainable development of e-commerce enterprises.

Table 5. The MAE value for each participant to independently perform demand forecasting learning.

Participant	LSTM	BiLSTM	ConvLSTM
1	2425.65	2088.26	2028.26
2	3213.35	2476.14	2022.91
3	3147.89	3063.81	3021.43
4	3088.95	2721.45	2519.63
5	3256.61	2880.67	2647.27
6	3101.05	2905.18	2853.63

Table 6. The MAPE value for each participant to independently perform demand forecasting learning.

Participant	LSTM	BiLSTM	ConvLSTM
1	0.2886	0.2173	0.2076
2	0.2879	0.2767	0.2476
3	0.2793	0.2749	0.2634
4	0.2682	0.2368	0.2363
5	0.1381	0.1348	0.1026
6	0.0923	0.0737	0.0558

Table 7. The BWE value for each participant to independently perform demand forecasting learning.

Participant	LSTM	BiLSTM	ConvLSTM
1	1.1391	0.8667	0.8819
2	0.7206	1.2149	0.8096
3	0.8725	1.1185	0.9058
4	0.9047	1.0947	1.0847
5	0.8672	0.8847	1.1153
6	1.1557	0.8479	1.1327

Table 8. The MAE value of the participant's federated demand forecasting learning.

Participant	HF-LSTM	HF-BiLSTM	HF-ConvLSTM
(1,2)	2264.96	2084.55	1979.45
(1,3)	2300.45	1933.24	1836.43
(1,4)	2088.05	1643.76	1271.07
(1,5)	2134.68	1766.73	1677.92
(1,6)	2127.88	1715.98	1519.06
(2,3)	3013.89	2685.15	2448.72
(2,4)	3046.40	2868.42	2613.40
(2,5)	3101.35	2369.03	1797.95
(2,6)	3087.15	2601.80	2110.68
(3,4)	3070.60	2429.39	2355.04
(3,5)	3013.65	2635.99	2382.52
(3,6)	2981.24	2520.39	2312.14
(4,5)	3038.78	2638.10	2628.71
(4,6)	3012.31	2626.32	2432.85
(5,6)	3088.55	2771.18	2554.51
(1,2,3,4,5,6)	2307.93	2176.45	1890.68

Table 9. The MAPE value of the participant's federated demand forecasting learning.

Participant	HF-LSTM	HF-BiLSTM	HF-ConvLSTM
(1,2)	0.2166	0.1992	0.1597
(1,3)	0.2708	0.2541	0.2172
(1,4)	0.2235	0.1715	0.1658
(1,5)	0.1062	0.1037	0.0833
(1,6)	0.0847	0.0788	0.0706
(2,3)	0.2784	0.2545	0.2296
(2,4)	0.2594	0.2280	0.2072
(2,5)	0.1373	0.1288	0.1062
(2,6)	0.0758	0.0720	0.0659
(3,4)	0.2489	0.1996	0.1561
(3,5)	0.1198	0.1072	0.0939
(3,6)	0.0809	0.0616	0.0472
(4,5)	0.1171	0.0963	0.0884
(4,6)	0.0876	0.0779	0.0618
(5,6)	0.0824	0.0750	0.0664
(1,2,3,4,5,6)	0.0807	0.0686	0.0655

Table 10. The BWE value of the participant's federated demand forecasting learning.

Participant	HF-LSTM	HF-BiLSTM	HF-ConvLSTM
(1,2)	1.1199	0.8801	0.8938
(1,3)	0.8967	1.0949	0.9071
(1,4)	0.9176	0.9176	1.0623
(1,5)	1.1095	1.0866	0.9335
(1,6)	1.1174	1.0881	0.9202
(2,3)	1.0957	1.0921	0.9133
(2,4)	0.9173	0.9208	1.0630
(2,5)	1.1135	0.8888	0.8983
(2,6)	0.8520	0.8872	0.9039
(3,4)	0.9226	0.9305	1.0691
(3,5)	0.8944	1.1007	1.0957
(3,6)	0.8904	0.9134	1.0663
(4,5)	1.0810	0.9335	0.9482
(4,6)	1.0836	1.0722	0.9395
(5,6)	1.1057	0.8993	1.0907
(1,2,3,4,5,6)	0.9117	0.9336	0.9439

