Information Sciences 625 (2023) 97–124

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

A multiple long short-term model for product sales forecasting based on stage future vision with prior knowledge

Daifeng Li [a,*], Xuting Li [a], Kaixin Lin [a], Jianbin Liao [a], Ruo Du [b], Wei Lu [c], Andrew Madden [d]

[a] *School of Information Management, Sun Yat-Sen University, Guangzhou, Guangdong, China*
[b] *Galanz Company, Shenzhen, Guangdong, China*
[c] *School of Information Management, Wuhan University, Wuhan, Hubei, China*
[d] *Information School, University of Sheffield, Sheffield, United Kingdom*

A R T I C L E   I N F O

Article history:
Received 13 June 2021
Received in revised form 5 November 2022
Accepted 29 December 2022
Available online 5 January 2023

Keywords:
Stage future vision
Dynamic co-integration
Prior knowledge
Two-layer convolutional neural network
Two-stage LSTM
Long short-term modeling

A B S T R A C T

Deep neural network (DNN) based multivariate time series (MTS) forecasting has been widely studied in many domains. The approach has also been successfully applied to product sales forecasting, which is invaluable in the strategic development of enterprises. However, one key challenge is to efficiently combine all influential factors into a unified framework by considering their long short-term correlations. This is particularly challenging for real time sales predictions with many important features unknown from the perspective of future vision, because of the complex and dynamic changing environment of sales time series. Besides, DNN based methods could not effectively capture stable linear correlations between multivariate sales time series. To address these challenges, a novel Stage future-vision-based multiple Long Short-term model with prior knowledge (SLST-PKNet) is proposed. The model constructs sub-models according to the types of different influential factors, and a stage future vision mechanism is used to model dependent correlations between future influential factors and product sales by combining a two-layer convolutional neural network (TLCNN) and a two-stage LSTM (TSLSTM) into a unified framework. The combination of TLCNN and TSLSTM is the basic component for long short-term modeling, and is an effective solution for capturing dynamic patterns by fusing the outputs from different layers and stages. A dynamic co-integration mechanism (DCI) is introduced to capture strong correlations between time series, which DNN is not good at. In order to further improve the capability of the model to capture long short-term patterns from complex environment, domain prior knowledge (PK) is integrated as supervision information. Extensive experiments are conducted on two sales datasets: Galanz and Cainiao. SLST-PKNet achieves significant performance improvements over 11 state-of-the-art baselines. The proposed model is also evaluated on two new datasets: Traffic and Exchange-Rate, to further verify its generalization capability. The data and code could be visited at: https://github.com/lixt47/SLST-PKNet.

* Corresponding author.
E-mail addresses: lidaifeng@mail.sysu.edu.cn (D. Li), lixt47@mail2.sysu.edu.cn (X. Li), ruodu56@hotmail.com (R. Du), weilu@whu.edu.cn (W. Lu).

https://doi.org/10.1016/j.ins.2022.12.099
0020-0255/© 2023 Elsevier Inc. All rights reserved.

1. Introduction

1.1. Background

Product sales prediction is an important task for enterprises, and has been widely studied in both academia and industry [10,33,34,35]. Accurate forecasts of product sales can help enterprises to significantly improve their revenues by, for example, streamlining inventory management. A key challenge of product sales prediction is to efficiently model the complex, non-linear dependencies that exist, not only between time steps but also in a variety of variables [4,36,37]. This could be regarded as a multivariate time series (MTS) prediction problem. Examples of relevant dependencies include the new arrival of similar products, new promotion strategies from competitors, the impact of the time that promotion occurs, etc. Traditional time series prediction methods such as vector auto-regression (VAR) [2,25] and gaussian process (GP) [32] usually assume certain distributions, and ignore dependencies between variables [27].

In recent years, deep neural network (DNN) based multivariate time series analysis has been successfully used to predict financial trends, analyze traffic jams, and forecast electricity consumption. DNN has advantages arising from their flexibility in capturing nonlinearity. One widely-used approach to DNN-based multivariate time series prediction is to use a recurrent neural network (RNN) [6,11,20]. long short-term memory (LSTM) [17] and gated recurrent unit (GRU) [6] have been used to capture long-range dependencies of time series due to the vanishing gradient problem associated with RNN [6,17]. Combining attention mechanisms [38] can further improve the capability of RNNs to help model temporal patterns between fragments of input time sequences and output predictions [13,15,18]. Other technologies that have performed well in time series forecasting include Transformer [23], tensor decomposition [36], adversarial training [37], time series decomposition [27], extreme value prediction [16,21], and explainable time series prediction [14,30]. Recently, Zhou et al [43] designed a novel Transformer framework for efficient long time-range dependency analysis, which attracts many researchers' attention in making long-term analysis.

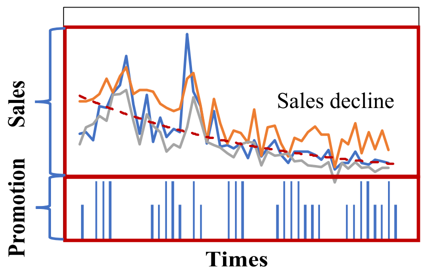
1.2. Limitations of existing research for sales prediction

Modeling a mixture of influences of long short-term patterns [2,28,41], for which traditional approaches may fail, is an important research direction for MTS prediction. The approach uses a convolutional neural network (CNN) to capture local correlations among different time series in each time step, and then applies LSTM/GRU to model long and short-term temporal patterns [20]. Existing research performs well on time series with significant patterns: for example, traffic time series have a strong periodic pattern; time series of different exchange rate have strong correlation patterns. Factors influencing sales time series however, are more complex, due to the changeable and complex environment. Consequently, traditional methods do not perform the task of sales prediction as well as expected. The main reasons for their limitations are summarized below:

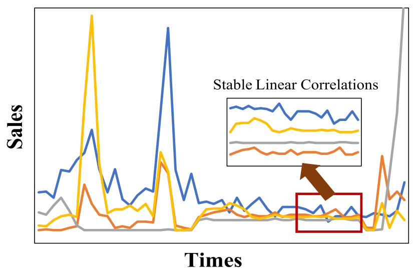
(1) Existing DNN-based models take all time series as input without considering their different types or attributes. The prediction of sales is influenced by many changeable and complicated internal and external factors. Such as price fluctuations, competitive products, ad hoc promotion activities, etc. Different types of time series and their inner correlations have different effects on the sales prediction. However, current researches seldom address this problem. An example can be seen in Fig. 1(a), which presents a situation where, although the promotional intensity is increasing, the sales of all products of the same type are falling. If we use traditional methods to model all the time series without considering their different types, the model will learn contradictory correlations, which will lead to prediction errors.

(2) The proposed CNN-LSTM framework seldom considers capturing stable correlations between input time series. The non-linear nature of the CNN-LSTM improves the capture of non-linear correlations, but its ability to detect stable linear correlations is reduced. However, traditional methods adopt auto-regression (AR) to model linear correlations, which overlook stable correlations between input time series. Fig. 1(b) shows an example where correlations between sales time series in a MTS are complex, but for a subset of the MTS, over a certain time period, there will exist stable correlations between the time series in the subset (a phenomenon that is often caused by bundle promotion strategies), and the stable correlations are useful for predicting sales.

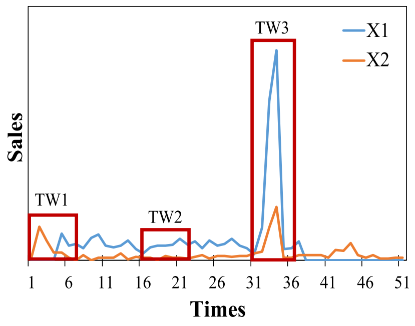
(3) Existing researches are limited when it comes to practical applications, such as the prediction of product sales for which future influence factors of target sales time series are usually unknown. For example, the challenge of predicting sales of a product over a given time period ($t + 1$). Promotion strategies affecting the product at $t + 1$ is usually unknown, because from the perspective of the model at time t, they have yet to occur. Relevant studies such as MLCNN, DARNN accomplish similar tasks by providing a means to fuse near and distant vision [5,28]. However, the solution seldom considers using a more logical method to model future representations of influence features or factors based on current time series. In addition (see Fig. 1(c)), because of the complex and dynamic changing environment, frequency and range of sales often fluctuates a great deal (As shown in windows TW1 $\sim$3), and the weight-sharing strategy of MCLNN and DARNN cannot effectively capture those dynamic patterns.



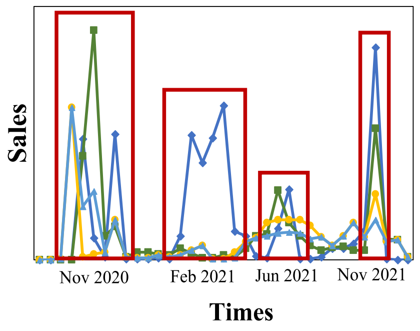
(a) Different influences of different time series types.



(b) Capturing stable correlations from MTS.



(c) Complex, dynamic changing patterns.



(d) Influences of domain prior knowledge

Fig. 1. Examples of the unique characteristics of sales multivariate time series.

(4) Previous research seldom considers using prior knowledge to address the influence of long short-term patterns. Memory time-series network (MTNet) adopt a memory-network to record long short-term variable dependencies [3]. MTNet performs well on time series with strong temporal patterns, such as traffic time series. For sales time series however, the influential factors are more complex, making patterns hard to detect without prior knowledge. As seen in Fig. 1(d), sales of products peak at several fixed time points such as Nov, Feb and Jun in a year, which indicate holidays and well-known e-commerce promotion days. Conventional promotion features, such as daily discounts cannot fully explain peaks in sales at the fixed time without indicating their importance in advance, because there are other promotions, such as advance deposit discounts, coupons, etc., running concurrently. A further consideration is that the performance of the products at those time points affect subsequent sales. Current studies cannot incorporate such potential correlations without specifying them in advance.

2. Research objective

The direct application of existing DNN-based methods to sales prediction will cause significant errors, because recent research seldom takes into account the unique characteristics of sales time series. Here, we propose a novel SLST-PKNet model that systematically optimizes the existing DNN-based model by considering such characteristics, including the influences of different feature time series, and the influences of long short-term patterns with prior knowledge, also by detecting stable linear correlations and dynamic non-linear patterns between MTS. In addition, the problems existing in the practical application of the model are taken into consideration.

2.1. Solution and innovations

The proposed SLST-PKNet model addresses these issues by combining stage future vision-based feature representations and MTS based long short-term patterns detection with prior knowledge (PK), into a unified framework. The main benefits of the new model are summarized below:

- Three sub-models have been developed to investigate different influences of different types or features of time series on target product sales time prediction: (1) Product sub-model investigates common temporal patterns from different time series dealing with the same product type; (2) Marketing sub-model investigates the influence of marketing features (such as promotion or marketing strategies); (3) Product-Marketing sub-model captures the complex influence of both product and marketing features on the target product.

- A TLCNN-TSLSTM framework is proposed to model future vision-based long short-term variable dependencies in a more logical way. This has the potential to increase the accuracy of predictions of product sales by adopting a Two-stage LSTM (TSLSTM) to fuse the outputs of a two-layer convolutional neural network (TLCNN). The methods can result in better modeling of the dynamic changeable environment of sales predictions compared with existing methods.
- A dynamic co-integration (DCI) component and a prior knowledge (PK) component are introduced to help construct the influence of dynamic temporal patterns. The DCI can efficiently capture stable linear correlation patterns between target and feature time series. This may significantly contribute to predictions of future product sales. The PK component integrates prior knowledge (such as periodic influences, promotional events and marketing plans) into the long short-term model, helping to guide model training more accurately.
- Extensive experiments were conducted on two real product datasets: Galanz and Cainiao. The results show that the proposed model significantly outperforms 11 state-of-the-art baselines. An ablation test verifies the contributions of each module. In addition, experiments conducted on two public datasets, from Traffic and Exchange-Rate, also show the potential value.

The SLST-PKNet is an optimization solution developed according to the characteristics of sales time series. However, it is also relevant to time series prediction tasks in other fields. In the following sections, ‘Related Work’ reviews other works in the field and discusses their limitations to the task of sales prediction. ‘Model Description’ focuses on the innovations and introduces each component in detail. ‘Experiments’ describes the process by which the performance of the proposed model was verified. In the ‘Discussion’ section, both theoretical and practical significance of the research is explored.

3. Related work

3.1. Statistical analysis-based time series prediction

Research into time series modeling has been integral to topics involving forecasts based on trends, such as financial predictions, traffic flow analysis, forecasts of electricity consumption, etc. Traditional methods mainly focus on statistical analysis based parametric models [2,15,20], such as AR, ARIMA, exponential smoothing, structural linear models, vector auto regression (VAR), simultaneous equation models, etc. When applying traditional methods to MTS prediction, it is necessary to carry out stationarity and co-integration tests between time series to ensure strict and stable linear correlations [2,12,26]; then to solve the pseudo regression problem to a certain extent. Periodic influence is also investigated in many studies. Some researchers treat the time interval p of the cycle as a known, (eg, seasonal influences), so when predicting the values at time $t+1$, the value at $t+1-p$ will be taken as an important feature [20,27]. Other researchers argue that there are many periodic patterns in a MTS, so they often use a window or a periodic function to scan the MTS for potential patterns [7,9,26,42].

Most existing studies use manual methods to select the appropriate time series for regression analysis through a co-integration test. This may lead to two problems when the method is directly applied to real sales predictions. Firstly, manual selection will omit many useful patterns; secondly, manual selection is not a good way to combine models for capturing non-linear correlations into a unified framework. Another problem is that current periodic research seldom considers the influence that the local sequence of periodic events has on the prediction, and the accumulation of the influence of different periodic events.

3.2. Deep neural network-based time series prediction

Deep neural networks (DNNs) are attracting increasing attention in the field of multivariate time series (MTS) forecasting. One popular learning framework that uses DNN-based methods is CNN-LSTM/GRU. LSTNET [20], for example, models long short-term patterns based on a CNN-GRU framework. To do so, it uses a recurrent-skip component to model long-term temporal patterns which can be controlled by assigning parameter p. Dual-stage attention-based recurrent neural nets (DA-RNN) [28] can also model long-term temporal patterns based on Multivariate Time Series. However, they are thought not to give sufficient consideration to spatial correlations among different components of exogenous data. Dual self-attention networks (DSANET) [18] adopt CNN to model global and local temporal patterns, and use a self-attention mechanism to extract sequence features. MTNet [4] designs a memory component and incorporates it with three separate encoders and an auto-regressive component; it then trains all the components jointly. MLCNN [5] provides a near and distant fusion vision method to improve predictive performance by adopting a multi-task learning framework and fusing forecasting information. Spectral time graph neural networks (StemGNN) [3] combine graph Fourier transforms (GFT) and discrete Fourier transforms (DFT) into an end-to-end framework. GFT and DFT are used to model inter-series correlations and temporal dependencies separately. After passing the input data through GFT and DFT, the spectral representations hold clear patterns and can be predicted effectively by convolution and sequential learning modules. Yakhchi et al. [40] proposed a novel convolutional attention network, which could effectively capture sequential patterns for future decision-making. Park et al. [31] proposed DeepGate, a novel time series forecasting framework based on the explicit global–local decomposition to further improve the performances of MTS predictions. Transformer [38] based methods have also been applied to the task of MTS prediction [23]. The multi-head attention and position embedding of Transformer can help capture useful information between different time points from both long and short-term perspectives. Informer [43] and Pyraformer [24] are two representative Transformer-based models. Informer mainly uses the sparsity of attention scores to make optimization. Pyraformer designs a new attention module to bridge the gap between capturing remote dependencies and reducing the temporal and spatial complexity. The two models have achieved excellent performance on multiple datasets.

3.3. Product sales prediction

Forecasting of product sales has been studied for decades and continues to attract attentions. Methods based on traditional machine learning rely mainly on ARIMA, vector auto-regression, feature engineering and ensemble learning [14,34]. Shen et al. [34] incorporated popular machine learning models such as ARIMA, Xgboost, support vector regression, gaussian process etc, into a unified ensemble model to forecast product sales for JD.com, one of the largest e-commerce companies in China. They also shared their experiences of constructing large-scale feature sets, such as correlation coefficients, entropy, SKU promotion et al, which could significantly improve forecasting accuracy.

In recent years, DNN-based methods have been used to improve sales predictions by capturing non-linear correlations, Fan et al. [13] proposed a novel data-driven model to learn hidden patterns' representations with DNN and attending to different parts of the history for forecasting the future. Shi et al. [36] proposed a novel BHT-ARIMA model, which incorporates multi-way delay embedding transform (MDT) tensors and tensor ARIMA into a unified framework to capture the intrinsic correlations among multiple time series. The model converts the input MTS into a higher order tensor by conducting duplication matrix operations; it then calculates the kernel tensor by using low-rank tensor decomposition. They used the kernel tensor to predict sales of computers. Ekambaram et al. [10] applied attention based multi-modal time-series forecasting to new products in a dataset from a famous fashion house. All the multiple time-series were modelled together based on product images and optional attributes. Kıymet et al. [19] adopted attention-LSTM to realize tourism demand forecasting, the use of DNN model for feature processing can improve model performance. Wang et al. [39] proposed a time series forecasting scheme based on a multivariate grey model and uses artificial fish swarm algorithm to optimize the settings. The model could obtain better performances in price predictions of products such as orange juice. Du et al. [9] proposed BODE method combines 10 disparate model candidates, including the latest DNNs to improve the performance of time series predictions. The model was also evaluated on the prediction task of monthly commodity prices.

Although recent DNN-based research has produced great improvements in the task of time series predictions, it seldom considers the unique characteristics of sales time series comprehensively (Sales prediction is often taken as a subtask in their research), which may limit the performances of existing solutions (As is introduced in subsection 1.2). For the first limitation, which does not consider different time series type, we design different sub-models to investigate the influences of different time series type. For the second limitation, which does not consider extracting time series with stable linear correlations from MTS, we design a dynamic co-integration (DCI) component to solve the problem. For the third limitation, which is that at future prediction time $t + 1$, important factors are unknown, and it is hard to model the future vision-based dependent correlations from the dynamic, changeable sales MTS. We design a novel TLCNN-TSLSTM framework to solve the problem to a certain extent. For the last limitation, which indicates that sales prediction is significantly influenced by periodic factors and important promotional activities. The periodic and promotional influence are hard to learn without prior knowledge, because of noisy information from other feature time series. We have therefore designed the prior knowledge (PK) component to enhance the influence of domain knowledge towards sales predictions, and the knowledge can be taken as the supervision information to guide the model training.

