2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9378380

Combining Global and Sequential Patterns for Multivariate Time Series Forecasting

Zhaoxi Li[1], Jun He[1,†], Hongyan Liu[2,†] and Xiaoyong Du[1]
[1]*Key Laboratory of Data Engineering and Knowledge Engineering, Ministry of Education*, China
School of Information, Renmin University of China, Beijing, China
{lizhaoxi, hejun, duyong}@ruc.edu.cn
[2]*School of Economics and Management, Tsinghua University*, Beijing, China
liuhy@sem.tsinghua.edu.cn

***Abstract*—Multivariate time series forecasting is very important for many applications. Many studies have been conducted for accurate and interpretable prediction methods. However, existing methods either cannot take both times series and covariates into consideration, lacking of interpretability, or ignore global trends across multivariate time series. In this paper, we aim to solve these issues. To this end, we propose a new model named TEDGE for accurate and interpretable time series prediction. In this model, we extract global trends hidden across multivariate times series to improve prediction accuracy. Meanwhile, we utilize a deep recurrent model with attention mechanism to find long- and short-term sequential patterns hidden in individual time series with interpretability. We conduct experiments on several datasets to evaluate the proposed models performance. Results demonstrate the superior performance of our proposed model.**

***Index Terms*—Time Series Forecasting, Matrix Factorization, Deep Learning.**

I. Introduction

Multivariate time series forecasting is an important problem in real-world with many practical applications such as energy production of power plants [1], traffic loads [2], stock prices [3], and air pollution [4] forecasting. In these applications, there tends to be much correlated time series with tens of thousands of time steps that shares the same timeline. There are often some global trends among these time series. For example, in NASDAQ stock prices time series [3], stock prices of different corporations may follow a similar trend with the NASDAQ index. For the traffic loads scenario [2], observed loads of different sensors coevolve as the city becomes more and less crowded over time. Besides, covariates of each series cannot be ignored, because there may be rich external information about the time series. This leads to a problem of forecasting M time-series (one for each of the M items), given past values and covariates of this series over T time-steps. More importantly, interpretable results are regarded as important characteristics of forecasting models.

Many studies have been conducted for accurate and interpretable time series forecasting methods. Except for traditional linear models [5], [6], deep learning methods, such as CNN [2], [7]–[10], RNN [3], [11]–[14], Temporal Convolution Network (TCN) [15], [16] and attention mechanism [3], [7], [17], [18], are widely used on time series forecasting tasks. However, to the best of our knowledge, existing methods either cannot take both times series and covariates into consideration, lack interpretability, or ignore global trends hidden in multivariate time series.

Aiming to address these issues, we propose a new model for accurate and interpretable time series forecasting. In this model, a tensorized deep recurrent encoder-decoder framework named TED was developed with attention mechanism to capture long- and short-term temporal patterns hidden in series with better interpretability, taking covariates into consideration. Both temporal and variable importance are learned to explain which time step and which covariate is more significant to each prediction step. Meanwhile, inspired by [16], global trends across multivariate times series are extracted by an autoregression based matrix factorization method. These trends are then utilized to improve prediction accuracy. Matrix factorization component and tensorized recurrent encoder-decoder framework are integrated into a joint model for better forecasting. The joint model is named TEDGE.

To the best of our knowledge, our proposed model is the first explainable time series forecasting model, considering both global trends and sequential patterns. With experiments on several real-world datasets, our model shows superior performance and meaningful interpretability.

The rest of this paper is organized as follows. Section II outlines the related works, including models extracting global temporal patterns and models dealing with covariates of each series. Section III describes our proposed model TEDGE. Section IV shows the evaluation results of our model compared to strong baselines on three real-world datasets, and Section V concludes the paper finally.

II. Related Work

Many studies have been conducted for multivariable time series forecasting. Classic autoregression [5] and exponential smoothing [6] based models have been studied extensively. Besides, matrix factorization is also used for time series prediction. Yu et. al. [19] proposed a matrix factorization method to factorize high-dimension time series with temporal regularization constraints. In recent years, deep learning models were widely used in time series forecasting tasks. Lai et. al. [2] and Shih et. al. [7] combined CNN and RNN

†Corresponding authors.

to extract short-term and long-term patterns among multiple time series variables for one-step forecasting. Huang et. al. [8] proposed a model named DSANet, in which CNN is used to capture temporal patterns and self-attention is applied to model dependencies between multiple series. Chang et. al. [9] used convolution and memory component to capturing extremely long-term patterns. These methods could capture global patterns from target time series. But, they ignored the function of covariates in prediction.