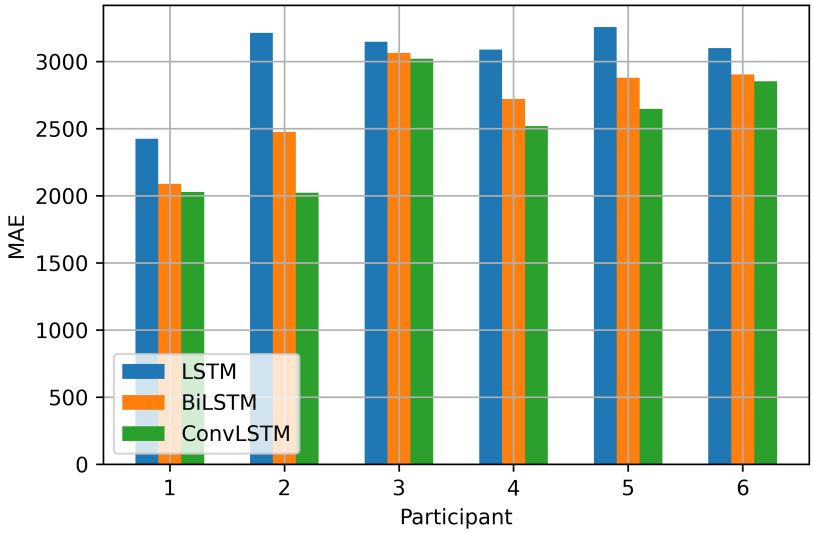

Figure 13. The MAE value for each participant to independently perform demand forecasting learning.

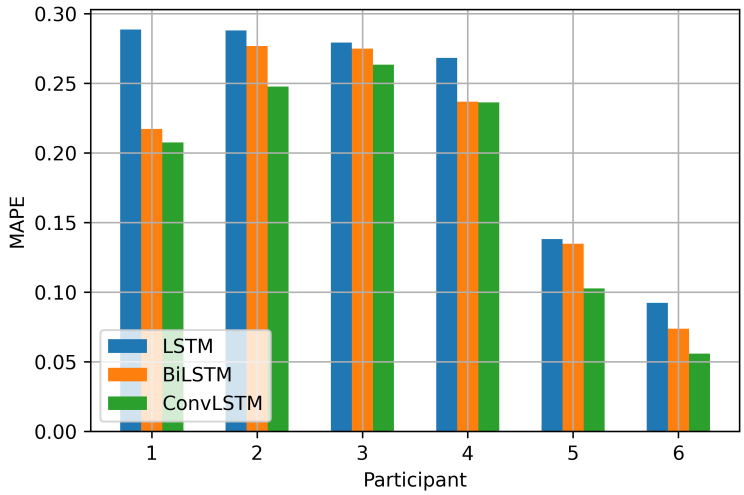

Figure 14. The MAPE value for each participant to independently perform demand forecasting learning.

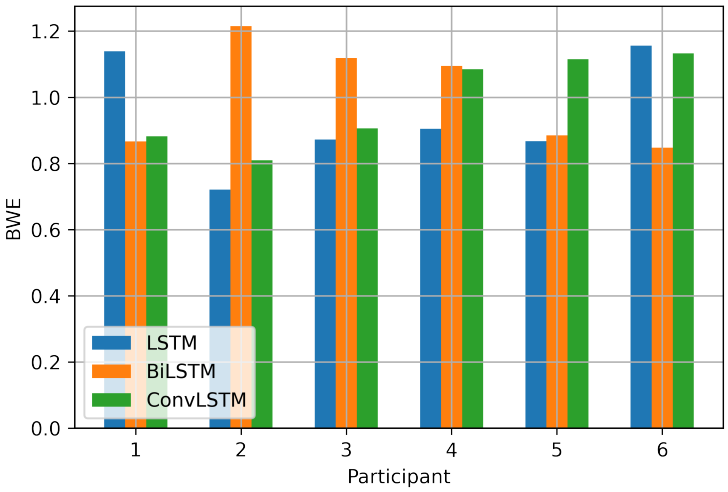

Figure 15. The BWE value for each participant to independently perform demand forecasting learning.

