



Article

# Research on Apparel Retail Sales Forecasting Based on xDeepFM-LSTM Combined Forecasting Model

Tian Luo <sup>1,\*</sup> , Daofang Chang <sup>2</sup> and Zhenyu Xu <sup>3</sup> <sup>1</sup> School of Economics & Management, Shanghai Maritime University, Shanghai 201306, China<sup>2</sup> Logistics Engineering College, Shanghai Maritime University, Shanghai 201306, China<sup>3</sup> Institute of Logistics Science and Engineering, Shanghai Maritime University, Shanghai 201306, China

\* Correspondence: luotian10@stu.shmtu.edu.cn

**Abstract:** Accurate sales forecasting can provide a scientific basis for the management decisions of enterprises. We proposed the xDeepFM-LSTM combined forecasting model for the characteristics of sales data of apparel retail enterprises. We first used the Extreme Deep Factorization Machine (xDeepFM) model to explore the correlation between the sales influencing features as much as possible, and then modeled the sales prediction. Next, we used the Long Short-Term Memory (LSTM) model for residual correction to improve the accuracy of the prediction model. We then designed and implemented comparison experiments between the combined xDeepFM-LSTM forecasting model and other forecasting models. The experimental results show that the forecasting performance of xDeepFM-LSTM is significantly better than other forecasting models. Compared with the xDeepFM forecasting model, the combined forecasting model has a higher optimization rate, which provides a scientific basis for apparel companies to make adjustments to adjust their demand plans.

**Keywords:** sales forecasting; extreme deep factorization machine algorithm; residual prediction; long short-term memory algorithm



**Citation:** Luo, T.; Chang, D.; Xu, Z. Research on Apparel Retail Sales Forecasting Based on xDeepFM-LSTM Combined Forecasting Model. *Information* **2022**, *13*, 497. <https://doi.org/10.3390/info13100497>

Academic Editors: Agnes Vathy-Fogarassy and János Abonyi

Received: 5 August 2022

Accepted: 9 October 2022

Published: 15 October 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Sales forecasting is the anticipation and projection of future sales of a product based on its historical sales data. For companies, managers can formulate production plans and allocate production resources more rationally according to the predicted results. Due to the large number of stock keeping units (SKU) in apparel retail shops, the fragmentation of sales data and the difficulty of accurately predicting future sales, it is very difficult to guide short-term planning for retail collection shops. In retail sales forecasting, sales are influenced by various factors such as style, color, size, category to which they belong, weather, and holidays. They usually exhibit a variety of characteristics such as non-linearity, uncertainty and randomness. The main common methods for sales forecasting are time series forecasting methods and machine learning forecasting methods. The time series forecasting method focuses on the change of the sales volume itself, and it mines change trends based on historical sales data to predict sales in the future. In practice, time series forecasting relies on simpler data, but often faces the problem of hysteresis, and there is a problem of low prediction accuracy when dealing with non-linear forecasting models [1,2]. Machine learning algorithms are also an important class of methods to solve sales forecasting problems. Some of the more widely used models are Support Vector Regression (SVR) and Neural Networks (NN). The SVR model can effectively solve the problem of local minima, but it is too inefficient when dealing with large amounts of data [3]. The NN model can be trained through extensive data analysis and iterative training to make it as close as possible to the real model so that the best prediction results can be achieved, and then the trained data set is predicted to obtain the prediction results [4,5]. However, the neural network model tends to fall into local minima and its convergence rate is slow.

In recent years, decomposition models have been widely used in the field of machine learning, and they have high accuracy prediction results in the field of recommendation [6]. The difference between decomposition models and other machine learning algorithms is the treatment of feature engineering, which explores the interactions between categorical variables in the scope of big data [7]. Factorization Machine (FM) is a common machine learning algorithm based on matrix decomposition [8]. In recent years, scholars have derived other decomposition models based on FM, such as Field-aware Factorization Machine (FFM), Deep Factorization Machine (DeepFM) and Field-aware Neural Factorization Machine (FNFM). For example, Lang et al. [9] used FFM to predict movie rating predictions in an educational context and considered a clustering algorithm in the field-aware factorization machine to improve the effectiveness of the model. Ding et al. [10] proposed a multi-view Laplacian regularized DeepFM model for predicting the relationship between motivation microRNAs and the prevention, diagnosis, and treatment of complex diseases, and demonstrated the validity of the model with a case study. Zhang et al. [11] measured the value of advertising for commercial applications by predicting click-through rates using an FNFM model.