RNN-based models such as DA-RNN [3] and DeepAR [11] were proposed for time series forecasting, which take covariates into consideration. Deep State Space Model (DSSM) [12] utilized LSTM to generate the parameters of linear state space model to generate multi-step probability forecasting. Fan et. al. [20] used multimodal attention mechanism to combine features of different history parts for better representation of future. Model DeepGLO [16] combined global trends, local properties and covariates in forecasting using temporal convolution models. More recently, Transformer architectures was explored in [17] for forecasting, and convolutional self-attention was developed to incorporate local context processing. But these methods fail to give interpretable evidence of results.

To interpretate forecasting results, some post-hoc explanation methods such as LIME [21] and SHAP [22], were proposed to explain forecasts from black-box machine learning models. However, these post-hoc methods focus on features importance, and may not be suitable for time series forecasting tasks due to ignoring sequence dependencies between inputs. Another choice is to design models with inherent explainable components. Oreshkin ei. al. [23] proposed model N-BEATS using stacked residual network for interpretable series forecasting, which is applicable to a wide range of domains. Assaf et. al. [10] converted forecasting problem into a classification task by dividing target value into several partitions, and proposed a gradient based method to generate heatmap, highlighting temporal importance of different covariates. Guo, Lin and Nino [18] proposed a model named IMV-LSTM, which adopted LSTM with tensorized hidden states and attention mechanism for one-step time series forecasting with interpretable results. Before this study, He et. al. [24] proposed a method named tensorized LSTM (tLSTM), which aimed to increase capacity of LSTM through tensorizing hidden states. But it was not designed to improve interpretability. Li et. al. [13] employed a deep state space model with Kullback-Leibler (KL) divergence as loss function, and weights of input covariates are learned as parameters to provide variable importance. Nevertheless, they didn't make full use of global patterns. To solve this issue, we propose to combine global trends across multiple time series with sequential patterns of each time series and develop a new model for multiple-step time series prediction.

III. Methodology

In this section, we formulate the multivariable time series forecasting problem and describe the proposed model. The design of our model allows us to give both accurate and explainable results of multivariable time series predictions.

TABLE I
Notation and Definitions

Notation	Definitions
T	Length of past known history
τ	Length of future horizon to be forecasted
$\boldsymbol{Y}_{1:T}$	M-dimension known time series, $\boldsymbol{Y}_{1:T} \in \mathbb{R}^{M\times T}$
$\hat{\boldsymbol{Y}}_{T+1:T+\tau}$	Forecasted time series, $\hat{\boldsymbol{Y}}_{T+1:T+\tau} \in \mathbb{R}^{M\times \tau}$
$\mathbf{y}^i$	The i-th variable series of $\boldsymbol{Y}$, $\mathbf{y}^i \in \mathbb{R}^T$
y_t^i	The i-th variable's value at time step t
$\mathbf{X}$	Covariates series, $\mathbf{X} \in \mathbb{R}^{M\times N\times (T+\tau)}$
$\boldsymbol{X}^i$	Covariates series associated with $\mathbf{y}^i$, $\boldsymbol{X}^i \in \mathbb{R}^{N\times (T+\tau)}$
$\mathbf{x}^{i,(j)}$	The j-th covariates series in $\boldsymbol{X}^i$, $\mathbf{x}^{i,(j)} \in \mathbb{R}^{T+\tau}$
$\mathbf{x}_t^i$	The value of $\boldsymbol{X}^i$ at step t in, $\mathbf{x}_t^i \in \mathbb{R}^N$

A. Problem Formulation

Consider we have M-dimension time series $\boldsymbol{Y}$ of past T time steps, $\boldsymbol{Y}_{1:T}$. $\mathbf{X}$ represents covariates series associated with $\boldsymbol{Y}$, whose future values are known or can be inferred, as shown in Table I. Our goal in this paper is to learn a model $\mathcal{F}$ to predict all time series' value in future τ steps, given their history $\boldsymbol{Y}_{1:T}$ and covariates $\mathbf{X}$, namely $\hat{\boldsymbol{Y}}_{T+1:T+\tau} = \mathcal{F}(\boldsymbol{Y}_{1:T}, \mathbf{X})$. Meanwhile, $\mathcal{F}$ is able to tell us which covariates at which steps contribute more to the prediction results.