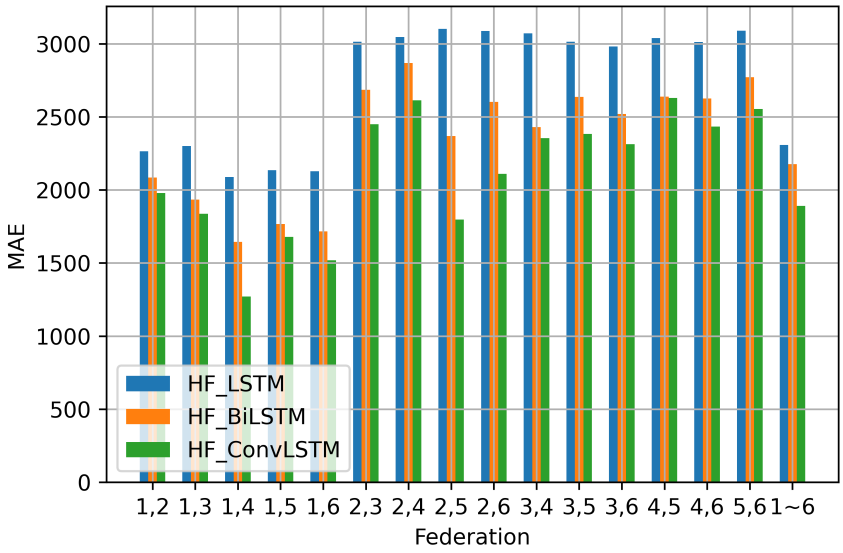

Figure 16. The MAE value of the participant's federated demand forecasting learning.

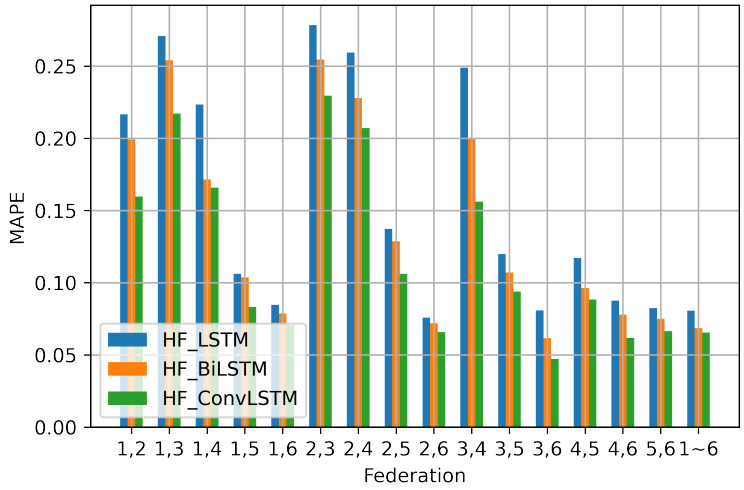

Figure 17. The MAPE value of the participant's federated demand forecasting learning.