4. Problem definition

Assume the sales time series of a product P (SKU) is $Y = \{Y_0, Y_1, \cdots, Y_t\}$, where Y_i is the sales at the i th ($i \in [0, t]$) time. $\widetilde{Y} = \left\{Y^1, Y^2, \cdots, Y^M\right\}$ is the set of time series of M products with the same type of P (for example: Microwave Ovens with different model numbers) and in the same warehouse with P. For each Y^j in $\widetilde{Y}(j \leq \mathrm{M})$, $Y^j = \left\{Y^j_0, Y^j_1, \cdots, Y^j_t\right\}$ is the sales time series of the jth product in set $\widetilde{Y}$. For example, Y^M_t indicates the sales of the Mth product at time t. $X = \left\{X^1, X^2, \cdots, X^N\right\}$ is the set of N time series of marketing features (such as promotions, users' visiting records etc) towards target time series Y, $X^k = \left\{X^k_0, X^k_1, \cdots, X^k_t\right\}$ is the kth time series of set X, where $k \leq N$. X and $\widetilde{Y}$ are also taken as feature sets of Y. The objective function of target product P could be described as: $\widehat{Y}_{t+1} = f\left(Y, \widetilde{Y}, X\right)$, where f is the proposed model SLST-PKNet, and $\widehat{Y}_{t+1}$ is the predicted sales of product P at time $t + 1$.

Another important issue which we should emphasize is that the division method of time is mainly based on the demands of manufacturing enterprises. In general, they assign two weeks as a time period, and would like to predict products' sales over the following fortnight. This is illustrated in Fig. 2. The standard practice is to divide a month into two time periods. The maximum range *Rmax* is used to capture dynamic co-integration (DCI) based stationary correlations, while the minimum range *Rmin* is for a neural network model to detect short-term complex patterns for predicting Y_{t+1}. The rest of the time series is defined as long-term parts.

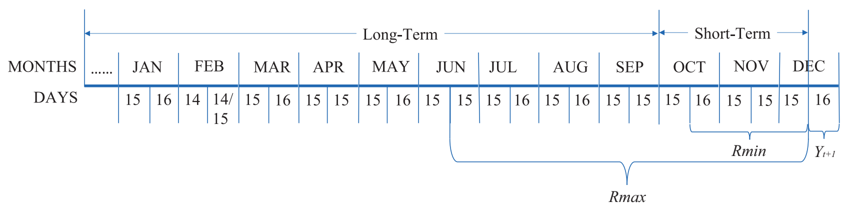


Fig. 2. Time divisions used in time series for model training and testing.

The main differences between the proposed model and the existing long short-term model can be summarized as follows. First, the model takes the influence of different time series types into considerations. Second, the model uses DCI to capture stable linear correlations from MTS. Third, the proposed long short-term model is based on a combination of TLCNN and TSLSTM. This combination can model future unknown dependent correlations, it can provide dynamic weight to each time point of the sales sequence by controlling the weight of each stage's input of TSLSTM. Fourth, the incorporation of prior knowledge (PK) helps by capturing long short-term patterns. PK is of two different types: seasonal influence (SI) and promotional influence (PI). Inclusion of a PK component allowed the impact of SI and PI to be incorporated in long short-term modeling.

5. Model description

The framework of the proposed SLST-PKNet is shown below in Fig. 3. The input Multivariate Time Series (MTS) consists of three types of time series: $Y, \widetilde{Y}, X$. The SLST-PKNet model comprises three sub-models, which model three types of correlation: $\left\{Y, \widetilde{Y}\right\}$, $\{Y,X\}$ and $\left\{Y, \widetilde{Y}, X\right\}$.

$\left\{Y, \widetilde{Y}\right\}$ captures temporal patterns of Y based on a set of time series, $\widetilde{Y}$, all of which have the same product type. Each Y^i in $\widetilde{Y}$ has the same product type as Y, and both Y^i and Y are in the same warehouse. Thus, according to consumer behavior theory [10,19,34], the correlation patterns show common sales trends and competitive relationships between time series in $\left\{Y, \widetilde{Y}\right\}$. This design is similar to those of previous studies, which group time series of the same type (such as traffic flow, electricity consumption), and take the group as the input of MTS prediction.

$\{Y,X\}$ captures the influence of the marketing feature set X on the target time series Y. The X of a product includes price, promotion, users' visiting records, etc. According to consumer behavior theory [10,19,34], the marketing features of a product have a significant influence on its future sales trends.

$\left\{Y, \widetilde{Y}, X\right\}$ emulates the combined influences of $\widetilde{Y}$ and X on Y. According to research into feature interactions and combinations [15,34], the inner-correlations between features with different types will make positive contributions towards sales predictions. However, combining different types of features will also interfere with the accuracy of sales prediction. So we design sub-models $\left\{Y, \widetilde{Y}\right\}$ and $\{Y,X\}$ to reduce the interference.

To simplify the description, we use a unified method to represent the three different sub-models: $S = \{Y, \Phi\{\varphi_1, \varphi_2, \cdots, \varphi_L\}\}$, where $\Phi \in \left\{\widetilde{Y}, X, X\widetilde{Y}\right\}$. φ_i is the ith time series of Φ. For example, if $\Phi = \widetilde{Y}$, then $\varphi_i = \widetilde{Y}_i$. For each sub-model, the next step uses Dynamic Co-Integration (DCI) to calculate the stationary short-term influence range for capturing strong temporal linear patterns. Meanwhile, a two-layer convolutional neural network (TLCNN) with dropout [1,5] models local correlations between different time series at time t (C_t) and t + 1 (C_{t+1}). The outputs C_t and C_{t+1} become the inputs for 2 two-stage.

LSTM (TSLSTM) models. These are a short-term two-stage LSTM (ST-TSLSTM) and a long-term two-stage LSTM (LT-TSLSTM). ST-TSLSTM models correlations of short-term temporal patterns, and $h^{2,ST}_{t+1}(Y,\Phi)$ is the hidden state at t + 1 of the second-stage of ST-TSLSTM. LT-TSLSTM uses prior knowledge to model correlations of long-term temporal patterns, and $h^{2,LT}_{t+1}(Y,\Phi)$ is the hidden state at t + 1 of the second-stage of LT-TSLSTM. Finally, outputs are taken from the following sub-models:

Outputs from $\left\{\boldsymbol{Y}, \widetilde{\boldsymbol{Y}}\right\}$: $\widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right) = \mathrm{REG}_1\left(h^{2,ST}_{t+1}\left(Y, \widetilde{Y}\right), h^{2,LT}_{t+1}\left(Y, \widetilde{Y}\right)\right)$.

Outputs from $\{\boldsymbol{Y}, \boldsymbol{X}\}$: $\widehat{Y}_{t+1}(Y,X) = \mathrm{REG}_2\left(h^{2,ST}_{t+1}(Y,X), h^{2,LT}_{t+1}(Y,X)\right)$.

Outputs from $\left\{\boldsymbol{Y}, \widetilde{\boldsymbol{Y}}, \boldsymbol{X}\right\}$: $\widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right) = \mathrm{REG}_3\left(h^{2,ST}_{t+1}\left(Y, X\widetilde{Y}\right), h^{2,LT}_{t+1}\left(Y, X\widetilde{Y}\right)\right)$.

$$\widehat{Y}^{\mathrm{DCI}}_{t+1} = \mathrm{VAR}\left(\mathrm{DCI}\left(Y, X, \widetilde{Y}\right)\right)$$

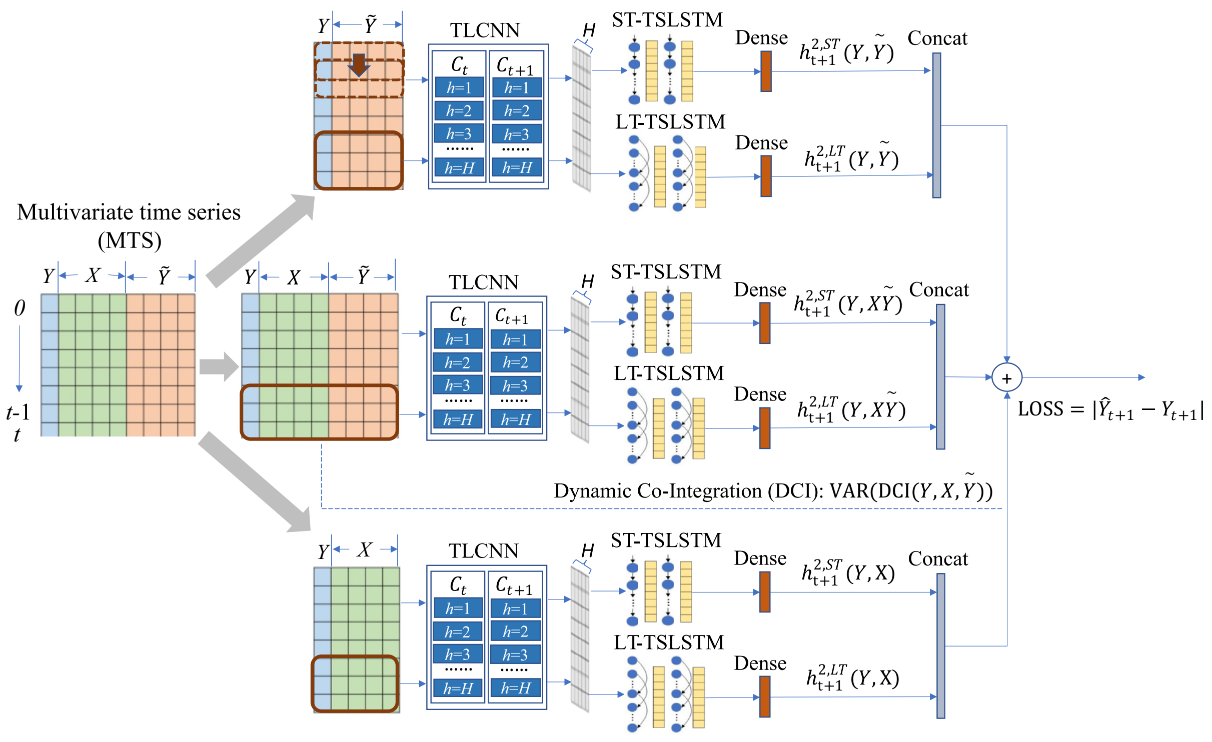


Fig. 3. The framework of the proposed SLST-PKNet model.

where REG_1, REG_2 and REG_3 represent three regression models. The outputs of the models are the predicted sales at time $t+1$, which are $\widehat{Y}_{t+1}\left(Y,\widetilde{Y}\right)$, $\widehat{Y}_{t+1}(Y,X)$ and $\widehat{Y}_{t+1}\left(Y,X\widetilde{Y}\right)$. The output of DCI is the set of selected time series with strong linear correlations from $\left\{Y,\widetilde{Y},X\right\}$, and is used as the input of a vector auto-regression based machine learning (VAR) model to predict sales $\widehat{Y}^{\mathrm{DCI}}_{t+1}$ at time $t+1$. Thus, the final predicted sales $\widehat{Y}_{t+1}$ can be represented as the weight-sum of all predicted sales from different components. $\widehat{Y}_{t+1}=F\left(\widehat{Y}_{t+1}\left(Y,\widetilde{Y}\right),\widehat{Y}_{t+1}(Y,X),\widehat{Y}_{t+1}\left(Y,X\widetilde{Y}\right),\widehat{Y}^{\mathrm{DCI}}_{t+1}\right)$, where F is the weight-sum function. The following content will introduce each component in detail.

5.1. Notation explanation

Table 1 summarizes important notations in the manuscript. The notations are divided into three parts: Input, Model and Output. Input describes the mathematical representation of input data. Model describes the mathematical representations of the main variables and components of the proposed SLST-PKNet model, which includes DCI component, TLCNN, TSLSTM, PK component, ST-TSLSTM and LT-TSLSTM. Output describes the mathematical representations of the model output.

5.2. Dynamic co-integration (DCI)

Existing DNN-based methods emphasize the ability to detect non-linear and complex correlations, but neglect strong linear patterns derived from sales time series. More importantly, existing methods tend to use auto-regression (AR) to model linear correlations of the target time series itself, but seldom consider correlations between target and feature time series. To address these limitations, we propose a DCI mechanism that assigns dynamic ranges for each prediction task, and focuses on detecting linear correlations between time series from different dynamic ranges. More importantly, the mechanism helps balance the performance between stationary linear correlations and complex non-linear correlations.

Table 1
Notations.

Type	Notation	Explain
INPUT	$Y = \{Y_0, Y_1, \cdots, Y_t\}$	The sales time series of a target product P from time 0 to t. The task is to predict its sales at time $t + 1$.
	$\widetilde{Y} = \left\{Y^1, Y^2, \cdots, Y^M\right\}$	The set of time series of M products of the same product type as P, and in the same warehouse as P.
	$Y^j = \left\{Y_0^j, Y_1^j, \cdots, Y_t^j\right\}$ in $\widetilde{Y}\,(j \le M)$	The sales time series of the jth product in set $\widetilde{Y}$.
	$X = \left\{X^1, X^2, \cdots, X^N\right\}$	The set of N time series of marketing features.
	$X^k = \left\{X_0^k, X_1^k, \cdots, X_t^k\right\}$ in $X(k \le \mathrm{N})$	The time series of the kth feature in set X.
	$\Phi = \{\varphi_1, \varphi_2, \cdots, \varphi_L\}$, $\Phi \in \left\{\widetilde{Y}, X, X\widetilde{Y}\right\}$	$\Phi \in \left\{\widetilde{Y}, X, X\widetilde{Y}\right\}$. The set Φ can represent any of the three sets of $\widetilde{Y}$, X and $X\widetilde{Y}$. φ_i is the ith time series of Φ.
	$S = \{Y, \Phi = \{\varphi_1, \varphi_2, \cdots, \varphi_L\}\}$	The set S of time series, includes both Y and its related Φ.
MODEL	**DCI component:** $\bar{S} \in S = \left\{Y, X\widetilde{Y}\right\}$	$\bar{S}$ is a subset of S. Any two time series in $\bar{S}$ can pass the co-integration test, which indicates that there exist stable linear correlations between the two time series.
	DCI component: *Rmax*	*Rmax* and time step t define which part (t – *Rmax* – t) of the whole time series should be processed by the DCI component.
	DCI component: $\mathrm{VAR}(\mathrm{DCI}(\cdot))$	DCI is used to detect subset $\bar{S}$ from S. VAR is used to build regression models based on all time series from $\bar{S}$.
	TLCNN: F_1 and F_2; $C^1_{0\ t}$ and $C^2_{1\ t+1}$	In formula (2), F_1 is the 1st convolutional layer that captures dependent correlations $C^1_{0\ t}$ from input S. F_2 is the 2nd convolutional layer that captures and infers dependent correlations $C^2_{1\ t+1}$ from $C^1_{0\ t}$.
	TSLSTM: $h^1_{R_a\ R_b}$ and $h^2_{R_a+1\ R_b+1}$	$R_a\ R_b$ is defined as a time segment on a 0 – t timeline. The start and end time are R_a and R_b. $h^1_{R_a\ R_b}$ is the 1st stage of TSLSTM, and takes $C^1_{R_a\ R_b}$ as input to encode the state of MTS from R_a to R_b. $h^2_{R_a+1\ R_b+1}$ is the 2nd stage of TSLSTM, and takes $C^2_{R_a+1\ R_b+1}$ as input to infer the state of MTS from $R_a + 1$ to $R_b + 1$ based on $h^1_{R_a\ R_b}$.
	TSLSTM: $h^1_{R_b} = \sum_{i=R_a}^{R_b} \alpha_{1i} \cdot h^1_i$	In formula (4), $h^1_{R_b}$ is the hidden state of the 1st stage of TSLSTM at time R_b. α_{1i} is the attention weight of the ith time of the 1st stage of TSLSTM.
	TSLSTM: $h^2_{R_b+1} = \sum_{i=R_a+1}^{R_b+1} \alpha_{2i} \cdot h^2_i$	In formula (4), $h^2_{R_b+1}$ is the hidden state of the 2nd stage of TSLSTM at time R_{b+1}. α_{2i} is the attention weight of the ith time of the 2nd stage of TSLSTM.
	PK component: $h^{2,\mathrm{SI}}_{R_b+1}(S, d = T)$	In formula (5), $h^{2,\mathrm{SI}}_{R_b+1}$ is the 2nd stage hidden state of TSLSTM at time $R_b + 1$ based on seasonal influence (SI). T is a seasonal cycle.
	PK component: $h^{2,\mathrm{PI}}_{R_b+1}(S, d = R_b + 1 - tp)$	In formula (6), $h^{2,\mathrm{PI}}_{R_b+1}$ is the 2nd stage hidden state of TSLSTM at time $R_b + 1$ based on promotional influence (PI). tp is a fixed promotion time.
	PK component: $h^{2,\mathrm{PK}}_{R_b+1}(S)$	In formula (7), $h^{2,\mathrm{PK}}_{R_b+1}(S)$ is the hidden state of PK component at time $R_b + 1$, and is a combination of both $h^{2,\mathrm{SI}}_{R_b+1}$ and $h^{2,\mathrm{PI}}_{R_b+1}$.
	ST-TSLSTM: $h^{2,\mathrm{ST}}_{t+1}(S, t - \mathrm{Rmin}\ t)$	For a TSLSTM, ST-TSLSTM is defined as $R_a = t - Rmin$ and $R_b = t$, where *Rmin* is a constant to determine the range of short-term (ST).
	LT-TSLSTM: $h^{2,\mathrm{LT}}_{t+1}(S)$	For a TSLSTM, LT-TSLSTM is defined as $R_a = 0$ and $R_b = t$. In addition, the hidden state $h^{2,\mathrm{PK}}_{t+1}(S)$ of PK component at time $t + 1$ is also integrated to model the seasonal and promotional influences from a long-term (LT) perspective. A detailed description is given in formula (8).
Type	**Notation**	**Explain**
OUTPUT	$\widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right)$	$\widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right)$ uses $\widetilde{Y}$ to predict $\widehat{Y}_{t+1}$ of Y based on both ST-TSLSTM and LT-TSLSTM.
	$\widehat{Y}_{t+1}(Y, X)$	$\widehat{Y}_{t+1}(Y, X)$ uses X to predict $\widehat{Y}_{t+1}$ of Y based on both ST-TSLSTM and LT-TSLSTM.
	$\widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right)$	$\widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right)$ uses $X\widetilde{Y}$ to predict $\widehat{Y}_{t+1}$ of Y based on both ST-TSLSTM and LT-TSLSTM.
	$\widehat{Y}^{\mathrm{DCI}}_{t+1}$	$\widehat{Y}^{\mathrm{DCI}}_{t+1}$ uses $\{Y, X, \widetilde{Y}\}$ to predict the value of Y at time t+1 based on a DCI component.
	$\widehat{Y}_{t+1}$	$\widehat{Y}_{t+1}$ is the final prediction of Y at time t+1 by integrating $\widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right)$, $\widehat{Y}_{t+1}(Y, X)$, $\widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right)$ and $\widehat{Y}^{\mathrm{DCI}}_{t+1}$.