In the field of sales forecasting, the sales data is a set of time series data, so the residuals corresponding to the sales forecast results are also a set of time series data, and the temporal correlation of the residual series of sales makes the residual forecasting feasible. To improve the accuracy of the forecast, the residuals of the forecast can be corrected. We can reduce the error by building the corresponding algorithmic model, studying and fitting the pattern of residuals, and using the predicted residual values to correct the prediction results. Among them, models such as the Autoregressive Integrated Moving Average (ARIMA) model and the grey forecasting model have been applied in the field of forecasting residual correction. For example, Gilbert [12] proposed an ARIMA-based multi-stage supply chain forecasting model and used it to explain the causes of the bullwhip effect. Zhu et al. [13] developed a seasonal gray prediction model to predict air quality in China, providing a new idea for the seasonal air quality forecasting problem. Wang et al. [14] combined the gray model with the ARIMA model to form the NMGM-ARIMA technique for forecasting the production of shale oil in the United States. Due to the large number of commodities that need to be predicted, there are also many ARIMA models that need to be established, which adds a lot of burden to the forecasting exercise. In addition, due to the large number of samples in the residual series and the large degree of dispersion, the prediction effect of the grey forecasting model becomes worse. In recent years, with the rapid development of deep learning technology, deep neural networks are able to extract features from complex data, and deep learning algorithms are applied to the field of residual correction. Among them, Long Short-Term Memory (LSTM) performs well in many sequence data prediction tasks by virtue of its context-aware memory mechanism. LSTM is a model based on Recurrent Neural Networks (RNN) for processing long time series data. Compared with other sequential models, LSTM can solve the long-term dependence problem and learn useful information from the disordered residual sequence data, so that the residuals can be well fitted and predicted. LSTM is currently achieving remarkable results in many fields such as natural language processing, language translation, biogenesis and video. We introduce the method into the field of apparel sales forecasting to improve the forecasting accuracy [15,16].

In this paper, a xDeepFM-LSTM forecasting model based on residual correction is proposed for merchandise sales data of an apparel retailer. First, we used the Extreme Deep Factorization Machine (xDeepFM) algorithm to model the store's product sales data from January 2018 to October 2019 and predicted product sales in November 2019. We then used the LSTM algorithm to build a residual forecasting model, took the residual prediction result as a correction of the sales forecast result, and then obtained the final prediction results and compared them with machine learning algorithms such as CatBoost and LSTM and a deep learning algorithm. The results show that the proposed combined xDeepFM-

LSTM forecasting model outperforms the single forecasting model in sales volume time series forecasting and has better forecasting capability.

## 2. Methods and Models

### 2.1. Principle of the xDeepFM Algorithm

With the great success of Deep Neural Network (DNN) in various fields, researchers have proposed several DNN-based factorization models in order to learn low-order features and higher-order feature interactions [17]. Although DNN have the power to learn arbitrary functions from data, the method generates element interactions implicitly at the bit level. Therefore, scholars have proposed a novel Compressed Interaction Network (CIN) based on DNN which aims to generate feature interactions in an explicit manner over a vector approach, and further combine CIN and DNN into a unified model and name this new model as xDeepFM [18]. In the previous feature interaction algorithm, three modules have been developed for the content of the embedding layer, implicit higher-order interaction and explicit higher-order interaction. On the basis of these three modules, the compressed interaction network is proposed.

#### (1) Embedding layer

In the feature data affecting the apparel retail industry, the input features are usually high-dimensional discrete features, and the role of the embedding layer is to compress the original features into low-dimensional continuous features. The embedding layer is shown in Figure 1, and its final result is a wide-connected vector:

$$e = [e_1, e_2, \dots, e_m], e_i \in \mathbb{R}^D \tag{1}$$

where  $m$  denotes the number of features and  $D$  denotes the dimensionality of the original features after being processed by the embedding layer. Although the number of features is different for each sample, the length of the final wide-connected vector is the same for each sample, i.e.,  $m * D$ .