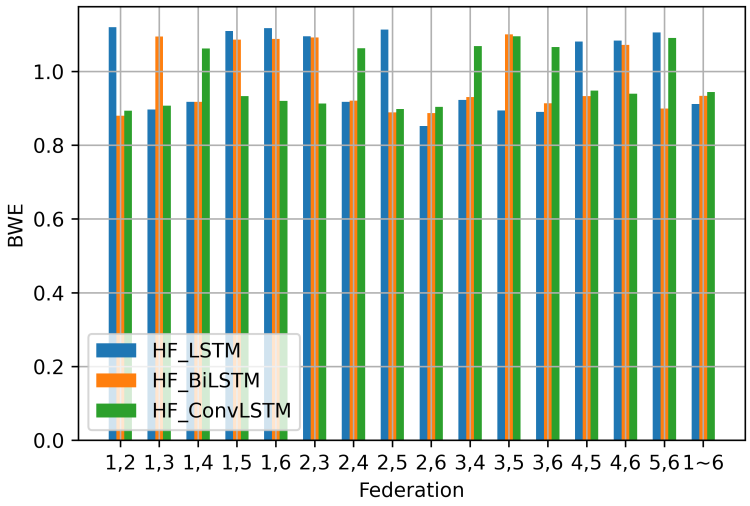

Figure 18. The BWE value of the participant's federated demand forecasting learning.

4. Conclusions

As an important strategic emerging industry, e-commerce plays an important role in the processes of social innovation and sustainable development. Under the influence of the COVID-19, the supply chain safety problem of e-commerce enterprises has received more attention. This paper proposed a demand forecasting method for e-commerce enterprises based on Horizontal Federated Learning and ConvLSTM, such that e-commerce enterprises of the same type can indirectly achieve the goal of demand information sharing modeling through Horizontal Federated Learning, under the premise that their private data is not available locally. Specifically, first, we used the convolutional neural network structure to optimize the traditional LSTM network, considering time-series data, extracting its multi-dimensional features, and proposing a demand forecasting model. Second, we used a Horizontal Federated Learning system to train the demand forecasting model with the data from multiple parties, without going out of the local situation. Finally, we used a large number of public data sets to experimentally verify the proposed model. The results showed that the demand forecast model based on Horizontal Federated Learning and ConvLSTM not only can improve the accuracy and stability of the demand forecast model but, more importantly, it avoids potential privacy leakage issues for supply chain companies when they use big data to forecast demand. The research in this paper has far-reaching

significance for the sustainable development of the supply chain and alleviation of the bullwhip effect.

However, this study also carries some limitations. When using Horizontal Federated Learning to study the demand forecast of e-commerce enterprises, due to the experimental cost and the requirements of data privacy protection regulations, this paper only considers a small number of participants. When the number of participants continues to increase, the stability of the model may decline, and the generalization ability of the model needs to be improved. Our future work will focus on optimizing parameters to improve the generalization ability of neural network models. In addition, in the whole supply chain system where e-commerce enterprises are located, the problem that different types of node enterprise demand information cannot be shared and exchanged is more prominent, and the application potential of Federated Learning is greater. In the future, we plan to change our research direction to the demand information prediction method for e-commerce enterprise supply chains based on vertical Federated Learning. On the premise that the data privacy is not local, we wish to continuously improve the accuracy of the supply chain demand prediction model for the whole e-commerce enterprise, reduce the inventory waste of enterprises at all levels, alleviate the bullwhip effect, and further promote the sustainable and healthy development of e-commerce enterprises.