The architecture of our proposed model is illustrated in Figure 2. The model consists of three modules: encoder, decoder, and global trends extraction.

B. Encoder: Tensorized LSTM with Hidden State Matrix

We use recurrent neural network to encode information of each time series $\mathbf{y}^i$ together with covariate $\boldsymbol{X}^i$, superscript i is omitted for simplicity in following parts. The traditional LSTM model is used to capture sequential information behind sequences and often used in times series prediction. However, traditional LSTM for time series forecasting blends the information of all covariates, which causes weak interpretability of forecasting results. We adopt tensorized LSTM as done by [18] to solve this problem. The main idea is to replace hidden state vector $\mathbf{h}_t$ at each time step t in LSTM with hidden state matrix $\boldsymbol{H}_t = [\mathbf{h}_t^1, \mathbf{h}_t^2, \cdots, \mathbf{h}_t^N]^\top \in \mathbb{R}^{N\times d}$, where $\mathbf{h}_t^j \in \mathbb{R}^d$, the j-th row of $\boldsymbol{H}_t$, means the hidden state relevant with $\mathbf{x}^{(j)}$.

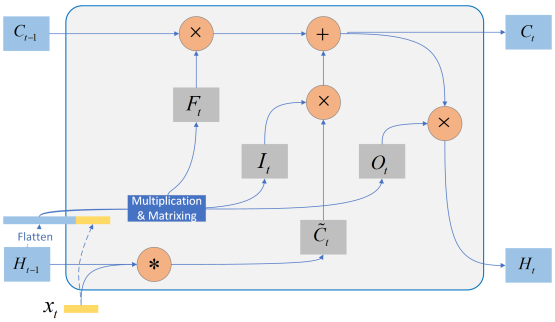


Fig. 1. Tensorized LSTM cell

The basic architecture of tensorized LSTM cell is illustrated in Figure 1. As shown in the figure, the formulations of tensorized LSTM cell are as follows:

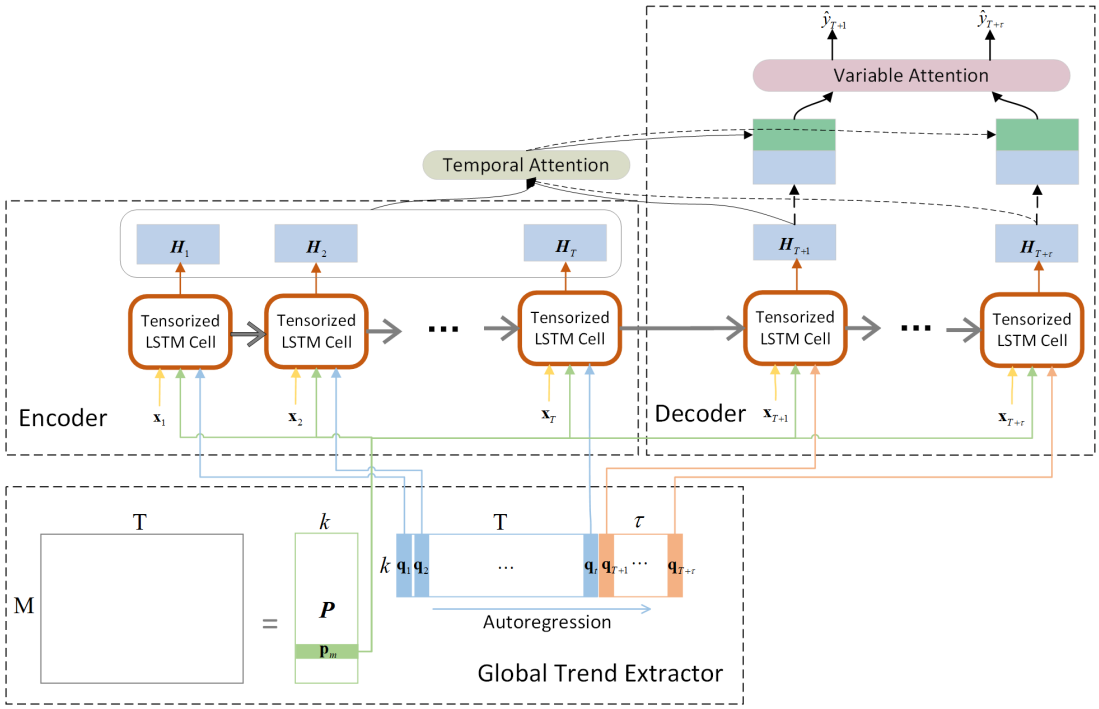


Fig. 2. The overview architecture of the proposed model

$$\begin{bmatrix} \boldsymbol{I}_t \\ \boldsymbol{F}_t \\ \boldsymbol{O}_t \end{bmatrix} = \text{mat}(\sigma(\begin{bmatrix} \boldsymbol{W}_i \\ \boldsymbol{W}_f \\ \boldsymbol{W}_o \end{bmatrix} [\mathbf{x}_t \oplus \text{vec}(\boldsymbol{H}_{t-1})] + \begin{bmatrix} \mathbf{b}_i \\ \mathbf{b}_f \\ \mathbf{b}_o \end{bmatrix})), \quad (1)$$

$$\tilde{\boldsymbol{C}}_t = \tanh(\mathbf{W}_c \otimes \boldsymbol{H}_{t-1} + \boldsymbol{U}_c \circledast \mathbf{x}_t + \boldsymbol{B}_c), \quad (2)$$

$$\boldsymbol{C}_t = \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + \boldsymbol{I}_t \odot \tilde{\boldsymbol{C}}_t, \quad (3)$$

$$\boldsymbol{H}_t = \boldsymbol{O}_t \odot \tanh(\boldsymbol{C}_t), \quad (4)$$

In Eq. (1), $\boldsymbol{I}_t, \boldsymbol{F}_t$ and $\boldsymbol{O}_t \in \mathbb{R}^{N \times d}$ are input, forget and output gate matrices corresponding to gate vectors $\mathbf{i}_t, \mathbf{f}_t$ and $\mathbf{o}_t$ in LSTM. $\boldsymbol{W}_i, \boldsymbol{W}_f$ and $\boldsymbol{W}_o \in \mathbb{R}^{Nd \times (Nd+1)}$ are weight matrices, and $\mathbf{b}_i, \mathbf{b}_f$ and $\mathbf{b}_o \in \mathbb{R}^{Nd}$ are bias vectors. $\text{vec}(\cdot)$ represents the operation that transforms a matrix into a vector by flattening a matrix in column order and $\text{mat}(\cdot)$ means reshaping the vector back to a matrix. $\sigma(\cdot)$ represents sigmoid function and operation $\oplus$ means concatenating two vectors.

In Eq. (2), $\mathbf{W}_c \in \mathbb{R}^{N \times d \times d}$ is a hidden state transition tensor, $\mathbf{W}_c = [\boldsymbol{W}_c^1, \cdots, \boldsymbol{W}_c^N]^\top$, where $\boldsymbol{W}_c^j \in \mathbb{R}^{d \times d}$. $\otimes$ is a tensor transition operation defined as: $\mathbf{W}_c \otimes \boldsymbol{H} = [\boldsymbol{W}_c^1 \boldsymbol{h}^1, \cdots, \boldsymbol{W}_c^N \boldsymbol{h}^N]^\top$. $\boldsymbol{U}_c \in \mathbb{R}^{N \times d}$ is an input transition matrix, $\boldsymbol{U}_c = [\mathbf{u}_c^1, \cdots, \mathbf{u}_c^N]^\top$, where $\mathbf{u}_c^j \in \mathbb{R}^d$, and $\circledast$ is an input transition operation defined as: $\boldsymbol{U}_c \circledast \mathbf{x}_t = [\mathbf{u}_c^1 x_t^1, \cdots, \mathbf{u}_c^N x_t^N]^\top$. In Eq. (3) and Eq. (4), $\odot$ represents element-wise production operation.