Algorithm 1 Description of Dynamic Co-Integration (DCI).

```
1. # INPUT: A set of time series: $S = \{Y, \varphi_1, \varphi_2, \varphi_3, \cdots, \varphi_L\}$, where $\varphi_i \epsilon X \cup \widetilde{Y}$.
      Time step: $t$. Max range: $Rmax$. Min short-term range: $Rmin$.
2. # OUTPUT: $O$ = [$\left\{Y, \bar{S}\right\}_k$], where $\bar{S}$ is subset of $S$ and $Y \notin \bar{S}$.
3. Set $O$, $\bar{S}$ as empty sets.
4. Function ciTest ($\{Y, \bar{S}\}$):
5.     Return $\{Y, \bar{S}\}$ If each $\bar{\varphi}_i$ in $\bar{S}$ can pass co-Integration test with $Y$.
```

```
6.       Else delete non-significant variables from $\bar{S}$ and Return {$Y$, $\bar{S}$}.
7.   MAIN:
8.   If $Y(t - Rmax\ t)$'s one-order sequence is not stationary:
9.     Return $O$ = [].
10.    For each $\varphi_i$ in $S$: # Stationarity Test for each $\varphi_i$.
11.      $\bar{S}$. append ($\varphi_i$) if $\varphi_i(t - Rmax\ t)$ is stationary or first-order stationary.
12.    Let $k$ = $Rmax$.
13.    While $k$ > 15:
14.      $O$.append (ciTest({$Y(t - k - t)$, $\bar{S}(t - k - t)$})).
15.      $k$ --.
16.    Return $O$.
```

A description of DCI is presented in Algorithm 1, where the input is described in line 1. The set of time series S includes both target time series Y and its related set of time series $\varphi_1, \varphi_2, \varphi_3, \cdots, \varphi_L$ with length as L, where $\varphi_i \epsilon X \cup \widetilde{Y}$, which means that φ_i may belong to the set of X or $\widetilde{Y}$. Time step t and Rmax define which part (t–$Rmax$ – t) of the whole time series should be processed by the DCI component. Rmin is the minimum time series range assignment, and is mainly taken as the input of the DNN for short-term non-linear patterns detection. The output O is a list containing a set of $\left\{Y, \bar{S}\right\}_k$. In each $\left\{Y, \bar{S}\right\}_k$, $\bar{S}$ is a set of selected time series from S, each of which can pass the co-integration test with Y, which indicates the presence of strong linear correlation patterns in the set of $\left\{Y, \bar{S}\right\}_k$. k means that the linear correlation is limited to t-k – t section, while k < $Rmax$. Lines 4 – 6 define a function, ciTest, which recursively detects $\left\{Y, \bar{S}\right\}$ from the original *{Y, S}* by using a co-Integration test. The main function is in Lines 7 – 16. Lines 10 – 11 test whether all the original time series from *{Y, X, $\widetilde{Y}$}* are stationary or first-order stationary. This operation is a necessary condition for the subsequent co-integration test. Lines 13 – 16 recursively detects all $\left\{Y, \bar{S}\right\}_k$, where k *>= 15* and k *<= Rmax*. When k is reduced from *Rmax* to 15, the detected linear correlations are stronger. The reason for assigning k>=15 is that the k of more than 80 % $\left\{Y, \bar{S}\right\}_k$ is greater than 15. Although a smaller k will obtain more optimal subsets, resulting in the detection of stronger correlations, their contributions are limited in the experiment, because most of the correlations can be explained by subsets with a bigger k value. A smaller k means fewer samples, which will result in lower confidence levels.

The output of Algorithm 1 helps to capture strong linear correlations between Y and selected time series. For each selected subset $\left\{Y, \bar{S}\right\}_k$ of output O, a VAR-based (vector auto-regression) machine learning model is adopted. Thus, $\mathrm{VAR}\left(\mathrm{DCI}\left(Y, X\widetilde{Y}\right)\right)$ is represented by formula (1).

$$\widehat{Y}^{\mathrm{DCI}}_{t+1} = \mathrm{VAR}\left(\mathrm{DCI}\left(Y, X\widetilde{Y}\right)\right) = \sum_{i=1}^{Len(O)} W_i^{\mathrm{DCI}} \times \mathrm{VAR}\left(\left\{Y(t-i\ t), \bar{S}(t-i\ t)\right\}_i\right) \tag{1}$$

where W_i^{DCI} is the weight matrix. The smaller i indicates that there exist stronger linear correlations in subset $\left\{Y, \bar{S}\right\}_i$. $\widehat{Y}^{\mathrm{DCI}}_{t+1}$ is the predicted sales of target time series Y at time t + 1.

5.3. Two-layer convolutional neural network (TLCNN)

The main aim of TLCNN is to learn the local dependencies between variables at time t and to infer dependencies at time t + 1. For all three time series combinations $\{Y, \widetilde{Y}\}$, $\{Y, X\}$ and $\{Y, \widetilde{Y}, X\}$, we use $S = \{Y,\ \Phi\{\varphi_1, \varphi_2, \cdots, \varphi_L\}\}$, where $\Phi \in \left\{\widetilde{Y}, X, X\widetilde{Y}\right\}$ to represent them in a unified way. Then the TLCNN output at time t and t + 1 can be defined as:

$$C_t^1 = F_1(S(0\ t)) = Stack_{h=1}^{H}\left(\sigma\left(\mathrm{W}_t^{1,h} * S(0\ t) + b_t^{1,h}\right)\right).F_1 : \mathbb{R}^{t \times L} \to \mathbb{R}^{t \times H}$$

$$C_{t+1}^2 = F_2\left(C_t^1\right) = Stack_{h=1}^{H}\left(\sigma\left(\mathrm{W}_{t+1}^{2,h} * C_t^1 + b_{t+1}^{2,h}\right)\right).F_2 : \mathbb{R}^{t \times H} \to \mathbb{R}^{t \times H} \tag{2}$$

where F_1 and F_2 are convolutional layers with H filters. 0 – t is the length of time series; C_t^1 is the representation of S's local dependencies at time t; C_{t+1}^2 is the inferred representation at time t + 1, which is mainly used for predictive tasks at time t + 1. The dropout mechanism [1,5] is also used on each layer's output to avoid over-fitting and to obtain more steady results. $Stack_{h=1}^{H}\left(\sigma\left(\mathrm{W}_t^{1,h} * S(0\ t) + b_t^{1,h}\right)\right)$ and $Stack_{h=1}^{H}\left(\sigma\left(\mathrm{W}_{t+1}^{2,h} * C_t^1 + b_{t+1}^{2,h}\right)\right)$ are formula expressions of F_1 and F_2 [5,20]. $\mathrm{W}_t^{1,h}$ and $\mathrm{W}_{t+1}^{2,h}$ are the weight matrix of the hth filter of F_1 and F_2. $b_t^{1,h}$ and $b_{t+1}^{2,h}$ are the bias items, and σ is an activation function, which is empirically assigned as LeakyReLU. Above all, for a whole time span (0 – t) of a MTS S with length L, the output of TLCNN can be represented as: $C^1_{0\ t} = \left\{C^1_0, C^1_1, \cdots, C^1_t\right\}$ and $C^2_{1\ t+1} = \left\{C^2_1, C^2_2, \cdots, C^2_{t+1}\right\}$.

5.4. Two-stage LSTM (TSLSTM)

The outputs of TLCNN, which are $C^1_{0\ t}$ and $C^2_{1\ t+1}$, are then fed into a TSLSTM to capture sequence correlation patterns. Assuming a time range of R_a to R_b (Fig. 4), the outputs $C^1_{R_a\ R_b}$ and $C^2_{R_a+1\ R_b+1}$ will be fed separately into the two-stage LSTM ($h^1_{R_a\ R_b}$ and $h^2_{R_a+1\ R_b+1}$). The purpose of this two-stage design is to predict the dependent correlations between future unknown influence variables (such as different promotion strategies at future time t + 1) and then to use the predicted correlations to forecast future sales at time t + 1. Following on from this, $C^1_{R_a\ R_b}$ is fed into the first stage LSTM ($h^1_{R_a\ R_b}$) to model the variable's influence on current sales at the same time step (R_a, $R_a + 1, \ldots, R_b$). The output of stage 1, which is $\left\{h^1_{R_a}, h^1_{R_a+1}, \cdots, h^1_{R_b}\right\}$, is then fed into a one-layer dense network to fit a loss function $\mathrm{LOSS}\left(\widehat{Y}^{h_1}_{R_b}, Y_{R_b}\right) = |\widehat{Y}^{h_1}_{R_b} - Y_{R_b}|$.

The final output $h^1_{R_b}$ of the first-stage is taken as the input of the second-stage LSTM$\left(h^2_{R_{a+1}\ R_{b+1}}\right)$. In this stage, values for C^2_{i+1} (mainly inferred from C^1_i where $i \in [R_a, R_b]$), are fed into each state h^2_{i+1} to capture the future influence of all variables at time i + 1. The output of stage 2, which is $\left\{h^2_{R_a+1}, h^2_{R_a+2}, \cdots, h^2_{R_b+1}\right\}$, is fed into another one-layer dense network with loss function $\mathrm{LOSS}\left(\widehat{Y}^{h_2}_{R_b+1}, Y_{R_b+1}\right) = |\widehat{Y}^{h_2}_{R_b+1} - Y_{R_b+1}|$. The hidden state at time i is h^k_i ($k \in \{1, 2\}$ represents the two stages of TSLSTM). The recurrent unit of TSLSTM includes: Input Gate in^k_i, Forget Gate f^k_i, Cell Status $cell^k_i$, Output Gate out^k_i, and the output hidden state h^k_i. $\mathrm{W}_{in,k}$, $\mathrm{W}_{c,k}$,$\mathrm{W}_{f,k}$ and $\mathrm{W}_{o,k}$ are the parameter matrices. σ is the sigmoid function and $\odot$ is the element-wise product.

Input Gate: $in^k_i = \sigma\left(\mathrm{W}_{in,k} \cdot \left[h^k_{i-1}, add\left(C^j_i\right)\right] + b_{in,k}\right)$, $where\, j \le k$.

Forget Gate: $f^k_i = \sigma\left(\mathrm{W}_{f,k} \cdot \left[h^k_{i-1}, add\left(C^j_i\right)\right] + b_{f,k}\right)$, $where\, j \le k$.

Cell Status:$c^k_i = \tanh\left(\mathrm{W}_{c,k} \cdot \left[h^k_{i-1}, add\left(C^j_i\right)\right] + b_{c,k}\right)$, $where\, j \le k$.

$cell^k_i = in^k_i \odot c^k_i + f^k_i \odot cell^k_{i-1}$.

Output Gate: $out^k_i = \sigma\left(\mathrm{W}_{o,k} \cdot \left[h^k_{i-1}, add\left(C^j_i\right)\right] + b_{o,k}\right)$, $where\, j \le k$.

$$\textbf{\textit{Hidden State}} : h^k_i = out^k_i \odot \tanh\left(cell^k_i\right) \tag{3}$$

Unlike the traditional CNN-LSTM model, which mainly adopts a weight-sharing strategy, an *add* function is designed. When k = 1, $add\left(C^j_i\right)$ is equal to C^1_i; when k = 2, $add\left(C^j_i\right)$ is $\omega_1 \times C^1_i + \omega_2 \times C^2_i$, where ω_1 and ω_2 are learnable parameters. This operation could provide additional useful information from the first-stage to the second-stage for capturing dynamic changing patterns by fusing C^1_i and C^2_i. An attention mechanism, which can improve capture of sequence correlations, is also incorporated into the TSLSTM. The final output of hidden state $h^k_{R_b}$ can be seen in formula (4).

$$h^1_{R_b} = \sum_{i=R_a}^{R_b} \alpha_{1i} \cdot h^1_i, \alpha_{1i} = \frac{\exp sim\left(h^1_i, h^1_{R_b}\right)}{\sum_{j=R_a}^{R_b} \exp sim\left(h^1_j, h^1_{R_b}\right)}$$

$$h^2_{R_b+1} = \sum_{i=R_a+1}^{R_b+1} \alpha_{2i} \cdot h^2_i, \alpha_{2i} = \frac{\exp sim\left(h^2_i, h^2_{R_b+1}\right)}{\sum_{j=R_a+1}^{R_b+1} \exp sim\left(h^2_j, h^2_{R_b+1}\right)} \tag{4}$$

where α_{1i} and α_{2i} are attention weights of h^1_i and h^2_i at time i; and *sim* is used to calculate similarity between the two hidden states by using dot production. Compared with traditional methods, such as DARNN, MTNet, MLCNN, TSLSTM has unique advantages: first, TSLSTM uses $C^2_{R_a+1\ R_b+1}$ to model future dependent correlations between variables, which is more practical when applied to sales prediction; second, the add function of TSLSTM uses two parameters ω_1 and ω_2 to combine C^1_i and C^2_i at the second-stage LSTM. It can provide more useful information for future trend predictions, and the dynamic ω_1 and ω_2 assignment can overcome the limitations of a traditional weight-sharing strategy of CNN-LSTM, enabling it to capture more dynamic non-linear patterns.

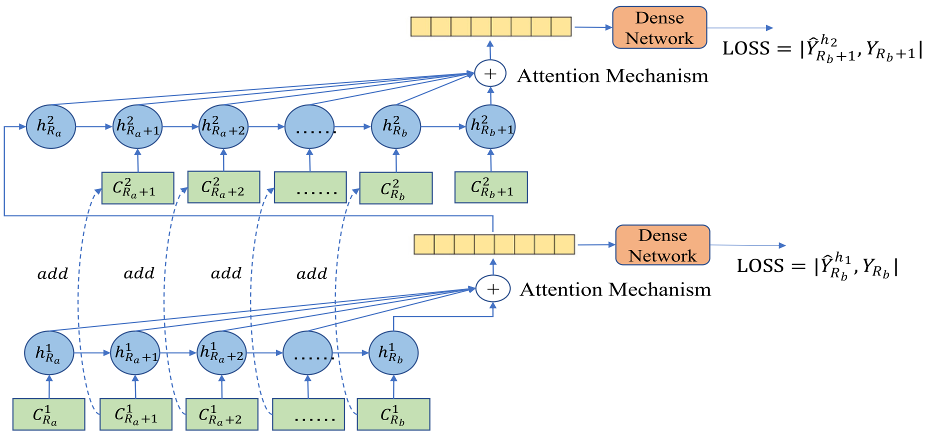


Fig. 4. The framework of TSLSTM component.

5.5. Prior knowledge (PK) component

Prior knowledge is defined as well-known domain specific experience or knowledge. The PK component incorporates prior knowledge into TSLSTM to improve sales prediction. Prior knowledge consists of two types: seasonal or cyclical influences (SI) and the influence of fixed annual promotional times (PI). SI is straightforward: sales of a product in a specific month are correlated to sales in the same month the following year (cycle interval T is approximately-one year). PI captures the influence of important promotional activities. Experts can determine the future sales potential of a product by analyzing its sales performances at fixed promotional times. Such times typically include important holidays, and promotional festivals in e-commerce (such as 11th Nov in China). If a product sells good over a fixed period, then there is likely to be another sales peak a few months later, so predictions of the peak value should not be based solely on promotional strategies, prior knowledge is also important.

PK component uses TSLSTM to construct new models for these knowledge factors. The model framework is shown in Fig. 5. The input includes target sales time series Y and a set of related time series $\Phi = \{\varphi_1, \varphi_2, \cdots, \varphi_L\}$, $\Phi \in \left\{X, \widetilde{Y}\right\}$. Assuming that the time section is still from R_a to R_b, then in order to predict future sales Y_{R_b+1}, the $S = \left\{Y_{R_a\ R_b}, \Phi_{R_a\ R_b}\right\}$ is fed into TLCNN to obtain a new MTS with information from local dependent variables. The new MTS is then processed by TSLSTM, which captures the correlation patterns of (Y_{R_b+1}, SI), (Y_{R_b+1}, PI) separately. PK component consists of three TSLSTM, which output three states: $h^2_{R_b+1}(S, R_a\ R_b)$, $h^{2,\mathrm{SI}}_{R_b+1}$ and $h^{2,\mathrm{PI}}_{R_b+1}$ separately. $h^2_{R_b+1}(S, R_a\ R_b)$ means to use TSLSTM to encode multivariate time series (MTS) S from start time R_a to end time R_b, and the superscript 2 of h indicates that $h^2_{R_b+1}(S, R_a\ R_b)$ is the output of the second stage of TSLSTM. $h^{2,\mathrm{SI}}_{R_b+1}$ is the output of seasonal or cyclical influences (SI), which mainly uses periodic patterns to represent the states of S at time $R_b + 1$, and can be used to capture correlation patterns between Y_{R_b+1} and SI. $h^{2,\mathrm{PI}}_{R_b+1}$ is the output of the influence of important promotional activities (PI), which mainly uses the influence patterns of promotional activities to represent the states of S at time $R_b + 1$, and can be used to capture correlation patterns between Y_{R_b+1} and PI. $h^2_{R_b+1}$ is taken as the initial model for fine-tuning of $h^{2,\mathrm{SI}}_{R_b+1}$ and $h^{2,\mathrm{PI}}_{R_b+1}$. The integration of $h^{2,\mathrm{SI}}_{R_b+1}$ and $h^{2,\mathrm{PI}}_{R_b+1}$ will generate $h^{2,\mathrm{PK}}_{R_b+1}$, which takes the prior knowledge of both PI and SI into consideration.

To model the influence of SI and PI towards Y_{R_b+1}, the first step is to construct the time range of SI and PI. The method for doing so is shown in Fig. 6.

Here, d represents the distance between two time periods, and T (where T = 24) represents a seasonal cycle, making the time range of SI: t + 1 – T – *1* – t + 1 – T + 1. Time period (tp) contains all the fixed promotion days in a year. Sales performance of a product around these days is likely to have an effect in determining sales in the subsequent months, so the time range of PI is set as tp – 1 – tp + 1. Two new models are constructed based on these time range settings: $\left[S = \{Y, \Phi\}_{R_b+1-T-1\ R_b+1-T+1}, \{Y, \Phi\}_{R_b+1}\right]$ for SI, and $\left[S = \{Y, \Phi\}_{tp-1\ tp+1}, \{Y, \Phi\}_{R_b+1}\right]$ for PI.