Author Contributions: This paper was completed by 6 authors. Their contributions are as follows: Conceptualization, J.L. and T.C.; methodology, T.C. and K.Y.; software, J.L., M.L., and K.Y.; validation, J.L., T.C., and K.Y.; formal analysis, K.Y.; investigation, T.C.; resources, J.L. and M.L.; data curation, R.Y. and L.H.; writing—original draft preparation, J.L., T.C., and K.Y.; writing—review and editing, T.C. and K.Y.; visualization, K.Y., M.L., and L.H.; supervision, J.L. and T.C.; project administration, R.Y.; funding acquisition, R.Y. and J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This paper was funded by National Natural Science Foundation of China (72101033, 71831001), Beijing Key Laboratory of Intelligent Logistics Systems (BZ0211), Canal Plan-Youth Top-notch Talent Project of Beijing Tongzhou District (YHQN2017014).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Oniszczuk-Jastrząbek, A.; Czermański, E.; Cirella, G.T. Sustainable Supply Chain of Enterprises: Value Analysis. *Sustainability* **2020**, *12*, 419. [CrossRef]
2. Papaioannou, G.; Sarakinos, I. *The General Data Protection Regulation (GDPR, 2016/679/EE) and the (Big) Personal Data in Cultural Institutions: Thoughts on the GDPR Compliance Process*; Springer: Cham, Switzerland, 2018.
3. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and open problems in federated learning. *Found. Trends Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
4. Nitish, S.; Elman, M.; Ruslan, S. Unsupervised learning of video representations using lstms. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015; pp. 843–852.
5. Konečný, J.; McMahan, H.B.; Ramage, D.; Richtárik, P. Federated Optimization: Distributed Machine Learning for on-device Intelligence. *arXiv* **2016**, arXiv:1610.02527.
6. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol.* **2019**, *10*, 1–19. [CrossRef]
7. Reza, S.; Vitaly, S. Privacy-Preserving Deep Learning. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, Denver, CO, USA, 12–16 October 2015; pp. 1310–1321.
8. Mcmahan, H.B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
9. Kallista, A.B.; Vladimir, I.; Kreuter, B.; Marcedone, A.; Mcmahan, H.B.; Patel, S.; Ramage, D.; Aaron, S.; Karn, S. Practical secure aggregation for privacy-preserving machine learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, Dallas, TX, USA, 30 October–3 November 2017; pp. 1175–1191.
10. Yoshinori, A.; Takuya, H.; Wang, L.; Moriai, S. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Trans. Inf. Forensics Secur.* **2017**, *13*, 1333–1345.
11. Virginia, S.; Chao-Kai, C.; Maziar, S.; Ameet, S.T. Federated Multi-Task Learning. In Proceedings of the Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017; pp. 4424–4434.