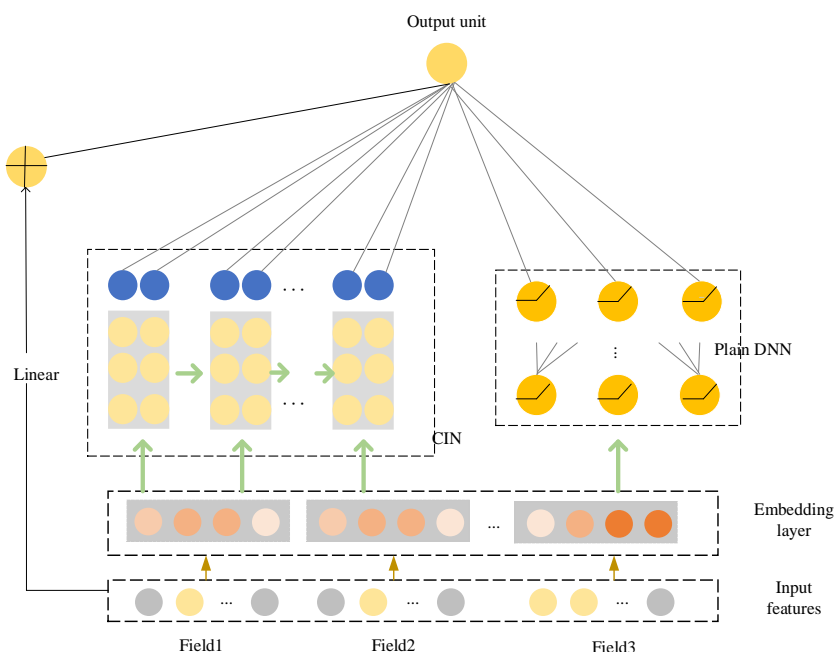


Figure 1. The structure of xDeepFM.

#### (2) Implicit higher-order interaction

Models such as Factorization Machine supported Neural Network (FNN), Deep & Cross Network (DCN), and Wide&Deep all use forward neural networks to fully learn the

information of higher-order feature interactions on the vector  $e$  of the embedding layer. The forward is calculated as follows:

$$x^1 = \sigma(W^{(1)}e + b^1) \tag{2}$$

$$x^k = \sigma(W^{(k)}x^{(k-1)} + b^k) \tag{3}$$

where  $k$  denotes the depth of the neural network layers,  $\sigma$  denotes the activation function, and  $x^k$  denotes the output of the  $k$ th layer of the neural network. Product-based Neural Networks (PNN) and DeepFM have slightly modified the above architecture [19,20]. In addition to applying DNN on the embedded vector  $e$ , they also add a bi-directional interaction layer to the architecture. Therefore, the model includes both bitwise and vector interactions. The main difference between PNN and DeepFM is that PNN connects the output of the product layer to the DNN, while DeepFM connects the FM layer directly to the output unit.

(3) Explicit higher-order interaction

In the DCN model, higher-order features are modeled explicitly, unlike forward neural networks that can only model higher-order features implicitly. the DCN modeling higher-order features are formulated as follows:

$$x_k = x_0x_{k-1}^T w_k + b_k + x_{k-1} \tag{4}$$

where each layer in CrossNet is a scalar multiple of  $x_0$ . CrossNet can learn feature interactions efficiently, but CrossNet also has its own limitations. The output of CrossNet can only be of the specified form, i.e., a scalar multiple with respect to  $x_0$ . Also, the feature crossover is obtained in bit-wise form.

(4) Compressed interaction network

Based on the above three modules, xDeepFM proposes a new interaction network, i.e., CIN. The advantages of CIN mainly include that the feature cross-correlation is obtained in vector-level form instead of element-level form; the higher-order cross-correlation of features can be obtained explicitly; and the parameter capacity does not increase exponentially as the number of network layers deepens.

Figure 1 outlines the architecture of the CIN. The figure shows the general structure of xDeepFM with three branches: Linear (sparse binary vector as input), DNN (dense vector after embedding as input), and CIN (compression perception layer). Let  $T$  denote the depth of the network. Each hidden layer  $X^k$  ( $K \in [1, T]$ ) is associated with an output unit. Applying the summation pool to each feature map in the hidden layer.

$$p_i^k = \sum_{j=1}^D X_{i,j}^k \tag{5}$$

where  $i \in [1, H_k]$ . Therefore, for the  $k$ th hidden layer, there is a merge vector  $P^k = [p_1^k, p_2^k, \dots, p_{H_k}^k]$ . All merge vectors from the hidden layer are concatenated before being connected to the output unit as:  $P^+ = [P^1, P^2, \dots, P^T] \in \mathbb{R}^{\sum_{i=1}^T H_i}$ . If the CIN is used directly for binary classification, the output unit is a sigmoid node on  $P^+$ :

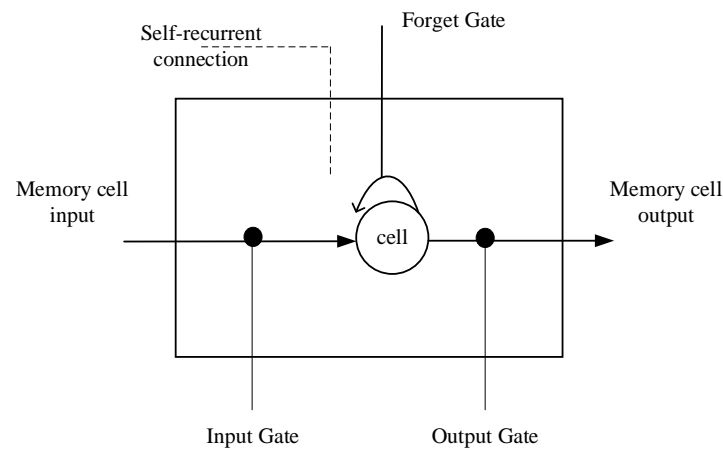
$$y = \frac{1}{1 + \exp(p^{+T}w^o)} \tag{6}$$

where  $w^o$  is the regression parameter.