The two new models adopt the attention mechanism of TSLSTM separately with sine and cosine position embedding pe (pos) [38], where $pos \in [R_b + 1 - T - 1, R_b + 1 - T + 1]$ in SI and $pos \in [tp - 1, tp + 1]$ in PI. The models then obtain two hidden state representations $h^{2,\mathrm{SI}}_{R_b+1}(S, d = T)$ and $h^{2,\mathrm{PI}}_{R_b+1}(S, d = R_b + 1 - tp)$ of both SI and PI in formulas (5) and (6):

$$h^{2,\mathrm{SI}}_{R_b+1}(S, d = T) = \alpha^{\mathrm{SI}}_{R_{b+1}} \times \left(h^2_{R_b+1} + pe(R_b + 1)\right) +$$

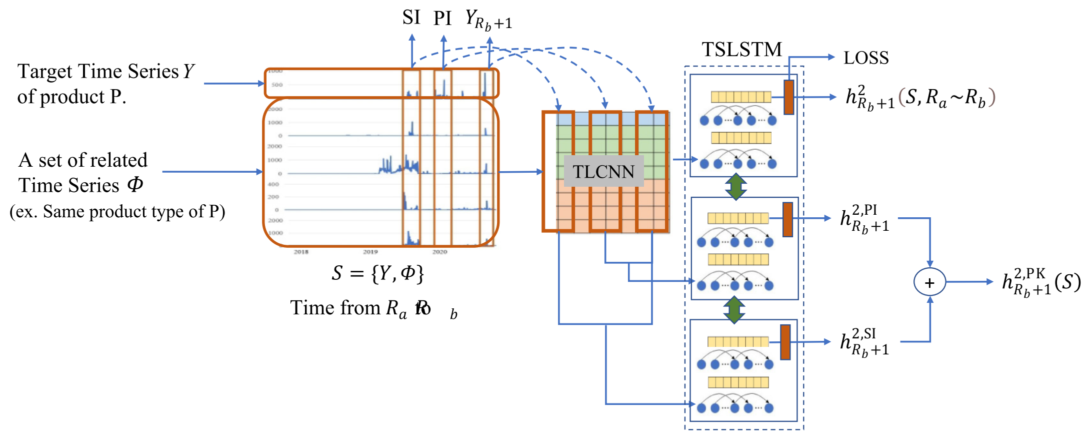


Fig. 5. The framework of PK component.

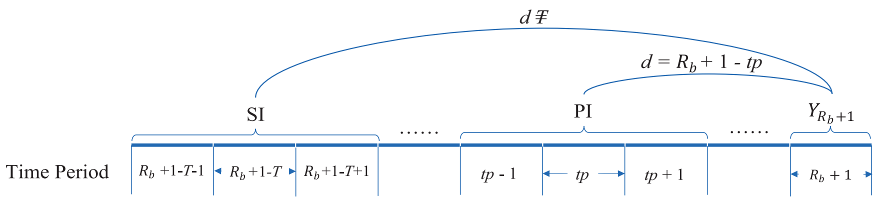


Fig. 6. Time range settings of SI and PI.

$$\sum_{i=0}^{2} \alpha_i^{SI} \times \left(h^2_{R_b+1-T-1+i} + pe(R_b + 1 - T - 1 + i)\right) \tag{5}$$

$$h^{2,PI}_{R_b+1}(S, d = R_b + 1 - tp) = \alpha^{PI}_{R_{b+1}} \times \left(h^2_{R_b+1} + pe(R_b + 1)\right) +$$

$$\sum_{i=0}^{2} \alpha_i^{PI} \times \left(h^2_{tp-1+i} + pe(tp - 1 + i)\right) \tag{6}$$

where $h^2_{R_b+1}$, $h^2_{R_b+1-T-1+i}$ and h^2_{tp-1+i} are the hidden states derived by TSLSTM at time $R_b + 1$, $R_b + 1 - T - 1 + 1$ and $tp - 1 + i$. $\alpha^{SI}_{R_{b+1}}$ and α^{SI}_i are the attention weights of SI, and $\alpha^{SI}_{R_{b+1}} + \sum_{i=0}^{2}\alpha^{SI}_i = 1$. $\alpha^{PI}_{R_{b+1}}$ and α^{PI}_i are the attention weights of PI, and $\alpha^{PI}_{R_{b+1}} + \sum_{i=0}^{2}\alpha^{PI}_i = 1$. Formulas (5) and (6) describe the dependent correlations using prior knowledge (SI and PI). Assume there are Z different tps in the MTS S (there are more than 1 fixed annual promotion times in a year), and all the outputs are integrated in formula (7) to obtain the PK representation $h^{2,PK}_{R_b+1}$ of S at time $R_b + 1$.

$$h^{2,PK}_{R_b+1}(S) = \sigma^{SI} \times h^{2,SI}_{R_b+1}(S, d = T) + \sum_{i=1}^{Z} \sigma_i^{PI} \times h^{2,PI}_{R_b+1}(S, d = R_b + 1 - tp_i) \tag{7}$$

where σ^{SI}, $\sigma^{PI}_i \in \mathbb{R}$, represent the important weight of SI $h^{2,SI}_{R_b+1}$ and the ith PI $h^{2,PI}_{R_b+1}$, the time point of which is tp_i, towards PK representation $h^{2,PK}_{R_b+1}$.

5.6. Short-term two-stage LSTM (ST-TSLSTM)

ST-TSLSTM helps to capture short-term complex and non-linear influence patterns not well handled by traditional methods. For a MTS $S = \{Y, \Phi\}$, it is assumed that the current task is to predict the sales at time $t + 1$. We then define the range of short-term as being from $R_a = t - Rmin$ to $R_b = t$, where $Rmin$ is a constant determined by experience and experiment. We then use TSLSTM to predict the future state representation $h^{2,ST}_{t+1}(S, t - Rmin\ t)$ at time $t + 1$ based on short-term MTS $S_{t-Rmin\ t}$.

5.7. Long-term two-stage LSTM (LT-TSLSTM)

LT-TSLSTM helps to capture the long-term complex and non-linear influence patterns in predicting future sales. For a MTS $S = \{Y, \Phi\}$, the time range is the whole time series, from $R_a = 0$ to $R_b = t$. We can use TSLSTM to obtain the hidden state representation of each time point as: $h_0^2, h_1^2, \cdots h_t^2, h_{t+1}^2$. For prior knowledge SI, we can obtain its hidden state $h_{t+1}^{2,\mathrm{SI}}(S, d = T)$ by using the PK component in formula (5), where T = 24 represents a cycle length of one year. For prior knowledge PI, we can obtain all its hidden states $h_{t+1}^{2,\mathrm{PI}}(S, d = t + 1 - tp_i)$ in formula (6). tp_i is the ith fixed promotion time, and tp_i satisfies $Rmin < t + 1 - tp_i < 24$. Consequently, a time point tp_i which is too close (within short-term) or too far away (above 1 year) from t + 1 is not considered by PI. According to formula (7), the final hidden state representation of PK component at time t + 1 is $h_{t+1}^{2,\mathrm{PK}}$. Therefore, the future hidden state $h_{t+1}^{2,\mathrm{LT}}$ of LT-TSLSTM can be represented as:

$$h_{t+1}^{2,\mathrm{LT}}(S) = \theta_1 \times h_{t+1}^{2}(S, 0\ t) + \theta_2 \times h_{t+1}^{2,\mathrm{PK}}(S) \tag{8}$$

where θ_1, θ_2 are weight parameters, which indicate the importance of $h_{t+1}^{2}(0\ t)$ and $h_{t+1}^{2,\mathrm{PK}}$.

5.8. Generating the predictions

The proposed SLST-PKNet is divided into three sub-models: $\{Y, X\}$, $\left\{Y, \widetilde{Y}\right\}$ and $\left\{Y, X\widetilde{Y}\right\}$, where Y is the target time series and Y_{t+1} is the sales to be predicted. For each sub-model, ST-TSLSTM and LT-TSLSTM are adopted to capture long short-term dependencies, the formulas of which are as below:

$$\widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right) = \mathrm{W}_{\widetilde{Y}} \cdot \mathrm{contact}\left[h_{t+1}^{2,\mathrm{ST}}\left(Y, \widetilde{Y}\right); h_{t+1}^{2,\mathrm{LT}}\left(Y, \widetilde{Y}\right)\right] + b_{\widetilde{Y}}$$

$$\widehat{Y}_{t+1}(Y, X) = \mathrm{W}_X \cdot \mathrm{contact}\left[h_{t+1}^{2,\mathrm{ST}}(Y, X); h_{t+1}^{2,\mathrm{LT}}(Y, X)\right] + b_X$$

$$\widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right) = \mathrm{W}_{\widetilde{XY}} \cdot \mathrm{contact}\left[h_{t+1}^{2,\mathrm{ST}}\left(Y, X\widetilde{Y}\right); h_{t+1}^{2,\mathrm{LT}}\left(Y, X\widetilde{Y}\right)\right] + b_{\widetilde{XY}} \tag{9}$$

where $\mathrm{W}_{\widetilde{Y}}, \mathrm{W}_X, \mathrm{W}_{\widetilde{XY}} \in \mathbb{R}^{1\times 2K}$ are learnable vectors, K is the dimension length of TSLSTM, and $b_{\widetilde{Y}}$, b_X, $b_{\widetilde{XY}} \in \mathbb{R}$. $h_{t+1}^{2,\mathrm{ST}}(S)$ and $h_{t+1}^{2,\mathrm{LT}}(S)$ are the output of ST-TSLSTM and LT-TSLSTM. The final prediction of SLST-PKNet is then obtained by integrating DCI (in formula (1)), and all the sub-model outputs below in formula (10), where $\widehat{Y}_{t+1}$ denotes the model's final prediction at time t + 1. The Adam algorithm is utilized to optimize the parameters [20,28].

$$\widehat{Y}_{t+1} = \beta_1 \times \widehat{Y}_{t+1}\left(Y, \widetilde{Y}\right) + \beta_2 \times \widehat{Y}_{t+1}(Y, X) + \beta_3 \times \widehat{Y}_{t+1}\left(Y, X\widetilde{Y}\right) + \beta_4 \times \widehat{Y}_{t+1}^{\mathrm{DCI}} \tag{10}$$

6. Experiment

6.1. Experimental data

On the basis of referring to the design of existing research experiments [5,22,36], we conducted extensive sales forecasting experiments on 2 benchmark datasets: Galanz and Cainiao. Table 2 summarizes the corpus statistics including descriptions of both samples and features.

- Galanz dataset: This dataset is collected from Galanz, one of the leading home appliance enterprises in China. Over a two-year period, they stored more than 600 products in 100 warehouses. We selected 190 products in 10 warehouses.
- Cainiao dataset: The second set of data comes from Cainiao company,[1] one of the largest Intelligent logistics companies in China. There are more than 200 products in 5 warehouses.

For the Galanz dataset, we implemented the following rules to select high-quality data (190 products from 10 warehouses). We first removed products that had no sales records for the most recent three months; next, we removed products, the total sales of which were less than a specific threshold; finally, we removed products for which the percentage of missing values (0 sales) was greater than 30 %. The Cainiao data came from a public dataset. We used the whole dataset for training and testing.

We used historical time-series to predict sales of each product stored in the warehouses. Each product was grouped, firstly by warehouse, then by product ID. For each group, training and testing samples were generated by dividing the whole time series into a set of sub-series. The minimum length of each sub-series was greater than 5; the last time period of each

[1] https://tianchi.aliyun.com/competition/entrance/231530/information.

Table 2
Data Description of Galanz and Cainiao.

	Galanz	Cainiao
Product type	190	200
Samples	55,361	74,595
Features	•Historical Sales	•Historical Sales
	•Amount of Shop Discount	•User visits Records
	•Perform Discount Amount	•Visits to Cart
	•Discount Rate	•Collections user visits.

Table 3
Data Description of Traffic and Exchange-Rate.

	Traffic	Exchange-Rate
Length of time series	17,544	7,588
Number of variables	862	8
Sample rate	1 h	1 day

sub-series was taken as the label for prediction; other periods were taken as features. This procedure yielded 55,361 samples from Galanz and 74,595 samples from Cainiao. All datasets were split in chronological order to produce a training set (60 %), a validation set (20 %) and a testing set (20 %). These were named, respectively GWN (Galanz) and CWN (Cainiao). Warehouse IDs were used to divide Galanz testing data into 10 groups (GW1, GW2, GW3, ...GW10) and Cainiao testing data into 5 groups (CW1, CW2, ...CW5).

We also tested the effect of the model on two new datasets: Traffic and Exchange-Rate (Table 3):

- **Traffic:** A collection consists of 48 months (2015–2016) hourly data from the California Department of Transportation, which describes the road occupancy rates (between 0 and 1) measured by different sensors on San Francisco Bay area freeways.
- **Exchange-Rate:** A collection consists of the daily exchange rates of eight foreign countries including Australia, British, Canada, Switzerland, China, Japan, New Zealand and Singapore ranging from 1990 to 2016.

Traffic and Exchange-Rate datasets have been split into training set (60 %), validation set (20 %) and test set (20 %) in chronological order.

6.2. Metrics

Assume the testing dataset is Ω_{Test}. The total number of testing samples is $|\Omega_{\text{Test}}|$. We consider three conventional evaluation metrics which are commonly used in the corresponding tasks. These are defined in equations (11) to (13), where, for the ith testing sample in Ω_{Test}, Y_i denotes the true sales of and $\widehat{Y}_i$ denotes the predicted sales. RMAE and RRSE are scaled version of the widely used mean absolute error (MAE) and root square error (RSE), and CORR is the correlation coefficient. A low value of RMAE, RRSE and high value of CORR indicates good performance.

Relative Mean Absolute Error (RMAE) [5,20]:

$$\text{RMAE} = \frac{\text{mean}\left(\sum_{i\in\Omega_{\text{Test}}}\left|Y_i - \widehat{Y}_i\right|\right)}{\text{mean}\left(\sum_{j\in\Omega_{\text{Test}}} Y_j\right)} \tag{11}$$

Root Relative Squared Error (RRSE) [18,20]:

$$\text{RRSE} = \frac{\sqrt{\sum_{i\in\Omega_{\text{Test}}}\left(Y_i - \widehat{Y}_i\right)^2}}{\sqrt{\sum_{i\in\Omega_{\text{Test}}}(Y_i - \text{mean}(Y))^2}} \tag{12}$$

Correlation Coefficient (CORR) [18,20]:

$$\text{CORR} = \frac{1}{|\Omega_{\text{Test}}|}\sum_{i=1}^{|\Omega_{\text{Test}}|}\frac{\sum_i (Y_i - \text{mean}(Y))\left(\widehat{Y}_i - \text{mean}\left(\widehat{Y}\right)\right)}{\sqrt{\sum_i (Y_i - \text{mean}(Y))^2\left(\widehat{Y}_i - \text{mean}\left(\widehat{Y}\right)\right)^2}} \tag{13}$$

CORR is used to investigate the linear correlation between true sales Y_{t+1} and predicted sales $\widehat{Y}_{t+1}$. Thus, to a degree, it reflects the stability of the prediction results. The lower value of RMAE is not the only criterion by which to judge the performance of the model, because the model may only perform well on specific testing data, but have large deviations when tested with other data. In this case, the CORR of the model will be very low. A high CORR value indicates that in most cases, the deviation distribution between predicted sales and real sales is stable and not prone to large fluctuations.

6.3. Models for comparison

WaveNet[2]: A sequence generation model proposed by DeepMind [29]. Its core is an extended causal convolutional layer, which allows it to properly deal with temporal sequences and long-term dependencies without a marked increase in model complexity.

DARNN[3]: The Dual-stage Attention-based Recurrent Neural Network [28] uses a two-stage attention mechanism for MTS predictions. The first stage attention learns the weights of input variables, and the second stage attention learns the weights of hidden states across all time steps for forecasting.

LSTNet[4]: The Long- and Short-term Time-series Network [20] model contains a convolutional layer to extract local dependency patterns, a recurrent layer to capture long-term dependency patterns, and a recurrent-skip layer to capture periodic properties in the input data for forecasting.

MTNet[5]: Memory Time-series Network [3] is jointly trained by a large memory component, three separate encoders, and an auto-regressive component. The memory and attention component store the long-term historical data and deal with periods of time rather than single time steps.

MLCNN[6]: Multi-Level Construal Neural Network [5] is a multi-task deep learning framework that improves predictions by fusing information from future times. The framework uses a convolutional neural network to extract multi-level representations, then models interactions between multiple predictive tasks and fuses their future visions through an encoder-decoder framework.

MLCNN-PK: MLCNN-PK is an optimized version of MCLNN by incorporating PK component with MLCNN.

StemGNN[7]: Spectral time graph neural network (StemGNN) [3] combines Graph Fourier Transform (GFT) and Discrete Fourier Transform (DFT) into an end-to-end framework. GFT and DFT are used to model inter-series correlations and temporal dependencies separately. The GFT and DFT based spectral representations can be predicted effectively by convolution and sequential learning.

ST-Norm[8]: ST-Norm [8] proposed two types of normalization modules including temporal normalization and spatial normalization to refine the high-frequency and local components of the original data respectively. They can be integrated into a canonical deep learning architecture.

Informer[9]: Informer [43] designs a probSparse self-attention mechanism and distilling operation to address the Transformer's challenges of secondary time complexity and secondary memory usage. Meanwhile, a carefully designed generative decoder alleviates the limitations of the traditional encoder-decoder architecture.

Pyraformer[10]: Pyraformer [24] can bridge the gap between capturing remote dependencies and achieving low temporal and spatial complexity. In addition, a pyramidal attention module is introduced, with an inter-scale tree structure that combines features of different resolutions, while adjacent connections within the same scale model contain time dependences of different ranges.

BHT-ARIMA[11]: BHT-ARIMA [36] is designed to solve practical application problems in MTS prediction. In this model, the source time series data are augmented by high-order tensors with the help of multi-path delay transformation technology, and the classical time series prediction model ARIMA is combined with tensor decomposition. The model has been used to predict PC sales.

6.4. Experiment details

Grid searching strategy is adopted to find the best hyper-parameters for the proposed model and all the baselines. For LSTM, hidden size and hidden layer are set at 256 and 1 respectively; full connect layer size is 128; batch size is set at 16. For WaveNet, dense hidden size is set at 64; dilation rates are set at 2^i for i in range 1 to 4, Kernel size is set at 3 and the number of filters is set at 128. For DARNN, the encoder and decoder hidden layer are all set at 128, batch size is 16; time

[2] https://github.com/ibab/tensorflow-wavenet.
[3] https://github.com/fanyun-sun/DARNN.
[4] https://github.com/laiguokun/LSTNet.
[5] https://github.com/Maple728/MTNet.
[6] https://github.com/smallGum/MLCNN.
[7] https://github.com/microsoft/StemGNN.
[8] https://github.com/JLDeng/ST-Norm.
[9] https://github.com/zhouhaoyi/Informer2020.
[10] https://github.com/alipay/Pyraformer.
[11] https://github.com/yokotatsuya/BHT-ARIMA.

step is assigned as 28. For LSTNet, CNN and RNN hidden size are set at 100; CNN kernel size is 6; batch size is 16; skip and skip hidden are set at 24 and 5; highway window is set at 28. For MTNet, (as reported in the research [4]), CNN hidden size and RNN hidden size are set at 16 and 32 respectively; CNN filter height is 3; batch size is 64 and the number of long-term memory series is 7.