12. Mcmahan, H.B.; Moore, E.; Ramage, D.; Arcas, B. Federated Learning of Deep Networks using Model. *arXiv* **2016**, arXiv:1602.05629.
13. Lin, Y.; Han, S.; Mao, H.; Wang, Y.; Dally, W.J. Deep gradient compression: Reducing the communication bandwidth for distributed training. In Proceedings of the ICLR, Vancouver, BC, Canada, 30 April–3 May 2018.
14. Zhao, B.; Liu, X.; Chen, W. When Crowdsensing Meets Federated Learning: Privacy-Preserving Mobile Crowdsensing System. *arXiv* **2021**, arXiv:2102.10109.
15. Mohassel, P.; Zhang, Y. SecureML: A System for Scalable Privacy-Preserving Machine Learning. *IEEE Symp. Secur. Privacy* **2017**, 19–38.
16. Gao, D.; Liu, Y.; Huang, A.; Ju, C.; Yu, H.; Yang, Q. Privacy-preserving Heterogeneous Federated Transfer Learning. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 2552–2559.
17. Zhao, Y.; Zhao, J.; Jiang, L.; Tan, R.; Niyato, D.; Li, Z.; Liu, Y. Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices. *IEEE Internet Things J.* **2021**, *8*, 1817–1829. [CrossRef]
18. Cong, M.; Yu, H.; Weng, X.; Yiu, S.M. A Game-Theoretic Framework for Incentive Mechanism Design in Federated Learning. In *Federated Learning*; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12500, pp. 205–222.
19. Wei, S.; Tong, Y.; Zhou, Z.; Song, T. Efficient and Fair Data Valuation for Horizontal Federated Learning. In *Federated Learning*; Springer: Berlin/Heidelberg, Germany, 2020; Volume 12500, pp. 139–152.
20. Kim, S. Incentive Design and Differential Privacy Based Federated Learning: A Mechanism Design Perspective. *IEEE Access* **2020**, *8*, 187317–187325. [CrossRef]
21. Zhan, Y.; Zhang, J.; Hong, Z.; Wu, L.; Guo, S. A Survey of Incentive Mechanism Design for Federated Learning. *IEEE Trans. Emerg. Top. Comput.* **2021**, *99*, 1. [CrossRef]
22. Alberternst, S.; Anisimov, A.; Andre, A.; Benjamin, D.; Hilko, H.; Michael, M.; Muhammad, M.; Daniel, S.; Ingo, Z. Orchestrating Heterogeneous Devices and AI Services as Virtual Sensors for Secure Cloud-Based IoT Applications. *Sensors* **2021**, *21*, 7509 [CrossRef]
23. Huang, W.; Yang, Y.; Chen, M.; Liu, C.; Feng, C.; Vincent, H.P. Wireless Network Optimization for Federated Learning with Model Compression in Hybrid VLC/RF Systems. *Entropy* **2021**, *23*, 1413. [CrossRef]
24. Vasiliki, K.; Vasileios, A.; Thomas, L.; George, F.; Elisavet, G.; Panagiotis, S. IDS for Industrial Applications: A Federated Learning Approach with Active Personalization. *Sensors* **2021**, *21*, 6743.
25. Venkataramanan, K.; Kaza, S.; Annaswamy, A.M. DER Forecast using Privacy Preserving Federated Learning. *arXiv* **2021**, arXiv:2107.03248.
26. Romano, F.; Benedetta, P. Federated learning framework for mobile edge computing network. *Trans. Intell. Technol.* **2020**, *5*, 15–21.
27. Alexander, L.B.; Michael, P.P.; Nicholas, J.W. Domain Adaptation and Federated Learning for Ultrasonic Monitoring of Beer Fermentation. *Fermentation* **2021**, *7*, 253.
28. Li, S.; Lv, L.; Li, X.; Ding, Z. Mobile App Start-Up Prediction Based on Federated Learning and Attributed Heterogeneous Network Embedding. *Future Internet.* **2021**, *13*, 256. [CrossRef]
29. Eoin, B.; Maarten, D.V.; Geraldine, B.B.; Toms, W. Estimation of Continuous Blood Pressure from PPG via a Federated Learning Approach. *Sensors* **2021**, *21*, 6311.
30. Moustris, K.P.; Nastos, P.T.; Larissi, I.K. Application of Multiple Linear Regression Models and Artificial Neural Networks on the Surface Ozone Forecast in the Greater Athens Area, Greece. *Adv. Meteorol.* **2012**, *2012*, 978–988. [CrossRef]
31. Fang, Y.; Wang, X.; Yan, J. Green Product Pricing and Order Strategies in a Supply Chain under Demand Forecasting. *Sustainability* **2020**, *12*, 713. [CrossRef]
32. Gardner, J.E.S. Exponential smoothing: The state of the art-Part II. *Int. J. Forecast.* **2006**, *22*, 637–666. [CrossRef]
33. Li, J.; Chen, W. Forecasting Macroeconomics Time Series: LASSO-based Approaches and Their Forecast Combinations with Dynamic Factor Models. *Int. J. Forecast.* **2014**, *30*, 996–1015. [CrossRef]
34. Firmino, P.; Neto, P.; Ferreira, T. Correcting and combining time series forecasters. *Neural Netw.* **2014**, *50*, 1–11. [CrossRef] [PubMed]
35. Büyükşahin, Ü.Ç.; Şeyda, E. Improving forecasting accuracy of time series data using a new ARIMA-ANN hybrid method and empirical mode decomposition. *Neurocomputing* **2019**, *361*, 151–163. [CrossRef]
36. Novri, S.; Suhartono; Dedy, D.P.; Baharuddin, A. Roll motion prediction using a hybrid deep learning and ARIMA model. *Procedia Comput. Sci.* **2018**, *144*, 251–258.
37. Huang, Z.; Lei, D.; Han, Z.; Zhang, P. Boundary Moving Least Square method for numerical evaluation of two-dimensional elastic membrane and plate dynamics problems. *Eng. Anal. Bound. Elem.* **2019**, *108*, 41–48. [CrossRef]
38. Luo, X.M.; Li, J.B.; Peng, H.U.; Management, S.O. E-commerce Inventory Optimization Strategy Based on Time Series Forecasting. *Syst. Eng.* **2014**, *32*, 91–98.
39. Lu, W.; Chen, X.; Pedrycz, W.; Liu, X.; Yang, J. Using interval information granules to improve forecasting in fuzzy time series. *Int. J. Approx. Reason.* **2015**, *57*, 1–258. [CrossRef]
40. Kittichotsatsawat, Y.; Jangkrajarng, V.; Tippayawong, K.Y. Enhancing Coffee Supply Chain towards Sustainable Growth with Big Data and Modern Agricultural Technologies. *Sustainability* **2021**, *13*, 4593. [CrossRef]
41. Choi, A.; Wang, R.; Darwiche, A. On the relative expressiveness of Bayesian and neural networks. *Int. J. Approx. Reason.* **2019**, *113*, 303–323. [CrossRef]