2.2. Principle of LSTM Algorithm

The LSTM is a kind of temporal recurrent neural network, which is specially designed to solve the long-term dependence problem of general RNN in dealing with long sequences. It solves the problems of insufficient long-term memory, gradient disappearance and

gradient explosion of RNN, and enables recurrent neural networks to really use the long-distance temporal information effectively [21]. Unlike RNN, LSTM adds three types of logic control units to RNN, namely input gate, output gate and forget gate. The LSTM connects each of these three cells to a multiplicative element, and controls the input and output of the information flow and the state of the memory cell by setting the weights at the edges where the memory cell of the neural network is connected to the other parts. The specific structure of the LSTM neuron is shown in Figure 2.



**Figure 2.** Internal structure of LSTM neuron.

The relevant descriptions of the components of Figure 2 are as follows:

**Input Gate:** Determines how much of the input data to the network at the current moment needs to be saved to the cell state, denoted as  $i_t$ .

**Forget Gate:** Determines how much of the cell state from the previous moment needs to be retained until the current moment, denoted as  $f_t$ .

**Output Gate:** Controls how much of the current cell state needs to be output to the current output value, denoted as  $o_t$ .

**Neuron:** A memory representing the state of a neuron that gives the LSTM unit the ability to save, read, reset and update long-range historical information, denoted as  $c_t$ .

At moment  $t$ , the expressions for the oblivion gate, the input gate, and the output gate are as follows:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{7}$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{8}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{9}$$

$$h_t = o_t * \tanh(c_t) \tag{10}$$

$$\tilde{c}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{11}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{12}$$

where  $f_t$  is the forgetting gate output,  $i_t$  is the input gate output, and  $o_t$  is the output gate output.  $W_f$ ,  $W_i$ ,  $W_o$  and  $W_c$  are the parameter matrices of forgetting gate, input gate, output gate, and neuron state, respectively.  $\sigma$  and  $\tanh$  are two activation functions.

In the training process of LSTM, first we input the feature data at moment  $t$  to the input layer and output the result using the excitation function. We then input the output results of the input layer, the output of the hidden layer at moment  $t - 1$  and the information stored in the neurons at moment  $t - 1$  into the nodes of the LSTM structure. Finally, by processing the input gates, output gates, forgetting gates and neuron units, we output the data to the output layer or the next hidden layer, output the results of the LSTM structure nodes to the output layer neurons, and calculate the back propagation error and update the individual weights.

### 3. xDeepFM-LSTM Combined Forecasting Model

The volume of sales data in the apparel retail industry is huge, and most of the sales influencing factors are category-based data, such as the major category of products to which the products belong, the minor category of products to which the products belong, the applicable gender category of products, the applicable age group category of products, the positioning category of products, and weather conditions. These data have the characteristics of high sparsity and high dimensionality. The xDeepFM model not only can solve these problems well, but can also explore the correlation between features and then predict the sales of goods.

In order to improve the prediction accuracy of the forecasting model, we can apply residual correction to the residuals of the prediction results. The residuals are largely due to the influence of the characteristics of the commodities themselves and external factors. Adding feature influence factor data to the input data will help to improve the residual prediction effect. In addition, the residuals have the characteristics of disorder, the unique gate structure of LSTM in deep learning can eliminate the useless historical information in the residual sequence and selectively retain the useful information.

Based on the above discussion, we propose a sales forecasting model that combines sales forecasting using xDeepFM with residual correction using LSTM, i.e., the xDeepFM-LSTM model. The algorithm flow chart of the xDeepFM-LSTM model is shown in Figure 3.