For SLST-PKNet, the short-term range *Rmin* is set at 4 for both Galanz and Cainiao and the *Rmax* is set at 20. Parameter of seasonal influence (SI) is set at 24. Important annual promotion days include: Dec.31, Feb.11, May.1, June.18, Oct.1, Nov.11 and Dec.12. These are selected as parameters of promotional influence (PI). The hidden dimension of the Recurrent and Convolutional layer is set at 100. CNN kernel size is 6. Dropout is performed after each layer, except the input and output ones, and the rate is set to 0.2. The adam algorithm is utilized to optimize the parameters.

6.5. Main results

Experimental evaluations of two benchmark sales datasets were conducted on each warehouse, and the results (10-folder cross validation), compared with 11 baselines, as reported in Tables 4-6. 10-folder cross-validation is adopted to evaluate the proposed and all baselines on the whole datasets as well as testing data in each warehouse of both Galanz and Cainiao (Tables 4-6). The error ranges of most of the results are smaller than 0.1, which indicates that all models have been effectively trained.

SLST-PKNet significantly outperforms the other baselines on the total testing data GWN and CWN. The average improvement in GWN is about 37 % by RMAE, 46 % by RRSE, and the average improvement in CWN is about 46 % by RMAE, 29 % by RRSE. Compared with the best baselines, the improvements are 3.2 % and 17.1 % on Galanz and 18.5 % and 6.8 % on Cainiao. The greater improvement in RRSE indicates that the predicted $\widehat{Y}_i$ derived from baselines, have stronger fluctuations when the values of testing samples are diverse, which suggests that the results are not stable if the sales of different products vary greatly (For example, products with high and low sales). In addition, by studying the CORR metrics, we find that the pre-

Table 4
Performance of the proposed SLST-PKNet model on Galanz dataset. (GW = Galanz Warehouse, GW1 – GW5 are the top 5 ranked warehouses).

Metrics	Methods	GW1	GW2	GW3	GW4	GW5	GWN
RMAE	**WaveNet**	0.56 ± 0.057	0.06 ± 0.001	0.54 ± 0.033	0.83 ± 0.076	0.82 ± 0.008	1.20 ± 0.023
	LSTNet	0.51 ± 0.002	0.06 ± 0.003	0.74 ± 0.005	0.76 ± 0.046	0.85 ± 0.032	1.18 ± 0.014
	MTNet	0.52 ± 0.011	0.18 ± 0.004	0.71 ± 0.003	0.79 ± 0.005	0.87 ± 0.004	1.24 ± 0.008
	DARNN	0.71 ± 0.006	0.57 ± 0.038	0.22 ± 0.004	0.82 ± 0.078	0.73 ± 0.028	1.25 ± 0.026
	MLCNN	0.20 ± 0.001	0.16 ± 0.004	0.24 ± 0.130	0.35 ± 0.003	0.45 ± 0.020	0.65 ± 0.006
	MLCNN-PK	0.18 ± 0.023	0.15 ± 0.001	0.21 ± 0.011	0.33 ± 0.003	0.44 ± 0.003	0.63 ± 0.023
	StemGNN	0.51 ± 0.000	**0.05 ± 0.008**	0.74 ± 0.001	0.76 ± 0.023	0.85 ± 0.008	1.10 ± 0.007
	ST-Norm	0.23 ± 0.043	0.08 ± 0.007	0.20 ± 0.007	**0.12 ± 0.023**	0.69 ± 0.015	0.83 ± 0.004
	Informer	0.51 ± 0.001	0.07 ± 0.001	0.75 ± 0.000	0.76 ± 0.006	0.82 ± 0.063	1.14 ± 0.007
	Pyraformer	0.51 ± 0.003	0.10 ± 0.003	0.76 ± 0.004	0.76 ± 0.000	0.88 ± 0.004	1.21 ± 0.003
	BHT-ARIMA	0.25 ± 0.000	0.06 ± 0.000	0.53 ± 0.000	0.36 ± 0.000	0.64 ± 0.001	0.95 ± 0.000
	SLST-PKNet	**0.17 ± 0.023**	0.15 ± 0.011	**0.19 ± 0.005**	0.30 ± 0.016	**0.38 ± 0.001**	**0.61 ± 0.010**
RRSE	**WaveNet**	1.01 ± 0.008	1.06 ± 0.012	1.03 ± 0.003	1.01 ± 0.013	1.10 ± 0.011	1.01 ± 0.003
	LSTNet	1.03 ± 0.012	1.16 ± 0.019	1.03 ± 0.012	1.02 ± 0.011	1.12 ± 0.006	1.03 ± 0.014
	MTNet	1.05 ± 0.006	2.77 ± 0.099	1.04 ± 0.060	1.12 ± 0.083	1.19 ± 0.064	1.02 ± 0.043
	DARNN	0.82 ± 0.014	3.56 ± 0.034	0.23 ± 0.013	0.85 ± 0.004	0.71 ± 0.045	0.68 ± 0.024
	MLCNN	0.65 ± 0.065	2.37 ± 0.299	0.22 ± 0.007	0.14 ± 0.015	0.55 ± 0.001	1.24 ± 0.019
	MLCNN-PK	0.64 ± 0.007	2.21 ± 0.075	0.21 ± 0.011	0.12 ± 0.011	0.54 ± 0.001	1.20 ± 0.052
	StemGNN	1.04 ± 0.001	**1.05 ± 0.148**	1.04 ± 0.004	1.03 ± 0.002	1.12 ± 0.002	1.29 ± 0.003
	ST-NORM	0.72 ± 0.015	1.21 ± 0.061	0.18 ± 0.002	**0.09 ± 0.028**	0.75 ± 0.009	0.64 ± 0.009
	Informer	1.02 ± 0.005	1.06 ± 0.022	1.01 ± 0.003	1.01 ± 0.002	1.11 ± 0.002	1.03 ± 0.001
	Pyraformer	1.01 ± 0.000	1.52 ± 0.088	1.00 ± 0.000	1.00 ± 0.002	1.09 ± 0.004	1.03 ± 0.002
	BHT-ARIMA	**0.44 ± 0.000**	1.18 ± 0.000	0.71 ± 0.000	0.41 ± 0.000	0.84 ± 0.000	1.49 ± 0.000
	SLST-PKNet	0.75 ± 0.002	2.42 ± 0.053	**0.14 ± 0.003**	0.15 ± 0.026	**0.52 ± 0.006**	**0.53 ± 0.008**
CORR	**WaveNet**	0.38 ± 0.019	0.24 ± 0.140	0.23 ± 0.023	0.50 ± 0.076	0.59 ± 0.058	0.28 ± 0.007
	LSTNet	0.08 ± 0.002	0.38 ± 0.012	0.17 ± 0.010	0.02 ± 0.026	0.01 ± 0.046	0.02 ± 0.003
	MTNet	0.05 ± 0.014	0.05 ± 0.003	0.23 ± 0.005	0.03 ± 0.026	0.12 ± 0.003	0.12 ± 0.011
	DARNN	0.70 ± 0.004	0.20 ± 0.089	0.99 ± 0.001	0.77 ± 0.046	0.76 ± 0.011	0.74 ± 0.042
	MLCNN	0.94 ± 0.006	0.68 ± 0.148	**1.00 ± 0.001**	0.96 ± 0.005	0.94 ± 0.004	0.59 ± 0.007
	MLCNN-PK	0.95 ± 0.003	**0.71 ± 0.063**	0.99 ± 0.000	0.97 ± 0.035	0.95 ± 0.022	0.60 ± 0.011
	StemGNN	0.95 ± 0.018	0.39 ± 0.100	0.96 ± 0.001	0.92 ± 0.024	0.65 ± 0.043	0.14 ± 0.013
	ST-NORM	0.93 ± 0.014	0.17 ± 0.043	0.94 ± 0.001	0.95 ± 0.002	0.80 ± 0.155	0.78 ± 0.009
	Informer	0.74 ± 0.186	0.31 ± 0.093	0.82 ± 0.105	0.86 ± 0.002	0.61 ± 0.051	0.53 ± 0.029
	Pyraformer	0.79 ± 0.019	0.42 ± 0.029	0.86 ± 0.039	0.80 ± 0.086	0.50 ± 0.095	0.57 ± 0.004
	BHT-ARIMA	0.95 ± 0.000	0.16 ± 0.000	0.94 ± 0.000	0.95 ± 0.000	0.80 ± 0.003	0.29 ± 0.000
	SLST-PKNet	**0.99 ± 0.007**	0.61 ± 0.022	0.96 ± 0.013	**0.97 ± 0.008**	**0.96 ± 0.012**	**0.85 ± 0.003**

Table 5
Performance of the proposed SLST-PKNet model on Galanz dataset. (GW = Galanz Warehouse, GW6 – GW10 are the last 5 ranked warehouses).

Metrics	Methods	GW6	GW7	GW8	GW9	GW10	GWN
RMAE	**WaveNet**	1.14 ± 0.021	1.05 ± 0.098	1.38 ± 0.114	1.19 ± 0.239	1.24 ± 0.004	1.20 ± 0.023
	LSTNet	0.04 ± 0.027	1.08 ± 0.035	0.93 ± 0.097	0.98 ± 0.008	1.05 ± 0.020	1.18 ± 0.014
	MTNet	0.29 ± 0.011	0.95 ± 0.001	1.25 ± 0.023	1.12 ± 0.006	1.37 ± 0.004	1.24 ± 0.008
	DARNN	0.29 ± 0.032	1.15 ± 0.068	1.08 ± 0.108	0.88 ± 0.056	0.98 ± 0.021	1.25 ± 0.026
	MLCNN	0.15 ± 0.013	0.75 ± 0.025	0.60 ± 0.030	0.62 ± 0.048	0.52 ± 0.010	0.65 ± 0.006
	MLCNN-PK	0.12 ± 0.005	0.73 ± 0.055	0.59 ± 0.020	**0.59 ± 0.125**	0.51 ± 0.005	0.63 ± 0.023
	StemGNN	0.04 ± 0.003	0.93 ± 0.013	1.25 ± 0.015	0.98 ± 0.008	1.23 ± 0.011	1.10 ± 0.007
	ST-NORM	0.08 ± 0.009	0.74 ± 0.011	0.85 ± 0.015	0.76 ± 0.006	0.94 ± 0.051	0.83 ± 0.004
	Informer	0.47 ± 0.042	0.51 ± 0.046	1.10 ± 0.123	1.12 ± 0.096	1.16 ± 0.135	1.14 ± 0.007
	Pyraformer	0.05 ± 0.003	0.98 ± 0.002	1.23 ± 0.001	1.03 ± 0.002	1.28 ± 0.004	1.21 ± 0.003
	BHT-ARIMA	**0.03 ± 0.000**	0.79 ± 0.000	1.06 ± 0.003	0.81 ± 0.000	1.00 ± 0.000	0.95 ± 0.000
	SLST-PKNet	0.18 ± 0.003	**0.50 ± 0.006**	**0.55 ± 0.011**	0.85 ± 0.007	**0.30 ± 0.016**	**0.61 ± 0.010**
RRSE	**WaveNet**	0.98 ± 0.001	1.01 ± 0.034	0.94 ± 0.003	0.98 ± 0.010	1.05 ± 0.001	1.01 ± 0.003
	LSTNet	1.03 ± 0.023	1.04 ± 0.003	1.09 ± 0.031	1.07 ± 0.004	1.10 ± 0.043	1.03 ± 0.014
	MTNet	1.08 ± 0.033	1.15 ± 0.105	1.11 ± 0.003	1.08 ± 0.012	1.14 ± 0.109	1.02 ± 0.043
	DARNN	5.16 ± 0.012	0.88 ± 0.015	0.74 ± 0.001	0.74 ± 0.044	0.84 ± 0.011	0.68 ± 0.024
	MLCNN	2.61 ± 0.009	1.87 ± 0.136	0.83 ± 0.003	0.65 ± 0.019	0.47 ± 0.022	1.24 ± 0.019
	MLCNN-PK	2.35 ± 0.033	1.83 ± 0.011	0.81 ± 0.038	**0.63 ± 0.021**	0.46 ± 0.102	1.20 ± 0.052
	StemGNN	1.05 ± 0.080	1.04 ± 0.000	1.12 ± 0.000	1.07 ± 0.001	1.05 ± 0.003	1.29 ± 0.003
	ST-NORM	0.99 ± 0.002	0.82 ± 0.018	0.70 ± 0.020	0.68 ± 0.012	0.74 ± 0.040	0.64 ± 0.009
	Informer	1.04 ± 0.010	1.04 ± 0.000	1.10 ± 0.000	1.06 ± 0.001	1.04 ± 0.001	1.03 ± 0.001
	Pyraformer	1.00 ± 0.000	1.04 ± 0.000	1.09 ± 0.004	1.06 ± 0.000	1.04 ± 0.000	1.03 ± 0.002
	BHT-ARIMA	0.63 ± 0.000	0.91 ± 0.000	0.91 ± 0.001	0.85 ± 0.000	0.79 ± 0.000	1.49 ± 0.000
	SLST-PKNet	**0.58 ± 0.012**	**0.45 ± 0.002**	**0.42 ± 0.009**	0.89 ± 0.031	**0.41 ± 0.001**	**0.53 ± 0.008**
CORR	**WaveNet**	0.29 ± 0.004	0.08 ± 0.005	0.51 ± 0.061	0.33 ± 0.012	0.02 ± 0.010	0.28 ± 0.007
	LSTNet	0.17 ± 0.012	0.05 ± 0.023	0.02 ± 0.065	0.02 ± 0.002	0.10 ± 0.090	0.02 ± 0.003
	MTNet	0.21 ± 0.028	0.07 ± 0.030	0.02 ± 0.005	0.03 ± 0.019	0.18 ± 0.012	0.12 ± 0.011
	DARNN	0.86 ± 0.004	0.52 ± 0.016	0.77 ± 0.033	0.67 ± 0.053	0.57 ± 0.016	0.74 ± 0.042
	MLCNN	0.86 ± 0.007	0.64 ± 0.012	0.87 ± 0.002	0.91 ± 0.004	0.93 ± 0.064	0.59 ± 0.007
	MLCNN-PK	0.88 ± 0.033	0.66 ± 0.003	0.88 ± 0.001	**0.93 ± 0.013**	0.94 ± 0.011	0.60 ± 0.011
	StemGNN	0.88 ± 0.003	0.61 ± 0.000	0.72 ± 0.000	0.72 ± 0.024	0.77 ± 0.018	0.14 ± 0.013
	ST-NORM	0.57 ± 0.013	0.60 ± 0.024	0.70 ± 0.021	0.72 ± 0.006	0.74 ± 0.094	0.78 ± 0.009
	Informer	0.23 ± 0.116	0.49 ± 0.003	0.71 ± 0.051	0.58 ± 0.139	0.55 ± 0.040	0.53 ± 0.029
	Pyraformer	0.36 ± 0.018	0.41 ± 0.007	0.50 ± 0.029	0.55 ± 0.115	0.74 ± 0.001	0.57 ± 0.004
	BHT-ARIMA	**0.88 ± 0.000**	0.62 ± 0.000	0.67 ± 0.000	0.65 ± 0.001	0.87 ± 0.000	0.29 ± 0.000
	SLST-PKNet	0.85 ± 0.001	**0.90 ± 0.011**	**0.92 ± 0.002**	0.65 ± 0.033	**0.96 ± 0.015**	**0.85 ± 0.003**

dicted $\widehat{Y}_i$ of SLST-PKNet has strong correlations with the true number of sales Y_i for both GWN and CWN (0.85 and 0.79 respectively). For the other baselines, correlations between $\widehat{Y}_i$ and Y_i were much weaker. As for the performances from different warehouses, the proposed SLST-PKNet obtained the highest scores of all the three metrics on 3 of the 5 Cainiao warehouses (CW1, CW4 and CW5), and on 4 of the 10 Galanz warehouses (GW5, GW7, GW8 and GW10). On each single indicator (RMAE, RRSE and CORR), the proposed model can obtain the best performances on at least 50 % of warehouses. While for the best baseline models, the maximum value is 40 %, which is obtained by ST-Norm. ST-Norm has the best RRSE on 2 of the 5 warehouses of Cainiao.

Transformer and tensor decomposition-based methods are also competitive baselines. For example, BHT-ARIMA is a tensor decomposition-based model, and obtains the best RRSE on GW1 and GW6. However, SLST-PKNet also outperforms BHT-ARIMA significantly, the average improvements on GWN and CWN are 42 %, 46 % and 31 % in terms of RMAE, RRSE and CORR. Compared with Informer and Pyraformer, the average improvements are 49 %, 41 % and 47 % respectively.

The main reason for the improved performance of SLST-PKNet is that, as stated in the Introduction, the model has four advantages over competitive baselines, such as MLCNN-PK, ST-NORM, Transformer-based models and BHT-ARIMA. First, the model takes the influences of different time series types into consideration. Second, by incorporating a DCI component, SLST-PKNet treats linear and non-linear correlations separately. Third, the two-layer and two-stage designs of TLCNN and TSLSTM can use the hidden state at time t to infer the future unknown dependent correlations at time t + 1 recursively by adopting dynamic weight strategy, and so can better capture dynamic changing patterns for future vision-based sales prediction from a more microscopic level. The state-of-the-art baselines, such as BHT-ARIMA, maps the internal correlations of the input MTS to each orthogonal dimension of the kernel tensor without particularly modeling the dynamic correlations, and this operation may lose some useful patterns, because it does not fully consider the complexity of factors affecting the future states of MTS. Similarly, Informer and Pyraformer mainly focus on capturing attention based long short-term

Table 6
Performance of the proposed SLST-PKNet model on Cainiao dataset. (CW = Cainiao Warehouse).