42. Zhang, G.; Li, G.; Peng, J. Risk Assessment and Monitoring of Green Logistics for Fresh Produce Based on a Support Vector Machine. *Sustainability* **2020**, *12*, 7569. [CrossRef]
43. Zhang, G. Study of Logistics Demand Forecast based on Grey-Markov Model. *Math. Pract. Theory* **2011**, *41*, 17–21.
44. Wang, Z.X.; Li, Q.; Pei, L.L. A seasonal GM(1,1) model for forecasting the electricity consumption of the primary economic sectors. *Energy.* **2018**, *154*, 533–534. [CrossRef]
45. Hu, Y.C. A genetic-algorithm-based remnant grey prediction model for energy demand forecasting. *PLoS ONE* **2017**, *12*, e0185478.
46. Liu, Y.; Ju, W.; Zhao, J.; Gao, J.; Zheng, J.; Jiang, A. Product life cycle based demand forecasting by using artificial bee colony algorithm optimized two-stage polynomial fitting. *J. Intell. Fuzzy Syst.* **2016**, *31*, 825–836.
47. Adamowski, J.F. Peak Daily Water Demand Forecast Modeling Using Artificial Neural Networks. *J. Water Resour. Plan. Manag.* **2008**, *134*, 119–128. [CrossRef]
48. Wang, G. Research On Supply Chain Demand Prediction Based On BP Neural Network Algorithm. *Inmateh-Agric. Eng.* **2013**, *40*, 27–34.
49. Noori, R. Uncertainty analysis of support vector machine for online prediction of five-day biochemical oxygen demand. *J. Hydrol.* **2015**, *527*, 833–843. [CrossRef]
50. Cao, J.; Jiang, Z.; Wangk, K. Customer demand prediction of service-oriented manufacturing using the least square support vector machine optimized by ppaper swarm optimization algorithm. *Eng. Optim.* **2017**, *49*, 1197–1210. [CrossRef]
51. Chalmeta, R.; Barqueros-Muoz, J.E. Using Big Data for Sustainability in Supply Chain Management. *Sustainability* **2021**, *13*, 7004. [CrossRef]
52. Guo, F.; Diao, J.; Zhao, Q.; Wang, D.; Sun, Q. A Double-level Combination Approach for Demand Forecasting of Repairable Airplane Spare Parts Based on Turnover Data. *Comput. Ind. Eng.* **2021**, *110*, 92–108. [CrossRef]
53. Thomas, S.N.G.; Cheung, S.O.; Skitmore, M.; Wong, T.C.Y. An integrated regression analysis and time series model for construction tender price index forecasting. *Constr. Manag. Econ.* **2004**, *22*, 483–493.
54. Chen, B.; Maung, K. Time-varying Forecast Combination for High-Dimensional Data. *arXiv* **2020**, arXiv:2010.10435.
55. Franses, P.H. Simple Bayesian Forecast Combination. *Ann. Financ. Econ.* **2020**, *15*, 2050016. [CrossRef]
56. Cerqueira, V.; Torgo, L.; Soares, C.; Bifet, A. Model Compression for Dynamic Forecast Combination. *arXiv* **2021**, arXiv:2104.01830.
57. Chen, W.; Xu, H.; Chen, Z.; Jiang, M. A novel method for time series prediction based on error decomposition and nonlinear combination of forecasters. *Neurocomputing* **2020**, *426*, 85–103. [CrossRef]