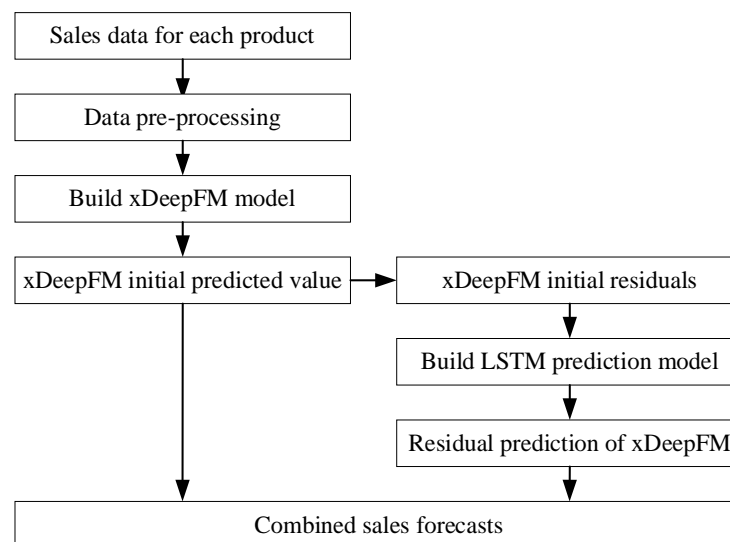


Figure 3. Flow chart of combined model prediction.

### 4. Example Analysis

#### 4.1. Experimental Data Description and Pre-Processing

The dataset in this paper ranges from 1 January 2017 to 31 December 2019 and is obtained from the ERP system of a fast fashion retailer in Guangdong Province, China. There are a total of 10,263 SKUs and 4,198,169 data records. The data mainly includes 21 dimensions, such as the product number, daily sales, product price, category of the product, weather condition, maximum weather temperature, minimum weather temperature, and whether it is a holiday. Due to the different sales time on the shelf, the corresponding SKU has different series lengths. In the actual dataset, the sales variables obeyed an approximate normal distribution. Detailed information of the series lengths is provided in Table 1.

**Table 1.** The series lengths of real-life datasets and simulated datasets.

Description	Real-Life Dataset	Simulated Dataset
Mean	211.93	277.91
Std	155.22	188.16
Min	60.00	60.00
25%	108.00	116.00
50%	151.00	177.00
75%	256.00	284.00
Max	1091.00	1095.00

The dataset used in the paper is a good representation of other real datasets for the following reasons: (1) The components of the apparel dataset comprise complex information (i.e., trend, cycle, and seasonality), and the validation results of the proposed model can be used as a reference for other industries. (2) The apparel dataset has more than 4 million records and contains feature data in more than 20 dimensions, which provides sufficient data to test the effectiveness of different models.

In the original sales data, different evaluation indicators have different magnitudes and magnitude units. In order to eliminate the influence of the scale between indicators and avoid the situation that changes in larger values will cover changes in smaller values, the data needs to be normalized. There are various ways of normalization, and the most-valued normalization method is used in this paper, which is calculated as follows:

$$x_{norm}(i) = \frac{x(i) - \min(x)}{\max(x) - \min(x)} \quad (13)$$

For a feature,  $\min(x)$  is the minimum value of the feature and  $\max(x)$  is the maximum value of the feature. The formula for the inverse normalization is as follows:

$$x(i) = x_{norm}(i) * (\max(x) - \min(x)) + \min(x) \quad (14)$$

For the data input requirements of the xDeepFM model, we encode the discrete data in the experimental data with labels. When using LSTM for residual correction, we perform one-hot encoding process for the discrete data in the experiment.

#### 4.2. Selection of Evaluation Indicators

The main objective of this paper is to predict the sales of a commodity at a specific date in the future. We use the Root Mean Square Error (RMSE) and the Mean Absolute Error (MAE) to evaluate the forecast results. These two indicators can reflect, to varying degrees, the deviation of the predicted value from the true value, as well as the degree of dispersion between them. For RMSE and MAE, smaller values indicate that the deviation of the predicted value from the true value is smaller and the degree of dispersion is smaller. The expressions for RMSE and MAE are:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n |x - x^*|^2} \quad (15)$$

$$MAE = \frac{1}{n} \sum_{i=1}^n |x - x^*| \quad (16)$$

where  $x$  is the actual value of product sales on the  $i$ th day,  $x^*$  is the predicted value of product sales on the  $i$ th day, and  $n$  is the total number of days involved in the test.

#### 4.3. xDeepFM-LSTM Sales Forecasting Model Implementation

This experiment used PyTorch as the framework for neural networks, and used Python to implement the code for xDeepFM. We categorized the sales data from January 2018 to

October 2019 as the training set and the sales data from 2019 as the test set. Then, we set the optimizer to “adam”, the loss function to “mse”, and the evaluation metric to “accuracy”. Next, we initialized the xDeepFM forecasting model empirically, setting batch\_size to 32 and epoch to 10. We apply the trained xDeepFM to predict all commodities from month 2018 to November 2019 and obtain the prediction  $X(t)$ . The equation for mse is:

$$MSE = \frac{1}{n} \sum_{i=1}^n (\tilde{Y}_i - Y_i)^2 \quad (17)$$

Based on the prediction results obtained using the xDeepFM model, we obtained the residual values using the following equation.

$$\varepsilon_i = x_i - x'_i \quad (18)$$

where  $x_i$  is the true sales value of the item,  $x'_i$  is the predicted value of xDeepFM, and  $\varepsilon_i$  is the residual value of the item.