Metrics	Methods	CW1	CW2	CW3	CW4	CW5	CWN
RMAE	**WaveNet**	1.364 ± 0.007	1.391 ± 0.000	1.053 ± 0.009	1.882 ± 0.022	1.135 ± 0.049	1.340 ± 0.002
	LSTNet	1.229 ± 0.003	1.109 ± 0.005	1.004 ± 0.001	1.814 ± 0.003	0.982 ± 0.004	1.159 ± 0.006
	MTNet	1.395 ± 0.004	1.255 ± 0.004	1.304 ± 0.003	2.274 ± 0.003	1.164 ± 0.004	1.461 ± 0.006
	DARNN	1.197 ± 0.016	1.247 ± 0.089	1.315 ± 0.081	2.064 ± 0.098	1.343 ± 0.090	1.479 ± 0.043
	MLCNN	1.198 ± 0.007	1.165 ± 0.000	0.994 ± 0.007	1.819 ± 0.008	1.016 ± 0.022	1.218 ± 0.001
	MLCNN-PK	1.173 ± 0.023	1.122 ± 0.013	0.975 ± 0.002	1.797 ± 0.002	0.985 ± 0.010	1.192 ± 0.001
	StemGNN	1.090 ± 0.234	0.789 ± 0.143	0.881 ± 0.282	1.569 ± 0.328	0.810 ± 0.249	0.988 ± 0.215
	ST-NORM	0.758 ± 0.097	0.638 ± 0.007	0.591 ± 0.007	1.254 ± 0.012	0.704 ± 0.143	0.788 ± 0.015
	Informer	1.329 ± 0.002	1.244 ± 0.000	1.056 ± 0.000	1.854 ± 0.047	1.063 ± 0.003	1.302 ± 0.001
	Pyraformer	1.335 ± 0.000	1.244 ± 0.000	1.056 ± 0.001	1.907 ± 0.001	1.068 ± 0.000	1.307 ± 0.001
	BHT-ARIMA	1.271 ± 0.000	1.188 ± 0.000	1.008 ± 0.000	1.796 ± 0.001	1.014 ± 0.000	1.241 ± 0.000
	SLST-PKNet	**0.651 ± 0.002**	**0.601 ± 0.013**	**0.546 ± 0.003**	**0.985 ± 0.021**	**0.443 ± 0.008**	**0.642 ± 0.008**
RRSE	**WaveNet**	1.875 ± 0.022	1.335 ± 0.033	1.977 ± 0.016	2.279 ± 0.026	1.165 ± 0.015	1.427 ± 0.021
	LSTNet	1.000 ± 0.001	0.969 ± 0.002	1.036 ± 0.001	1.003 ± 0.001	0.991 ± 0.002	0.997 ± 0.003
	MTNet	1.094 ± 0.005	1.132 ± 0.006	1.531 ± 0.001	1.346 ± 0.002	1.037 ± 0.004	1.093 ± 0.006
	DARNN	0.897 ± 0.004	0.976 ± 0.029	0.952 ± 0.062	1.012 ± 0.011	1.005 ± 0.001	0.990 ± 0.010
	MLCNN	0.951 ± 0.008	1.048 ± 0.039	1.015 ± 0.000	0.990 ± 0.003	0.982 ± 0.004	0.988 ± 0.001
	MLCNN-PK	0.942 ± 0.016	1.032 ± 0.103	1.003 ± 0.003	0.965 ± 0.021	0.968 ± 0.075	0.976 ± 0.007
	StemGNN	0.847 ± 0.197	0.916 ± 0.284	2.792 ± 2.090	0.909 ± 0.120	0.804 ± 0.229	0.751 ± 0.031
	ST-NORM	0.646 ± 0.035	**0.619 ± 0.012**	**0.673 ± 0.028**	0.770 ± 0.018	0.650 ± 0.067	0.732 ± 0.008
	Informer	1.043 ± 0.001	1.040 ± 0.001	1.049 ± 0.000	1.029 ± 0.000	1.032 ± 0.001	1.033 ± 0.000
	Pyraformer	1.042 ± 0.001	1.038 ± 0.001	1.047 ± 0.000	1.030 ± 0.000	1.032 ± 0.000	1.032 ± 0.000
	BHT-ARIMA	0.978 ± 0.000	1.019 ± 0.000	0.998 ± 0.000	0.936 ± 0.000	0.985 ± 0.000	0.961 ± 0.000
	SLST-PKNet	**0.605 ± 0.036**	0.648 ± 0.001	0.895 ± 0.001	**0.651 ± 0.003**	**0.502 ± 0.002**	**0.682 ± 0.011**
CORR	**WaveNet**	0.231 ± 0.024	0.262 ± 0.003	0.246 ± 0.001	0.184 ± 0.004	0.212 ± 0.001	0.087 ± 0.050
	LSTNet	0.278 ± 0.003	0.337 ± 0.005	0.161 ± 0.000	0.216 ± 0.001	0.271 ± 0.008	0.246 ± 0.001
	MTNet	0.012 ± 0.001	0.031 ± 0.002	0.005 ± 0.001	0.004 ± 0.001	0.029 ± 0.001	0.006 ± 0.001
	DARNN	0.668 ± 0.010	0.557 ± 0.026	0.617 ± 0.020	0.615 ± 0.003	0.809 ± 0.012	0.192 ± 0.011
	MLCNN	0.826 ± 0.014	0.743 ± 0.003	0.634 ± 0.002	0.803 ± 0.008	0.806 ± 0.011	0.713 ± 0.008
	MLCNN-PK	0.833 ± 0.005	0.752 ± 0.007	0.641 ± 0.001	0.811 ± 0.001	0.818 ± 0.020	0.721 ± 0.013
	StemGNN	0.741 ± 0.013	0.713 ± 0.071	**0.796 ± 0.051**	0.790 ± 0.026	0.792 ± 0.028	0.509 ± 0.233
	ST-NORM	0.792 ± 0.037	**0.776 ± 0.010**	0.771 ± 0.017	0.743 ± 0.020	0.765 ± 0.055	0.739 ± 0.003
	Informer	0.182 ± 0.066	0.177 ± 0.070	0.150 ± 0.013	0.199 ± 0.026	0.251 ± 0.081	0.185 ± 0.009
	Pyraformer	0.195 ± 0.002	0.114 ± 0.000	0.146 ± 0.040	0.129 ± 0.032	0.178 ± 0.029	0.139 ± 0.003
	BHT-ARIMA	0.681 ± 0.000	0.415 ± 0.000	0.676 ± 0.000	0.800 ± 0.003	0.618 ± 0.000	0.721 ± 0.002
	SLST-PKNet	**0.847 ± 0.012**	0.773 ± 0.002	0.651 ± 0.000	**0.826 ± 0.000**	**0.863 ± 0.003**	**0.799 ± 0.005**

dependencies from MTS, while seldom consider how the state evolves from t to $t + 1$ at the micro level, and what kind of linear or nonlinear influences of factors are effective in this process.

The MLCNN-PK model is a more suitable baseline for making comparison, and can be used to further illustrate the three advantages mentioned above. In one aspect, MLCNN-PK is a very competitive baseline, which obtains one best RMAE, one best RRSE and three best CORRs on both Galanz and Cainiao datasets. In another aspect, MLCNN-PK also incorporates PK component, thus the comparison between MLCNN-PK and SLST-PKNet can be seen as an evaluation of the three advantages of SLST-PKNet. As shown in the experiment, SLST-PKNet outperforms MLCNN-PK significantly, the average improvements on both GWN and CWN are 25 %, 43 % and 16 % in terms of RMAE, RRSE and CORR.

Fourth, the model is able to capture sales patterns from a set of products of the same type by incorporating prior knowledge. For example, by capturing number of sales of a specific product during and after promotions, and in different seasons, the model can determine whether the product will be popular in subsequent months, while many similar products with the same type have similar sales patterns along with the time line. Existing research, such as the representative baselines referred to in this article, can obtain better performances on datasets with strong seasonal and cyclical patterns, such as traffic, electronic consuming time series, etc [4,20,37], but when applying those baseline models to the Galanz and Cainiao datasets, we found that they performed less well than expected. The main reason is that the time series from the two datasets have no obvious seasonal and cyclical patterns due to a longer time span. However, patterns were discernable if prior knowledge was applied.

The SLST-PKNet model is also evaluated on two new datasets: Traffic and Exchange-Rate. The two datasets only contain one type of time series, thus SLST-PKNet only needs to consider one sub-model. In addition, PK component only needs to consider the cyclical impact in this experiment. The experimental results can be seen in Table 7. Horizon represents different training windows, which is assigned as 3, 6, 12, 24 respectively for both Traffic and Exchange-Rate. Performances of AR, RNN-GRU, LSTNet and MTNet are reported in [4] and [20]. Three most competitive baselines MLCNN, StemGNN and ST-Norm are also selected. 10-folder cross validation is adopted to make evaluations. Different from sales time series, traffic time series have strong a periodic pattern; time series of different exchange rate have strong correlation patterns, therefore, the effects

Table 7
Performance of SLST-PKNet on Traffic and Exchange-Rate datasets.

Dataset		Traffic				Exchange-Rate			
		Horizon				Horizon			
Methods	Metrics	3	6	12	24	3	6	12	24
AR†	**RRSE**	0.5991	0.6218	0.6252	0.6293	0.0228	0.0279	0.0353	0.0445
	CORR	0.7752	0.7568	0.7544	0.7519	0.9734	0.9656	0.9526	0.9357
RNN-GRU†	**RRSE**	0.5358	0.5522	0.5562	0.5633	0.0192	0.0264	0.0408	0.0626
	CORR	0.8511	0.8405	0.8345	0.8300	0.9786	0.9712	0.9531	0.9223
LSTNet-Skip†	**RRSE**	0.4777	0.4893	0.4950	0.4973	0.0226	0.0280	0.0356	0.0449
	CORR	0.8721	0.8690	0.8614	0.8588	0.9735	0.9658	0.9511	0.9354
LSTNet-Attn†	**RRSE**	0.4897	0.4973	0.5173	0.5300	0.0276	0.0321	0.0448	0.0590
	CORR	0.8704	0.8669	0.8540	0.8429	0.9717	0.9656	0.9499	0.9339
MTNet†	**RRSE**	0.4764	0.4855	0.4877	0.5023	0.0212	0.0258	0.0347	0.0442
	CORR	0.8728	0.8681	0.8644	0.8570	0.9767	0.9703	0.9561	0.9388
StemGNN*	**RRSE**	–	–	–	–	0.068 ± 0.0012	0.072 ± 0.0016	0.098 ± 0.0004	0.113 ± 0.0004
	CORR	–	–	–	–	0.853 ± 0.0007	0.842 ± 0.0003	0.652 ± 0.0011	0.557 ± 0.0004
ST-Norm*	**RRSE**	0.987 ± 0.0080	1.032 ± 0.0010	0.946 ± 0.0065	0.977 ± 0.0080	0.071 ± 0.0010	0.062 ± 0.0005	0.048 ± 0.0065	0.058 ± 0.0015
	CORR	0.378 ± 0.0605	0.351 ± 0.0175	0.314 ± 0.0130	0.116 ± 0.0120	0.321 ± 0.1450	0.199 ± 0.0010	0.204 ± 0.0070	0.167 ± 0.0265
MLCNN*	**RRSE**	0.472 ± 0.0005	0.480 ± 0.0015	0.485 ± 0.0016	0.512 ± 0.0020	0.021 ± 0.0003	0.028 ± 0.0013	0.042 ± 0.0006	0.052 ± 0.0025
	CORR	0.872 ± 0.0007	0.868 ± 0.0013	0.863 ± 0.0007	0.844 ± 0.0003	0.976 ± 0.0003	0.967 ± 0.0004	0.948 ± 0.0008	0.932 ± 0.0020
SLST-PKNet	**RRSE**	**0.461 ± 0.0006**	**0.475 ± 0.0020**	**0.485 ± 0.0008**	**0.500 ± 0.0012**	**0.018 ± 0.0005**	**0.025 ± 0.0003**	**0.034 ± 0.0010**	**0.043 ± 0.0021**
	CORR	**0.878 ± 0.0010**	**0.872 ± 0.0011**	**0.865 ± 0.0006**	**0.862 ± 0.0008**	**0.979 ± 0.0003**	**0.971 ± 0.0005**	**0.956 ± 0.0008**	**0.939 ± 0.0015**

†: The results are retrieved from [4], [5] and [20]. *: For a fair comparison, we reproduce the results using their released implementation code and configuration on the same datasets.

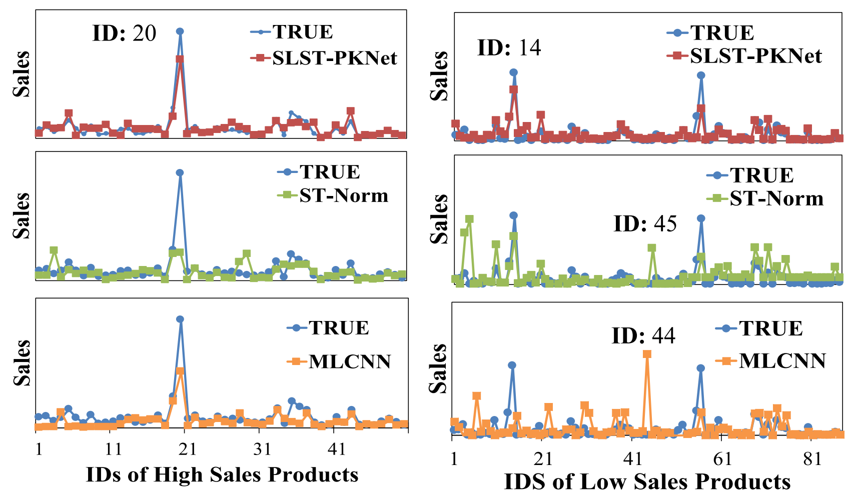


Fig. 7. Case studies of 137 samples from Galanz. (a) shows the results of 50 high sales products; (b) shows the results of 87 low sales products.

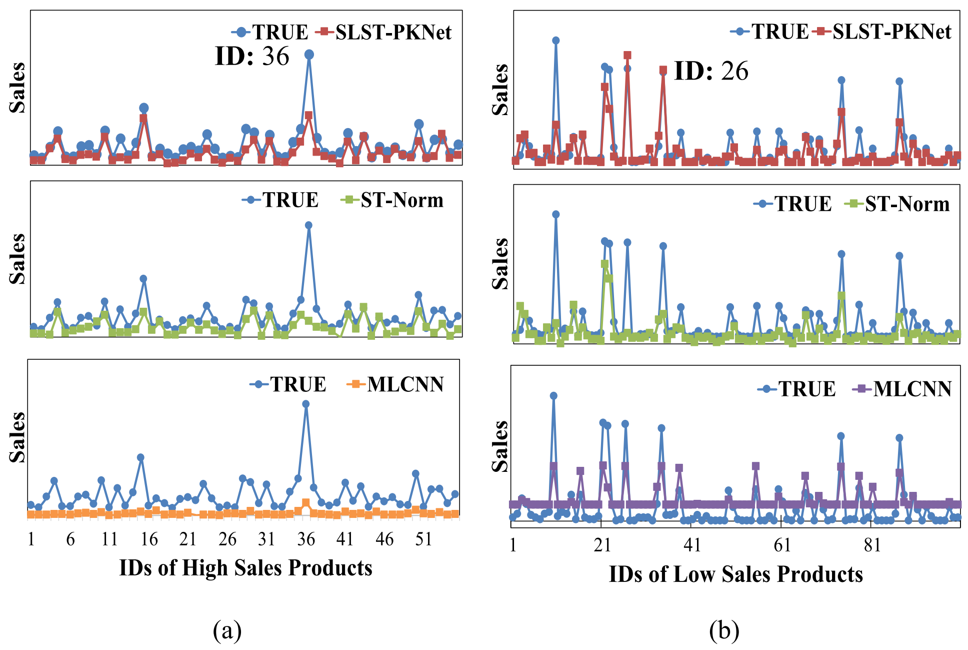


Fig. 8. Case studies of 155 samples from Cainiao. (a) shows the results of 55 high sales products; (b) shows the results of 100 low sales products.

of all models are relatively close. However, SLST-PKNet still outperforms all the baselines in terms of all the metrics on both Traffic and Exchange-Rate. Compared with all the baselines, the average improvements are 18 % and 15 % from Traffic data, 35 % and 28 % from Exchange-Rate data in terms of RRSE and CORR. Compared with the best baselines, the average improvements are 1.3 % and 0.5 % in terms of RRSE and CORR respectively.

Case studies are introduced to further illustrate the performance of the proposed model. Figs. 7 and 8 summarize the performances of SLST-PKNet on 50 high sales products and 87 low sales products from Galanz; 55 high sales products and 100 low sales products from Cainiao. High sales are defined as the average sales is larger than 500 and low sales are defined as the sales is between 100 and 200.

The experiment shows that the proposed model can obtain better performances on testing samples with different levels of sales. Two competitive models MLCNN and ST-Norm are taken as baselines for comparison. For most of the cases, the predictions by SLST-PKNet matched the true number of sales more accurately than predictions by MLCNN and ST-Norm, neither of which was able to effectively capture the patterns of sales time series for both high and low sales products. The greater performance of SLST-PKNet is particularly noticeable with Galanz ID 20 and 14, and with Cainiao ID 36 and 26. When the true sales are small (Low Sales), MLCNN and ST-Norm often over-predict (ID 44 and 45 in Fig. 7). A possible reason is that the baselines do not identify different time series types, and so cannot model their respective influences. Another reason is that the attention mechanism was not able to obtain expected results due to the lack of domain prior knowledge.

Experiments were also conducted on data from different warehouses at the same time *t*. Here we select GW6 and CW1 as examples, and ST-Norm, MLCNN, StemGNN and DARNN are selected as baselines. Fig. 9 shows the performance on GW6 and Fig. 10 relates to CW1.

The results show that the proposed model could identify different sales patterns of similar products in the same warehouse at the same time *t*. For example, SLST-PKNet correctly predicted that product ID 36, 40 would have the highest sales at time *t* while other methods failed to make correct predictions (Fig. 9). Similarly, the proposed model successfully predicted peaks for product IDs 286, 505, 640 at time *t* (Fig. 10). Therefore, the proposed model is better at capturing common trend patterns from MTS. In most cases, the predicted sales of the proposed model are very close to the true sales, which leads to a higher CORR. For the other baselines, there exist many cases where the deviations between predicted and true sales are large, which will significantly reduce the CORR. For example, though the RMAE of ST-Norm are relatively high, its CORR reduces by 17 % compared with SLST-PKNet.

In help understand why the proposed model gave better predictions, we visualized the time series of ID 7 and its related products in GW4. The results are shown in Fig. 11 and Fig. 12. Fig. 11 visualizes the prediction results of the proposed model on four selected product IDs: 7, 65, 82 and 122 at the same time *t*. ST-norm and MLCNN are taken as baselines. Experimental results show that the prediction results of SLST-PKNet are significantly closer to the true sales of all the selected products.