We abbreviate equations (1) - (4) as:

$$\boldsymbol{H}_t = \text{TLSTM}(\mathbf{x}_t, \boldsymbol{H}_{t-1}). \quad (5)$$

Notice that in Eq. (2), the candidate cell state, $\tilde{\boldsymbol{C}}_t = [\tilde{\mathbf{c}}_t^1, \cdots, \tilde{\mathbf{c}}_t^N]^T$, corresponds to cell state of each input, where $\tilde{\mathbf{c}}_t^j \in \mathbb{R}^d$ is obtained through processing $\mathbf{h}_{t-1}^j$ and $\mathbf{x}^j$. This process is independent from other inputs and ensures the interpretability of the model (details about interpretability will be further explained later).

C. Decoder

Tensorized LSTM is designed to encode the historical information of every time series independently. [18] adopts tensorized LSTM to predict the value of next time step $T+1$ for each time series. To predict values of multiple future steps, we design decoder with dual attentions to utilize both historical and future covariate information. We name this model TED (Tensorized LSTM with Encoder-Decoder framework).

In encoder part, we use TLSTM to extract features for each input represented by hidden state at each time step $\mathbf{H}_{1:T} = \{\boldsymbol{H}_1, \cdots, \boldsymbol{H}_T\}$, where $\boldsymbol{H}_t = \text{TLSTM}(\mathbf{x}_t, \boldsymbol{H}_{t-1})$. The decoder part shares the parameters of TLSTM to get hidden state matrices of future τ steps $\mathbf{H}_{T+1:T+\tau}$. At decoding step t from $T+1$ to $T+\tau$, hidden state matrix $\boldsymbol{H}_t$ encodes historical information and future information till time step t. Temporal attention is designed to know which historical step is more relevant with the time step t. For the j-th covariate, let $\alpha_{i,t}^j$ measures the relevance of its value at the time step i to its value at time step t, formulated as:

$$\begin{aligned} r_{i,t}^j &= \mathbf{v}^\top \tanh(\boldsymbol{W}_r[\mathbf{h}_i^j \oplus \mathbf{h}_t^j] + \mathbf{b}_r), \\ \alpha_{i,t}^j &= \frac{\exp(r_{i,t}^j)}{\sum_{k=1}^T \exp(r_{k,t}^j)}, \\ \mathbf{c}_t^j &= \sum_{i=1}^T \alpha_{i,t}^j \mathbf{h}_i^j, \end{aligned} \quad (6)$$

$\mathbf{c}_t^j$ is the context vector of the j-th covariate at decoding step t. Parameters $\mathbf{v} \in \mathbb{R}^d$ and $\boldsymbol{W}_r \in \mathbb{R}^{d\times 2d}$ are weight vector and weight matrix respectively. Concatenating the context vector and the current time step's hidden state, we get $\mathbf{g}_t^j = [\mathbf{h}_t^j \oplus \mathbf{c}_t^j]$, where $\mathbf{g}_t^j \in \mathbb{R}^{2d}$.

Given $\boldsymbol{G}_t = [\mathbf{g}_t^1, \cdots, \mathbf{g}_t^N]^\top$, we predict the target value at future time step t based on each j-th covariate's state $\mathbf{g}_t^j$, denoted by μ_t^j:

$$\mu_t^j = \mathbf{w}_\mu^\top \mathbf{g}_t^j + b_\mu. \tag{7}$$

Then variable attention is defined to combine these predictions:

$$\begin{aligned} s_t^j &= \mathbf{w}_s^\top \mathbf{g}_t^j + b_s, \\ \beta_t^j &= \frac{\exp\left(s_t^j\right)}{\sum_{n=1}^N \exp\left(s_t^n\right)}, \end{aligned} \tag{8}$$

$$\hat{y}_t = \sum_{j=1}^N \beta_t^j \mu_t^j, \tag{9}$$

$\hat{y}_t$ is the weighted sum of prediction results based on each covariate at step t.