This experiment used PyTorch as the framework for neural networks, and used Python to implement the code for residual correction of the LSTM model. This paper initializes the LSTM neural network model empirically, and after several debuggings and experiments, the optimizer is set to Adam, the training number is set to 20 times, and the LSTM model is packaged as a regression model. Finally, we use the trained LSTM network to make predictions on the test set, and the output is  $L(t)$ . We combined the prediction results of the xDeepFM model with those of the LSTM model to obtain the final prediction results, which are given by the following equation.

$$Y(t) = X(t) + L(t) \quad (19)$$

where  $X(t)$  is the prediction result of xDeepFM model,  $L(t)$  is the result of residual correction by LSTM, and  $Y(t)$  is the final prediction result.

#### 4.4. Analysis of Results

Through error calculation and comparison, the total number of commodities in the test set is 1263, and the number of commodities whose error value after residual correction is smaller than the error value before correction is 964, and the correction optimization rate has reached 76%. To further verify the superiority of the xDeepFM-LSTM model, this study compares it with both NN, SVR, ARIMA, Naive, CatBoost, LSTM, and xDeepFM prediction models. With regard to the parameters of models, we utilize the Akaike's Information Criterion (AIC) to determine the order of the ARIMA model. In addition, the grid search method is employed to tune the parameters for the NN, SVR, Catboost, LSTM, xDeepFM, and xDeepFM\_LSTM model. In addition, to be able to evaluate the performance of the models proposed in this study more objectively, we also calculated the error values of different models based on the simulation data. The error evaluation results of each model are shown in Table 2. Subsequently, we selected three top-performing algorithms, the xDeepFM-LSTM algorithm, the LSTM algorithm and the CatBoost algorithm, and compared their predictions with actual sales, and the results are shown in Figure 4.

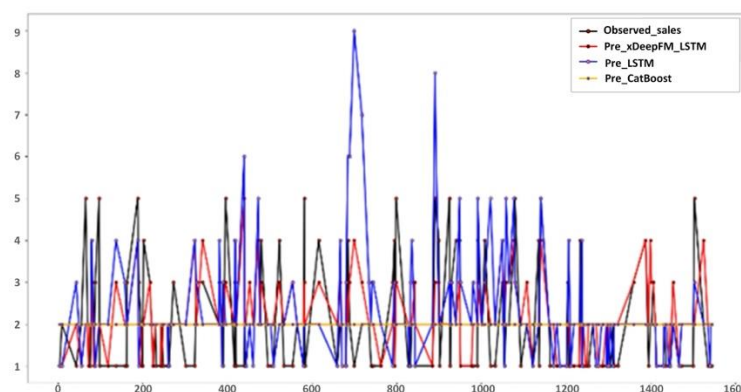
As shown in Table 2, the overall prediction performance of the xDeepFM model is better than that of forecasting models such as CatBoost and LSTM. The prediction accuracy of the combined xDeepFM-LSTM model is also improved relative to the xDeepFM model.

As can be seen from Figure 4, the prediction results of the combined xDeepFM-LSTM model are closer to the true value of sales than the LSTM algorithm and the CatBoost algorithm. It can be concluded that the combined xDeepFM-LSTM forecasting model is more applicable than other forecasting models in this enterprise.



**Table 2.** Comparison of error values for each model under real-life and simulated data.

Dataset	Forecasting Models	RMSE	RMSE_Me	RMSE_Std	MAE	MAE_Me	MAE_Std
Real-life data	Naive	<b>1.89</b>	1.73	<b>0.15</b>	<b>1.23</b>	<b>1.18</b>	<b>0.18</b>
	ARIMA	1.85	<b>1.76</b>	0.13	1.18	1.12	0.16
	NN	1.77	1.63	0.09	1.09	1.01	0.13
	SVR	1.79	1.60	0.09	1.06	0.99	0.10
	CatBoost	1.70	1.65	0.08	1.05	0.96	0.09
	LSTM	1.40	1.38	0.07	0.91	0.88	0.09
	xDeepFM	1.37	1.33	0.07	0.91	0.84	0.09
	xDeepFM_LSTM	<b>1.32</b>	<b>1.28</b>	<b>0.06</b>	<b>0.85</b>	<b>0.80</b>	<b>0.08</b>
Simulated data	Naive	<b>2.35</b>	<b>2.22</b>	<b>0.32</b>	<b>2.11</b>	<b>2.06</b>	0.25
	ARIMA	2.22	2.10	0.30	2.02	1.95	<b>0.27</b>
	NN	2.10	2.03	0.28	1.99	1.91	0.23
	SVR	2.06	2.01	0.26	1.85	1.80	0.21
	CatBoost	1.98	1.95	0.22	1.81	1.77	0.21
	LSTM	1.95	1.87	0.18	1.76	1.72	0.19
	xDeepFM	1.88	<b>1.75</b>	<b>0.15</b>	1.68	1.61	0.18
	xDeepFM_LSTM	<b>1.81</b>	<b>1.75</b>	<b>0.15</b>	<b>1.54</b>	<b>1.51</b>	<b>0.17</b>