58. Choi, E.; Cho, S.; Kim, D.K. Power demand forecasting using long short-term memory (lstm) deep-learning model for monitoring energy sustainability. *Sustainability* **2020**, *12*,1109. [CrossRef]
59. Goyal, A.; Kumar, R.; Kulkarni, S.; Krishnamurthy, S.; Vartak, M. A Solution to Forecast Demand Using Long Short-Term Memory Recurrent Neural Networks for Time Series Forecasting. In Proceedings of the Midwest Decision Sciences Institute Conference, Indianapolis, IN, USA, 12–14 April 2018.
60. Greff, K.; Srivastava, R.K.; Koutník, J.; Steunebrink, B.R.; Schmidhuber, J. LSTM: A Search Space Odyssey. *IEEE Trans. Neural Netw. Learn. Syst.* **2016** *28*, 2222–2232. [CrossRef]
61. Bandara, K.; Bergmeir, C.; Hewamalage, H. LSTM-MSNet: Leveraging Forecasts on Sets of Related Time Series with Multiple Seasonal Patterns. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 1586–1599. [CrossRef]
62. Xu, J.; Cooke, F.L.; Gen, M.; Ahmed, S.E. Stock Price Forecast Based on Lstm Neural Network. In Proceedings of the Twelfth International Conference on Management Science and Engineering Management, Ontario, ON, Canada, 5–8 August 2019; pp. 139–152.
63. Chimmula, V.; Zhang, L. Time series forecasting of COVID-19 transmission in Canada using LSTM networks. *Chaos Solitons Fractals* **2020**,*135*, 109864. [CrossRef]
64. Abbasimehr, H.; Shabani, M.; Yousefi, M. An optimized model using LSTM network for demand forecasting. *Comput. Ind. Eng.* **2020**, *143*, 106435. [CrossRef]
65. Lu, X.; Ma, C.; Qiao, Y. Short-term demand forecasting for online car-hailing using ConvLSTM networks. *Phys. A Stat. Mech. Its Appl.* **2021**, *570*, 125838. [CrossRef]
66. Agga, A.; Abbou, A.; Labbadi, M.; Houm, Y.E. Short-Term Self Consumption PV Plant Power Production Forecasts Based on Hybrid CNN-LSTM, ConvLSTM Models. *Renew. Energy* **2021**, *177*, 101–112. [CrossRef]
67. Nistor, S.C.; Moca, M.; Nistor, R.L. Discovering novel memory cell designs for sentiment analysis on tweets. *Genet. Program. Evolvable Mach.* **2020**, *22*, 147–187. [CrossRef]
68. Cho, K.; Merrienboer, B.V.; Gulcehre, C.; Ba Hdanau, D.; Bougares, F.; Schwenk, H. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In Proceedings of the EMNLP2014, Doha, Qatar, 25–29 October 2014; pp. 1724–1734.
69. Lipton, Z.C.; Berkowitz, J.; Elkan, C. A critical review of recurrent neural networks for sequence learning. *arXiv* **2015**, arXiv:1506.00019.
70. Bayer, J.; Wierstra, D.; Togelius, J.; Schmidhuber, J. Evolving memory cell structures for sequence learning. In Proceedings of the Artificial Neural Networks—ICANN 2009, Limassol, Cyprus, 14–17 September 2009; pp. 755–764.
71. Leiva-Aravena, E.; Leiva, E.; Zamorano, V.; Rojas, C.; Regan, J.M.; Vargas, I.T. Organotrophic acid-tolerant microorganisms enriched from an acid mine drainage affected environment as inoculum for microbial fuel cells. *Sci. Total Environ.* **2019**, *678*, 639–646. [CrossRef] [PubMed]