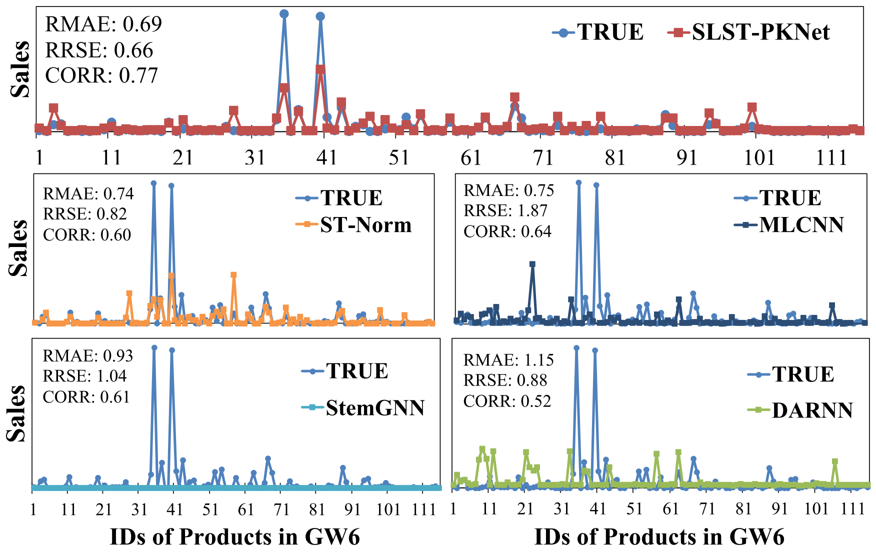


Fig. 9. Case Studies of 115 products from warehouse GW6 of Galanz.

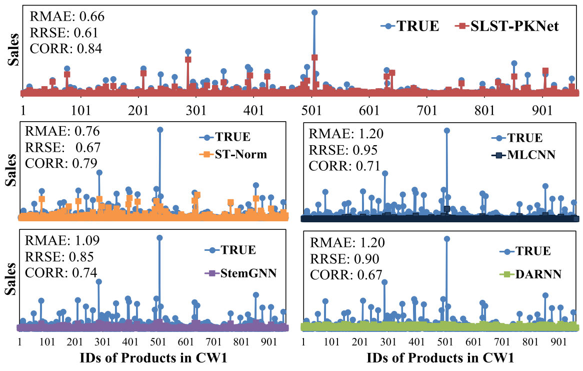


Fig. 10. Case Studies of 961 products from warehouse CW1 of Cainiao.

Fig. 12 visualizes the correlations between the four selected products in GW4. Given the target predicting sales of product ID 7 at time *t*, it is difficult to directly identify sales patterns of ID 7 without prior knowledge. Existing DNNs could not find evidence to support their predictions why there is a significant increase in ID 7's sales at time *t*. By incorporating prior knowledge of SI and PI, we find that similar products have common patterns (eg, ID 7, 65 and 82 in Fig. 12) during Nov, Feb and Jun.

As shown in the last subfigure of Fig. 12, if only SI is considered, there will be large errors because there are no strong seasonal patterns in sales of product ID 122. In Nov 2019, for example, sales were extremely high: by contrast, in Nov 2020, they were low. Various possible reasons could account for such fluctuations. The product may have been less popular than expected; or rival products might have emerged on the market. The information needed to support such suggestions was not available in the databases, but we could observe the sales patterns of similar products stored in the warehouses. For each selected product there were significant sales improvements during Feb-Mar 2020 and Jun-Jul 2020. Feb and Jun are usually-two important months for promotions in Chinese e-business companies. This is captured in the PI factors. Unlike the baselines, the proposed model was able to predict sales of ID 7 in Nov 2020 by additionally using LT-TSLSTM to analyze sales during those two time periods.

In order to test whether the model is over-fitting on the two sales datasets, a new experiment is conducted to analyze the performance of SLST-PKNet on the validation dataset (Fig. 13).

Fig. 13(a and c) show the convergence curves on validation data from both Galanz and Cainiao. As the number of iterations increases, the LOSS reduces rapidly. When the number of iterations reaches 60, the validation LOSS begins to converge.

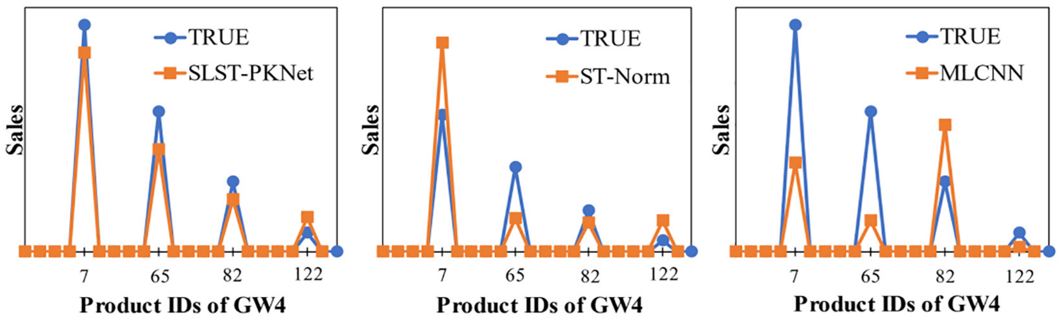


Fig. 11. Case Studies of 4 selected products from warehouse GW4 of Galanz.

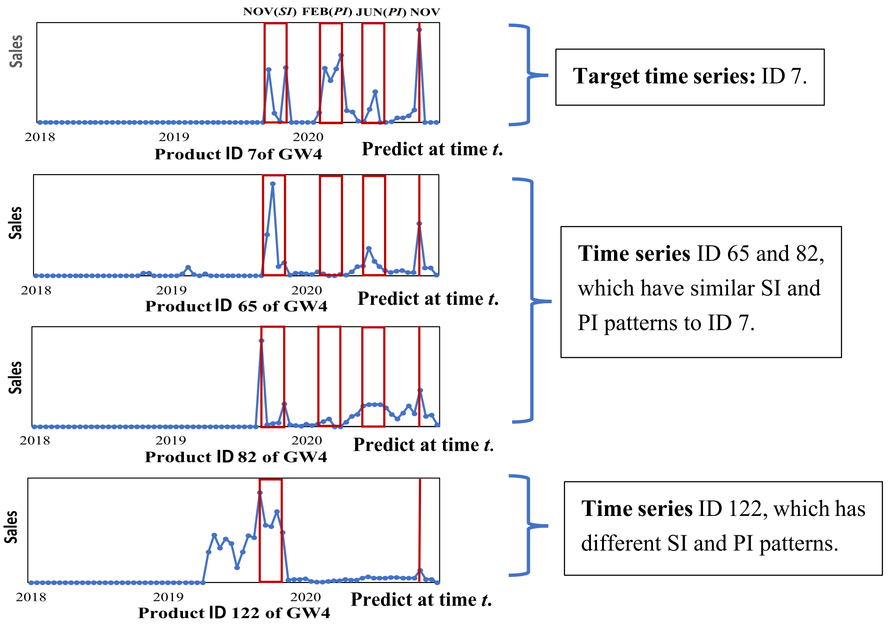
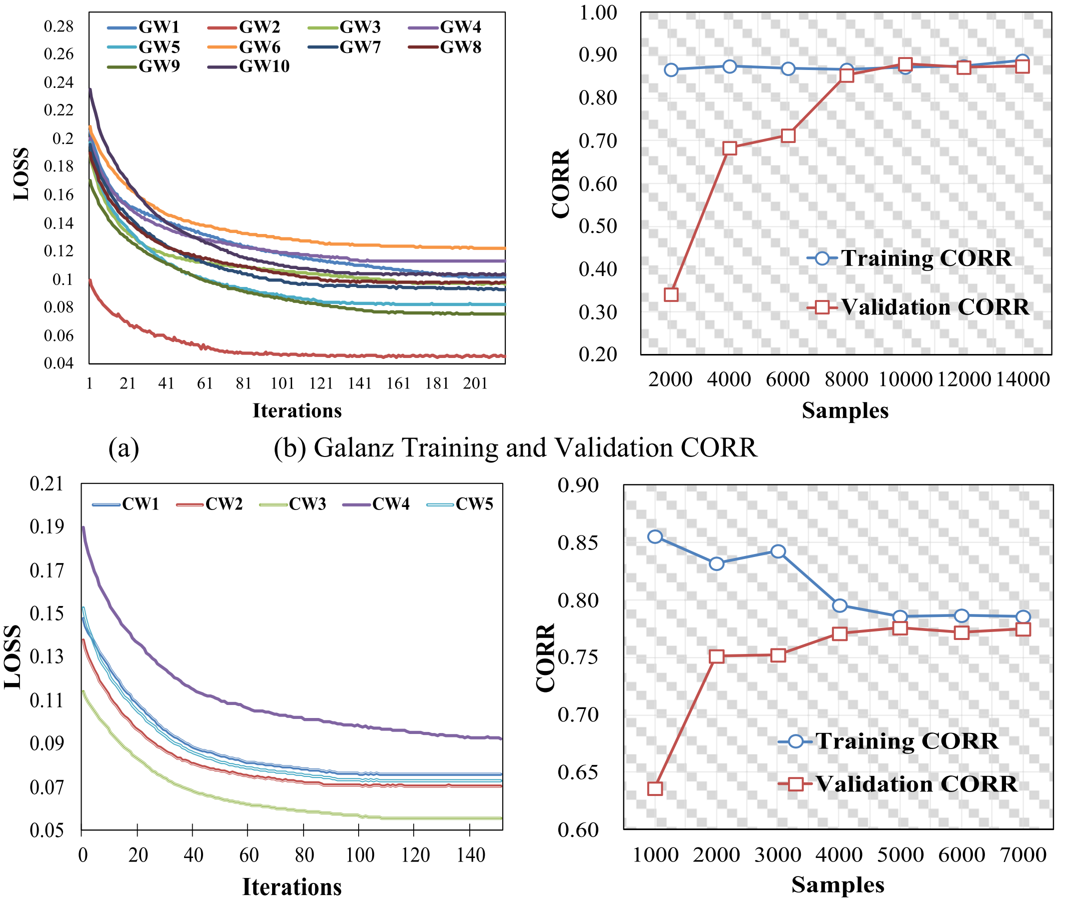


Fig. 12. Examples of SI and PI patterns of ID 7, 65, 82 and 122 in Warehouse GW4.



(a) (b) Galanz Training and Validation CORR

(c) Cainiao Convergence Curve (d) Cainiao Training and Validation CORR

Fig. 13. Convergence and over-fitting analysis of SLST-PKNet.

Fig. 13(b and d) show the training and validation curves of both Galanz and Cainiao along with the increasing number of training samples in terms of CORR. In Fig. 13(b), when the number of training samples reaches up to 8,000, the validation CORR of SLST-PKNet is very close to the training CORR. With the further increase of training samples, the CORR can reach 0.9, which indicates a quite small bias. The small deviations between training and validation CORR also indicates small variances. Fig. 13(d) shows similar results. These results indicate that the probability of over-fitting is small. The reason is that adopting dropout could effectively control the amount of training parameters. On the other hand, TLCNN + TSLSTM can generate dynamic weights that provide more choices for the model to learn different patterns, reducing the chance of fitting incorrect patterns. Finally, the PK component could help the model to avoid learning noisy information.

6.6. Ablation study

An ablation study was conducted to further verify the contribution of each component of the proposed model. Variations of the proposed SLST-PKNet were tested, in which without specific components were removed. These are as follows:

- SLST-PK w/o $\widetilde{Y}$: SLST-PKNet removed time series $\widetilde{Y}$ of all related products.
- SLST-PK w/o $X\widetilde{Y}$: SLST-PKNet removed all $X\widetilde{Y}$ combinations.
- SLST-PK w/o X: SLST-PKNet removed all features' time series X.
- SLST-PK w/o PK: SLST-PKNet removed prior knowledge from LT-TSLSTM.
- SLST-PK w/o DCI: SLST-PKNet removed dynamic co-integration component.
- SLST-PK w/o ST: SLST-PKNet removed ST-TSLSTM.
- SLST-PK w/o LT: SLST-PKNet removed LT-TSLSTM (PK is also removed).
- SLST-PK: The complete SLST-PKNet model.

- SLST-PK: The complete SLST-PKNet model.

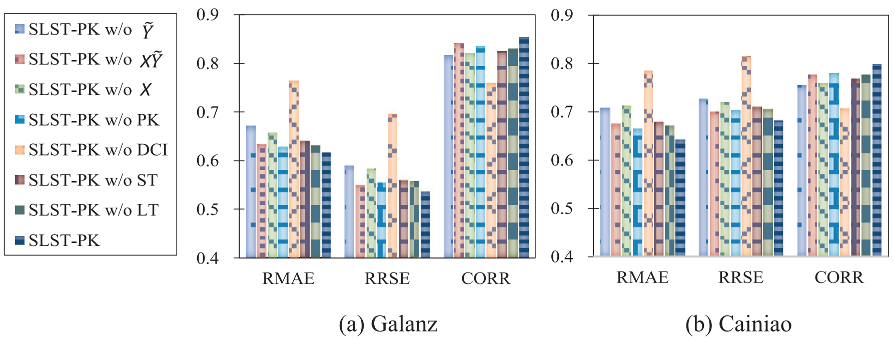


Fig. 14. Results of SLST-PKNet ablation test on Galanz and Cainiao.

The test results were evaluated using RMAE, RRSE and CORR, and are shown in Fig. 14. DCI makes the highest contribution towards predicting sales, because it can make up for the limitations of DNN-based methods. Although a DNN can obtain a good performance at capturing complex non-linear correlations from a set of time series, it is still hard for them to identify the pseudo regression phenomenon in the sales time series. The model may learn lots of irrelevant correlations from the time series of different products. The use of DCI can help to find smooth and stable correlations from MTS.

The sub-model $\left\{Y,\widetilde{Y}\right\}$ contributes about 8.7 %, 7.6 % and 5.2 % improvements in RMAE, RRSE and CORR respectively. The sub-model $\{Y,X\}$ contributes about 7.9 %, 6.7 % and 4.6 % improvements in RMAE, RRSE and CORR respectively. The combinations of $\left\{Y,\widetilde{Y},X\right\}$, which consider interactions between X and Y, produce a 3.7 % improvement in RMAE, a 2.5 % improvement in RRSE and a 2.2 % improvement in CORR. $\left\{Y,\widetilde{Y}\right\}$ produces the best performances. This indicates that the sales of target product are significantly influenced by the sales trends of other products of the same type. $\{Y,X\}$ also makes a positive contribution, though the improvement is not as great as that of $\left\{Y,\widetilde{Y}\right\}$. The main reason is that the feature time series in X are only related with target time series Y, and do not properly reflect the external changes of the marketing environment. The improvement of $\left\{Y,\widetilde{Y},X\right\}$ is limited compared with $\left\{Y,\widetilde{Y}\right\}$ and $\{Y,X\}$. One reason is that most of the contributions of $\left\{Y,\widetilde{Y},X\right\}$ can be explained by $\left\{Y,\widetilde{Y}\right\}$ and $\{Y,X\}$. Another reason is that concurrently considering the influences of $\widetilde{Y}$ and X on Y will produce some interference information. For example, when the sales of most products with the same type reduce, promotional activities may produce information which misleads the model, because the model could not identify different types (products or marketing features) of MTS in $\left\{Y,\widetilde{Y},X\right\}$.

Although Prior Knowledge (PK) is only effective on a small number of samples, those samples often have strong sales patterns and a high number of sales. So incorporation of PK can lead to the average of 2.6 %, 3.2 %, 2.3 % improvements in RMAE, RRSE and CORR. In another aspect, the performances of SLST-PKNet w/o PK on Galanz GWN are 0.628, 0.554 and 0.836 in terms of RMAE, RRSE and CORR. These values on Cainiao CWN are 0.665, 0.703 and 0.779 respectively. Compared with the best state-of-the-art baselines on each indicator (For Galanz GWN: the best baselines on RMAE, RRSE and CORR are MLCNN and ST-NORM respectively. For Cainiao CWN: the best baselines are all ST-NORM), the average improvements of SLST-PKNet w/o PK on Galanz are 3.4 %, 13.4 % and 7.2 %, and the corresponding values on Cainiao are 15.7 %, 3.9 % and 2.5 %. Experimental results show that SLST-PKNet can still obtain optimal results without the PK component, and the improvement is still significant compared with the best baselines. This further indicates that when PK is removed, the consideration of different time series types, the two-layer and two-stage designs of TLCNN and TSLSTM, and the design of the DCI component, still have their unique advantages compared with state-of-the-art baselines.

Due to the information sharing design of both ST-TSLSTM and LT-TSLSTM, the two components will learn a lot of similar knowledge during the training process, thus removing one component will not cause big influence towards model performances. ST-TSLSTM provides the average of 4.5 %, 4.1 % and 3.7 % improvements on both Galanz and Cainiao datasets in terms of RMAE, RRSE and CORR; while the average improvements of LT-TSLSTM are 3.2 %, 3.6 % and 2.9 % respectively.

We also tried to integrate the prior knowledge into ST-TSLSTM, which is named as ST-TSLSTM-PK. Assume at time t, we want to predict sales at time $t + 1$, then the fixed annual promotional activities between *Rmin* and t are taken as PI. For SI assignment, we set the cycle of short-term to 2, which is used to investigate the correlations between a time period of the current month and the corresponding time period of the previous month. Experiments are conducted on both Galanz and Cainiao, and the results in Table 8 show that the short-term PI and SI also obtain a small improvement. The method is similar to the idea of using different windows for combined feature design. However, the improvement is limited. One

Table 8
Experimental results of adding prior knowledge to ST-TSLSTM.

	Metrics	Galanz (GWN)	Cainiao (CWN)
ST-TSLSTM-PK	RMAE	**0.6369 ± 0.013**	**0.6558 ± 0.006**
	RRSE	**0.5501 ± 0.004**	**0.6885 ± 0.012**
	CORR	**0.8467 ± 0.007**	<u>0.7876 ± 0.010</u>
ST-TSLSTM	RMAE	<u>0.6470 ± 0.005</u>	<u>0.6571 ± 0.005</u>
	RRSE	<u>0.5532 ± 0.013</u>	<u>0.6910 ± 0.008</u>
	CORR	<u>0.8436 ± 0.001</u>	**0.7913 ± 0.007**

reason is that the time range of short-term is small, and the influence weight of key time points can be effectively learned without assigning prior knowledge in advance. In addition, the probability of noise interference in the learning process will also be quite small.

6.7. Running efficiency

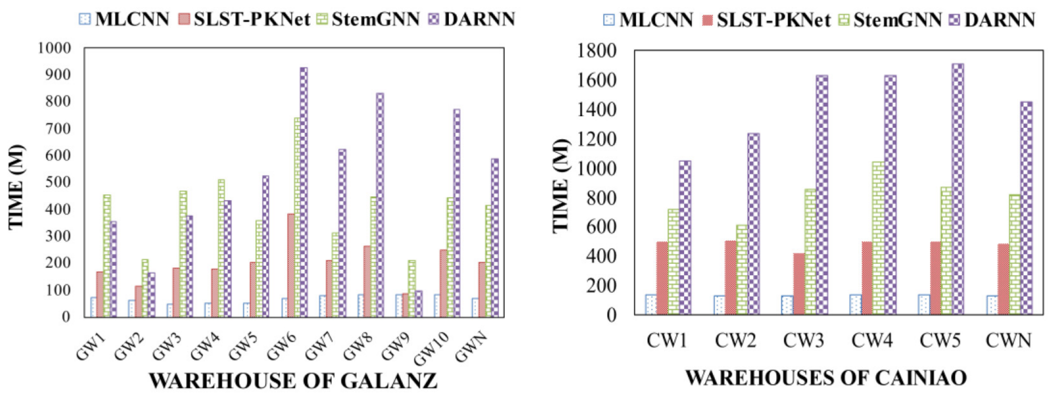
We conducted additional experiments to evaluate the running efficiency of the proposed model. Running efficiency is determined by three factors: learning of the three sub-models, calculation of DCI, and calculation of TLCNN-TSLSTM. For the first factor, we used the strategy of Distributed Data Parallel (DDP) to train the three sub-models in parallel. For the second factor, we constrained the number of each MTS to no bigger than 20, which can accelerate the calculation speeds of DCI. For the third factor, we designed a multi-dimensional tensor to store and update all the parameters in each iteration by directly making tensor calculation. This operation is different from earlier research (such as DARNN), which updates the parameters of each time series in MTS by traversal. We took MLCNN, DARNN and StemGNN as baselines, and ran experiments on both the Galanz and Cainiao datasets.