**Figure 4.** Comparison of the prediction performance of each algorithm.

A statistical test is essential to determine whether the better performance is statistically significant. We run the KSPA test (the Kolmogorov-Smirnov Predictive Accuracy test) to provide statistical evidence. The KSPA test enables the distinguishing of the forecast distribution of the two models and determines whether lower error also represents lower stochastic error.

The results from the KSPA test compare the prediction errors of xDeepFM\_LSTM with CatBoost, xDeepFM\_LSTM and LSTM, xDeepFM\_LSTM and xDeepFM based on different datasets, which are shown in Table 3. The results show that there exists a statistically significant difference between the distribution of prediction errors from xDeepFM\_LSTM and other models at a 99% confidence level.

**Table 3.** The results of KSPA test.

Dataset	Groups	Two-Sided ( <i>p</i> -Value)	Greater ( <i>p</i> -Value)
Real-life data	xDeepFM_LSTM vs. CatBoost	<0.001 *	<0.001 *
	xDeepFM_LSTM vs. LSTM	<0.001 *	<0.001 *
	xDeepFM_LSTM vs. xDeepFM	<0.001 *	<0.001 *
Simulated data	xDeepFM_LSTM vs. CatBoost	<0.001 *	<0.001 *
	xDeepFM_LSTM vs. LSTM	<0.001 *	<0.001 *
	xDeepFM_LSTM vs. xDeepFM	<0.001 *	<0.001 *

Note: \* indicates results are statistically significant based on a *p*-Value of 0.001.

## 5. Conclusions

For apparel retailers, analyzing the patterns and characteristics of apparel sales data to improve the accuracy of sales forecasting as much as possible can guide them to make effective marketing strategies. According to the data characteristics of the sales volume of an apparel retailer, this paper proposes a combined forecast model of xDeepFM-LSTM. First, we used the xDeepFM forecasting model to mine the interactions between categorical variables under the data range and made predictions. We then utilized the LSTM model for residual correction and then proposed the combined xDeepFM-LSTM forecasting model. We compared it with the pre-combination xDeepFM model, other forecasting models in machine learning and deep learning fields. The experimental results show that the combined xDeepFM-LSTM forecasting model outperforms the CatBoost and LSTM models in terms of prediction performance. Compared with xDeepFM, the xDeepFM-LSTM achieves a prediction performance optimization rate of 76%. To conclude, the xDeepFM-LSTM proposed in this paper has better prediction performance in predicting the merchandise sales of retail stores.

The combined forecasting model of xDeepFM-LSTM proposed in this paper has an optimization rate of 76% in prediction results, and further optimization studies can be conducted on this basis in the future to improve the optimization rate. In addition, the parameters of the LSTM model in this paper are adjusted according to manual experience, and further research can be conducted on the optimization parameters to optimize the performance of the model as much as possible.

**Author Contributions:** Methodology, software, validation, formal analysis, data curation, writing—original draft preparation and writing—review and editing, T.L.; resources, supervision, project administration and funding acquisition, D.C.; Conceptualization, formal analysis, investigation and visualization, Z.X.; All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Ministry of Industry and Information Technology of China for Cruise Program [No. 2018-473]; the National Key Research and Development Program of China [No. 2019YFB1704403].

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Informed consent was obtained from all subjects involved in the study.

**Data Availability Statement:** The data used to support the findings of this study are included within the article.

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

## References

1. Xu, F.L.; Lin, Y.Y.; Huang, J.X.; Wu, D.; Shi, H.Z.; Song, J.; Li, Y. Big Data Driven Mobile Traffic Understanding and Forecasting: A Time Series Approach. *IEEE Trans. Serv. Comput.* **2016**, *9*, 796–805. [[CrossRef](#)]
2. Yang, Y.; Lu, J.G. A Fusion Transformer for Multivariable Time Series Forecasting: The Mooney Viscosity Prediction Case. *Entropy* **2022**, *24*, 528. [[CrossRef](#)] [[PubMed](#)]
3. Panagopoulos, O.P.; Xanthopoulos, P.; Razzaghi, T.; Seref, O. Relaxed support vector regression. *Ann. Oper. Res.* **2019**, *276*, 191–210. [[CrossRef](#)]
4. Jin, Y.H.; Lee, K.H.; Choi, D.W. QueryNet: Querying neural networks for lightweight specialized models. *Inf. Sci.* **2022**, *589*, 186–198. [[CrossRef](#)]
5. Lee, S.H.; Lee, S.J. Flexible Forecasting Model Based on Neural Networks. *Wirel. Pers. Commun.* **2017**, *94*, 283–300. [[CrossRef](#)]
6. Shi, J.M.; Leau, Y.B.; Li, K.; Park, Y.J.; Yan, Z.W. Optimization and Decomposition Methods in Network Traffic Prediction Model: A Review and Discussion. *IEEE Access* **2020**, *8*, 202858–202871. [[CrossRef](#)]
7. Pholsena, K.; Pan, L.; Zheng, Z.P. Mode decomposition based deep learning model for multi-section traffic prediction. *World Wide Web* **2020**, *23*, 2513–2527. [[CrossRef](#)]
8. Zhou, J.D.; Zhang, Q.P.; Li, X. Fuzzy factorization machine. *Inf. Sci.* **2021**, *546*, 1135–1147. [[CrossRef](#)]
9. Lang, F.; Liang, L.L.; Huang, K.; Chen, T.; Zhu, S.X. Movie Recommendation System for Educational Purposes Based on Field-Aware Factorization Machine. *Mob. Netw. Appl.* **2021**, *26*, 2199–2205. [[CrossRef](#)]
10. Ding, Y.L.; Lei, X.J.; Liao, B.; Wu, F.X. MLRDFM: A multi-view Laplacian regularized DeepFM model for predicting miRNA-disease associations. *Brief. Bioinform.* **2022**, *23*, bbac079. [[CrossRef](#)] [[PubMed](#)]

11. Zhang, L.; Shen, W.; Huang, J.; Li, S.; Pan, G. Field-Aware Neural Factorization Machine for Click-Through Rate Prediction. *IEEE Access* **2019**, *7*, 75032–75040. [[CrossRef](#)]
12. Gilbert, K. An ARIMA supply chain model. *Manag. Sci.* **2005**, *51*, 305–310. [[CrossRef](#)]
13. Zhu, X.Y.; Dang, Y.G.; Ding, S. Forecasting air quality in China using novel self-adaptive seasonal grey forecasting models. *Grey Syst. Theory Appl.* **2021**, *11*, 596–618. [[CrossRef](#)]
14. Wang, Q.; Song, X.X.; Li, R.R. A novel hybridization of nonlinear grey model and linear ARIMA residual correction for forecasting US shale oil production. *Energy* **2018**, *165*, 1320–1331. [[CrossRef](#)]
15. Wang, L.L.; Wang, C.Y.; Wang, F.W.; Chu, Y.H.; Yang, Z.D.; Wang, H. EPI phase error correction with deep learning (PEC-DL) at 7 T. *Magn. Reson. Med.* **2022**, *88*, 1775–1784. [[CrossRef](#)] [[PubMed](#)]
16. Su, H.Y.; Tang, C. Dynamic-Error-Compensation-Assisted Deep Learning Framework for Solar Power Forecasting. *IEEE Trans. Sustain. Energy* **2022**, *13*, 1865–1868. [[CrossRef](#)]
17. Ghasemi, F.; Mehridehnavi, A.; Fassihi, A.; Perez-Sanchez, H. Deep neural network in QSAR studies using deep belief network. *Appl. Soft Comput.* **2018**, *62*, 251–258. [[CrossRef](#)]
18. Chen, L.; Shi, H.Y. DexDeepFM: Ensemble Diversity Enhanced Extreme Deep Factorization Machine Model. *ACM Trans. Knowl. Discov. Data* **2022**, *16*, 1–17. [[CrossRef](#)]
19. Qu, Y.R.; Fang, B.H.; Zhang, W.N.; Tang, R.M.; Niu, M.Z.; Guo, H.F.; Yu, Y.; He, X.Q. Product-Based Neural Networks for User Response Prediction over Multi-Field Categorical Data. *ACM Trans. Inf. Syst.* **2019**, *37*, 5. [[CrossRef](#)]
20. Teng, S. Route planning method for cross-border e-commerce logistics of agricultural products based on recurrent neural network. *Soft Comput.* **2021**, *25*, 12107–12116. [[CrossRef](#)]
21. Santra, A.S.; Lin, J.L. Integrating Long Short-Term Memory and Genetic Algorithm for Short-Term Load Forecasting. *Energies* **2019**, *12*, 2040. [[CrossRef](#)]