The results in Fig. 15 show that SLST-PKNet is slower than MLCNN, because SLST-PKNet considers a greater number of linear and non-linear correlations from the perspective of long short-term. However, its running efficiency is significantly better than that of DARNN and StemGNN. The structure of StemGNN contains two blocks, each of which has four-layer encoding and decoding operations based on time and frequency domains. So it has more parameters than SLST-PKNet.

7. Discussion

7.1. TSLSTM for modeling future vision

One more important innovation that should be further discussed, is the design of TSLSTM for modeling future vision. In designing a TSLSTM component, our target was to apply it to realistic sales prediction task, i.e., to predict sales of a product at time period t. In reality, we usually do not know its newest promotional strategies (X) and sales of other related products ($\widetilde{Y}$) at t. TSLSTM firstly predicts $X(t)$ and $\widetilde{Y}(t)$, then uses its predictions to predict sales of a target product $Y(t)$. We compared the component with two traditional methods: "Multivariate time series forecasting" (Multi-TSF) and "Univariate time series



(a) Running efficiency on Galanz (b) Running efficiency on Cainiao

Fig. 15. Running efficiency of the proposed SLST-PKNet.

Table 9
Performance of two-Stage LSTM for Future Vision Modeling.

	Metrics	Galanz (GWN)	Cainiao (CWN)
LST-PK-Uni-TSF	RMAE	0.7116	<u>0.6672</u>
	RRSE	0.6015	<u>0.6943</u>
	CORR	0.8091	0.7351
LST-PK-Multi-TSF	RMAE	<u>0.6965</u>	0.6787
	RRSE	<u>0.5836</u>	0.7092
	CORR	<u>0.8234</u>	<u>0.7523</u>
SLST-PKNet	RMAE	**0.6130**	**0.6420**
	RRSE	**0.5322**	**0.6821**
	CORR	**0.8534**	**0.7991**

forecasting" (Uni-TSF). Multi-TSF is similar to DARNN [28] and LSTNet [20], which could predict all the time series' values at time t. Uni-TSF uses an auto-regression model to predict the value at t of each time series in X and $\widetilde{Y}$ separately, then uses the predicted value to forecast $Y(t)$. We replaced the TSLSTM with Multi-TSF (LST-PK-Multi-TSF) and Unit-TSF (LST-PK-Unit-TSF) respectively, and conducted new experiments. Experimental results can be seen in Table 9.

Results of the experiment show significant reduction in errors (RMAE and RRSE). These findings suggest that TSLSTM has the potential to better model future correlations of different variables.

7.2. Temporal attention weight distributions of SLST-PKNet

To help understand why the model improves predictions, we draw temporal attention weight distributions of the proposed SLST-PKNet (Fig. 16). The first subfigure of Fig. 16 shows the sales time series of product ID 7 in GW4. The red rectangle indicates that we would like to predict real sales $Y(t)$ = 1676 at time t. The second subfigure indicates the weight distributions of baseline DARNN. We find it hard to identify patterns from the temporal attention sequences of DARNN, because the information in the sales time series is noisy, with long sequences of low or zeros sales. The predicted value of DARNN at time period t is 273, deviates from the real $Y(t)$ by 1403.

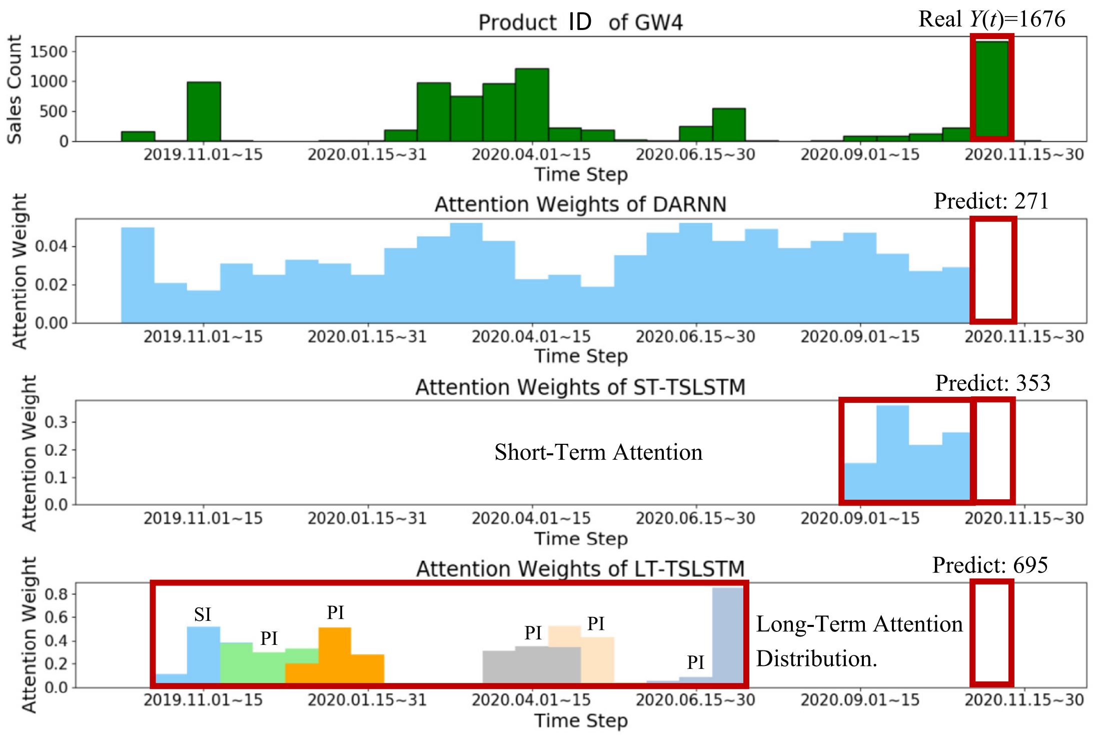


Fig. 16. Temporal attention weight distributions of SLST-PKNet.

Unlike previous studies, SLST-PKNet does not calculate the attention weights of the whole time series. Instead, it focuses on long short-term modeling, and captures local information by using prior knowledge. The third subfigure of Fig. 16 shows attention weight distributions of short-term time series by adopting ST-TSLSTM. Temporal attention of short-term time series only considers 4 time periods ($t-4-t-1$), which concentrates on the influence of the nearest sequence to $Y(t)$. The predicted value of ST-TSLSTM is 353, which indicates that the inclusion of longer time series may lead to deviations due to the noisy information in the added time series.

The fourth subfigure in Fig. 16 shows attention weight distributions of long-term series by adopting LT-TSLSTM. Unlike DARNN, LT-TSLSTM analyzes a set of sub time series based on prior knowledge. For example, the target $Y(t)$ is at time period 2020.11.01 – 15, so the sub time series of its SI is from 2019.10.16 – 31 to 2019.11.16 – 30. The assignment of Promotional Influence (PI) is mainly based on important promotional time periods of e-business companies, mostly associated with major festivals and holidays. SI and PI help to analyze the performance of the target product at the key time, and reduce the influence of noisy time series segments. The predicted value of LT-TSLSTM at time period t is 695. This is a significant improvement on results obtained using DARNN and ST-TSLSTM.

8. Conclusion

In this research, we propose a novel SLST-PKNet model to overcome the limitations of applying traditional DNN based MTS prediction model to sales time series. First, SLST-PKNet contains three sub-models to take different types of MTS into consideration. Second, the design of TLCNN-TSLSTM combinations can model the future vision based dependent correlations in a more effective way, and capture dynamic changing patterns from sales time series, which traditional methods may fail. Third, A DCI component is designed to capture stable linear correlations between sales time series. Fourth, the excessive influence of noisy information was dealt with by using PK component, which provide supervised information to guide the model training, and reduce the influence of noise information.

The performance of SLST-PKNet was tested in a series of experiments, the findings of which support the view that the innovations incorporated in the model made a significant contribution to sales prediction based on data from two industrial datasets: Galanz and Cainiao, and two public datasets: Traffic and Exchange-Rate. The proposed model has also been verified in Galanz online test. It was applied to three months of new data from 2021 and improved MAE by an average of 10 % compared to existing prediction methods with human operator strategies.

There are still some limitations of the research that will be the focus of future research. The running speed of the proposed model is not high. Sparsity of sales time series is an issue, and may result in noisy information continuing to have excessive influence on the results. In the future, we will adopt new structures to accelerate the training speed. Meta-learning, and data argument will also be taken into consideration to capture more sales patterns. This, we anticipate, will further improve predictions.

CRediT authorship contribution statement

Daifeng Li: Conceptualization, Methodology, Formal analysis, Software, Investigation, Validation, Writing – original draft, Writing – review & editing. **Xuting Li:** Data curation, Software, Methodology, Formal analysis, Validation, Writing – original draft, Investigation. **Kaixin Lin:** Visualization, Investigation, Data curation, Formal analysis, Validation, Writing – original draft, Software. **Jianbin Liao:** Formal analysis, Data curation, Visualization. **Ruo Du:** Conceptualization, Methodology, Supervision. **Wei Lu:** Conceptualization, Methodology, Supervision. **Andrew Madden:** Formal analysis, Writing – review & editing.

Data availability

The data and code is introduced in the manuscript. The data and code could be visited at: https://github.com/lixt47/SLST-PKNet

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This research was supported by National Natural Science Foundation of China (NSFC) under Grant No. 72074231, Soft Science Foundation of Guangdong Province in China under Grant No. 2019A101002020, Key Project of Guangdong "Pandeng" program in China under Grant No. 20000-52910001.

References

[1] Baldi P, Sadowski P. Understanding Dropout. In Proceedings of Twenty-seventh Conference on Neural Information Processing Systems. Dec 5th-10th. Lake Tahoe. 2013.
[2] G.E. Box, G.M. Jenkins, G.C. Reinsel, G.M. Ljung, Time series analysis: forecasting and control, John Wiley & Sons, 2015.
[3] Cao D, Wang Y, Duan J, Zhang C, Zhu X, Huang C, Tong Y, Xu B, Bai J, Tong J. et al. Spectral temporal graph neural network for multivariate time-series forecasting. In Proceedings of 33rd Conference on NIPS. 2020.
[4] Chang Y, Sun F, Wu Y, Lin S. A Memory-Network based Solution for Multivariate Time-Series Forecasting. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence. Jan 27-Feb 1. Hawaii, USA. 2019.
[5] Chen J, Huang K, Zheng Z. Towards Better Forecasting by Fusing Near and Distant Future Vision. In Proceedings of Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI'20). 2020. February 7-12. New York. USA.
[6] Chung J, Gulcehre C, Cho K, and Bengio Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555, 2014.
[7] Dasgupta S and Osogami T. Nonlinear dynamic boltzmann machines for time-series prediction. In Proceedings of AAAI'17. 2017.
[8] Deng J, Chen X, Jiang R, Song X, Tsang I. ST-Norm: Spatial and Temporal Normalization for Multivariate Time Series Forecasting. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. 269-278. 2021.
[9] L. Du, R. Gao, P. Suganthan, D. Wang, Bayesian optimization based dynamic ensemble for time series forecasting, Information Sciences. 591 (2022) 155–175.
[10] Ekambaram V, Manglik K, Mukherjee S, Sajja S, Dwivedi S, Raykar V. Attention based Multi-Modal New Product Sales Time-series Forecasting. KDD 2020. August 23-27. USA.
[11] J.L. Elman, Finding structure in time, Cognitive science. 14 (2) (1990) 179–211.
[12] Faye B, Le F, Prat S. Exogeneous Shocks, Risk and Market Convergence of Real Alternative and Financial Assets: Evidence from Nonlinear Dynamics. Annals of Operations Research. 2022.
[13] Fan C, Zhang Y, Pan Y, Li X, Zhang C, Yuan R, Wu D, Wang W, Pei J, Huang H. Multi-Horizon Time Series Forecasting with Temporal Attention Learning. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. July, 2019. pp: 2527-2535.
[14] Franceschi J, Dieulevecut A, Jaggi M. Unsupervised Scalable Representation Learning for Multivariate Time Series. In Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS'19) 2019. Vancouver, Canada.
[15] Gao H, Kong D, Lu M, Bai X, Yang J. Attention Convolutional Neural Network for Advertiser-level Click-through Rate Forecasting. WWW'18. Spril 23-27. 2018. Lyon France.
[16] V. Guen, N. Thome, Shaped and Time Distortion Loss for Training Deep Time Series Forecasting Models, NeurIPS (2019).
[17] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation. 9 (8) (1997) 1735–1780.
[18] Huang S, Wang D, Wu X, Tang A. DSANet: Dual Self-Attention Network for Multivariate Time Series Forecasting. CIKM'19. November 3-7. 2019. Beijing China.
[19] K. Kıymet, Y. Yaren, Y. Yusuf, G. Şule, F. Çıngı, Demand forecasting model using hotel clustering findings for hospitality industry, Information Processing & Management 59 (2021).
[20] G. Lai, W. Chang, Y. Yang, H. Liu, Modeling Long and Short-Term Temporal Patterns with Deep Neural Networks, IGIR (2018).
[21] Laptev N, Yosinski J, Li L. E, Smyl S. Time-series Extreme Event Forecasting with Neural Networks at Uber. ICML 2017 Time Series Workshop. Sydney. Australia.
[22] D. Li, K. Lin, X. Li, J. Liao, R. Du, D. Chen, A. Madden, Improved sales time series predictions using deep neural networks with spatiotemporal dynamic pattern acquisition mechanism, Inf. Process. Manag. 59 (4) (2022) 102987.
[23] Lim B, Arik S. O, Loeff N, and Pfister T. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting. ArXiv191209363 Cs Stat, Sep. 2020, Accessed: Feb. 08, 2021. [Online]. Available: http://arxiv.org/abs/1912.09363.
[24] Liu S, Yu H, Liao C, Li J, Lin W, Liu A. X, et al. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In International conference on learning representations. 2021. URL https://openreview.net/forum?id=0EXmFzUn5I.
[25] H. Lütkepohl, New introduction to multiple time series analysis, Springer Science & Business Media, 2005.
[26] A. Mayer, Estimation and Inference in Adaptive Learning Models with Slowly Decreasing Gains, Journal of Time Series Analysis (2021).
[27] Nguyen L, Pan Z, Openiyi O, Abu-gellban A, Moghadasi M, Jin Fang. Self-boosted Time-series Forecasting with Multi-task and Multi-view Learning. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence. AAAI 2020. 7-12 February. NY. USA.
[28] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, G.W. Cottrell, A Dural-Stage Attention-Based Recurrent Neural Network for Time Series Prediction, IJCAI (2017).
[29] Oord A, Dieleman S, Zen H et al. WaveNet: A Generative Model for Raw Audio. In Proceedings of 9th ISCA Speech Synthesis Workshop 13-15 Sep 2016. Sunnyvale, USA. 2016.
[30] Oreshkin B, Carpov D, Chapados N, Bengio Y. N-BEATS: Neural Basis Expansion Analysis for Interpretable Time Series Forecasting. In the proceedings of the Eighth International Conference on Learning Representations (ICLR). 2020. Apr 26th – May 1st.
[31] J. Park, C. Park, J. Choi, S. Park, DeepGate: Global-local Decomposition for Multivariate Time Series Modeling, Information Sciences. 590 (2022) 158–178.
[32] S. Roberts, M. Osborne, M. Ebden, S. Reece, N. Gibson, S. Aigrain, Gaussian processes for time-series modelling, Phil. Trans. R. Soc. A. 371 (2013) 20110550.
[33] Sen R, Yu H, Dhillon I. Think Globally, Act Locally: A Deep Neural Network Approach to High-Dimensional Time Series Forecasting. In Proceedings of 33rd Conference on Neural Information Processing Systems (NeurlPS 2019). Vancouver, Canada.
[34] Shen Z, Yuan R, Wu D, Pei J. Data Science in Retail-as-a-Service. KDD 2018. London UK.
[35] Shi J, Yao H, Wu X, Li T. Relation-aware Meta-learning for Market Segment Demand Prediction with Limited Records. arXiv:2008.00181v1. 2021.
[36] Shi Q, Yin J, Cai J, Cichocki A, Yokota T, Chen L, Yuan M, Zeng J. Block Hankel Tensor ARIMA for Multiple Short Time Series Forecasting. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence. AAAI 2020. 7-12 February. New York. USA.
[37] Tang X, Yao H, Sun Y, Aggarwal C, Mitra P, Wang S. Joint Modeling of Local and Global Temporal Dynamic Multivariate Time Series Forecasting with Missing Values. In Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence. AAAI 2020. 7-12 February. New York. USA.
[38] Vaswani A, Shazeer N, Parmar N. Attention Is All You Need[J/OL]. ArXiv:1706.03762 [Cs], 2017[2021–02–28]. http://arxiv.org/abs/1706.03762.
[39] R. Wang, X. Pei, J. Zhu, Z. Zhang, X. Huang, J. Zhai, F. Zhang, Multivariable time series forecasting using model fusion, Information Sciences. 585 (2022) 262–274.
[40] Yakhchi, S., Behehsti, A., mohssen Ghafari, S., Razzak, I., Orgun, M., & Elahi, M. A convolutional attention network for unifying general and sequential recommenders. Information Processing & Management. 59. 2022.
[41] M. Ye, J. Luo, C. Xiao, F. Ma, LSAN: Modeling Long-term Dependencies and Short-term Correlations with Hierarchical Attention for Risk Prediction, CIKM (2020. October 19–23.).
[42] T.W. Yoo, I.S. Oh, Time Series Forecasting of Agricultural Products' Sales Volumes Based on Seasonal Long Short-Term Memory, APPLIED SCIENCES-BASEL. 10 (22) (2020).
[43] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H and Zhang W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In Proceedings of 35th AAAI Conference on Artificial Intelligence. Virtual Conference. Feb 2-9. 2021.