By decoding in this way step by step, we get final predictions of i-th series $\hat{\mathbf{y}}_{T+1:T+\tau}^i$. Prediction of all series can be bound as $\hat{\boldsymbol{Y}}_{T+1:T+\tau} = [\hat{\mathbf{y}}_{T+1:T+\tau}^1, \cdots, \hat{\mathbf{y}}_{T+1:T+\tau}^M]^T$. Temporal importance $\boldsymbol{A} \in \mathbb{R}^{N\times T}$ and variable importance $\boldsymbol{\beta} \in \mathbb{R}^N$ are also available. The former explains which time steps are more relevant to each prediction, while the latter explains which covariate is more significant to each prediction.

D. Global Trends Extraction

With the model TED, we can predict values of multiple future steps for each time series variable based on its covariates and historical time series values. In real-world time series applications, there are often much correlated time series that share the same timeline. For example, many stock prices change dynamically at the same time every day and electricity loads of tens of thousands of houses coevolve every day. To find the global evolution trends across multivariate time series, Yu et. al. [19] proposed a matrix factorization method with temporal regularization (TRMF). The main idea of TRMF is to factorize historical series matrix $\boldsymbol{Y}_{1:T} \in \mathbb{R}^{M\times T}$ into low-rank factors, $\boldsymbol{P} \in \mathbb{R}^{M\times d_k}$ and $\boldsymbol{Q}_{1:T} \in \mathbb{R}^{d_k\times T}$, where M is the number of time series and $d_k \ll M$. $\boldsymbol{P}$ contains variable features and $\boldsymbol{Q}$ contains global change patterns. By employing temporal dependence in $\boldsymbol{Q}$, future time patterns $\boldsymbol{Q}_{T+1:T+\tau}$ can be inferred. Predictions are then obtained:

$$\hat{\boldsymbol{Y}}_{T+1:T+\tau} = \boldsymbol{P}\ \boldsymbol{Q}_{T+1:T+\tau}. \tag{10}$$

Temporal dependence is incorporated in the loss function of TRMF:

$$\begin{aligned} \mathcal{L}_{MF} &= \sum_{m,t} (y_{m,t} - \mathbf{p}_m^\top \mathbf{q}_t)^2 + \lambda_{\mathrm{r}} \mathcal{R}(\boldsymbol{Q}) + \lambda_p \|P\|_2 + \lambda_q \|Q\|_2, \\ \mathcal{R}(\boldsymbol{Q}) &= \frac{1}{2} \sum_t \|\mathbf{q}_t - \sum_l w_{\mathrm{r}}^l \mathbf{q}_l\|^2 + \lambda_w \|\mathbf{w}_{\mathrm{r}}\|_2, \end{aligned} \tag{11}$$

$\mathcal{R}(\boldsymbol{Q})$ is an autoregressive temporal regularizer, which uses $\boldsymbol{Q}$ at previous steps to infer $\mathbf{q}_t$ via linear autoregressive approach, $\mathbf{w}_{\mathrm{r}}$ is the autoregressive weights vector, and $\|\cdot\|_2$ is L2 norm.

However, this method only utilizes time series information, and can not utilize covariates $\boldsymbol{X}$ for better prediction. While model TED neglects global evolution trends hidden in multiple time series. To solve this problem, we propose to incorporate the global evolution trends with TED. We name this model TEDGE (Tensorized LSTM with Encoder-Decoder framework and Global trEnd).

In our model, we adopt TRMF to mine both global trends and variable features. For the m-th time series, both series patterns $\mathbf{p}_m$ and global trends $\mathbf{q}_t$ together with other covariates are fed into the model TED as input in each time step. Series pattern $\mathbf{p}_m$ can be regarded as embedding of each series to capture sequential pattern hidden in each time series, and global trend $\mathbf{q}_t$ show common trend across multiple times series. Notice that in encoder, $\mathbf{q}_t$ is the factorization result, and during decoding stage $\mathbf{q}_t$ is inferred by previous patterns, which avoids information leakage. More details about combining both components are shown in Section III-E.

E. Training Process

The overall model training process is as follows:

- Train TRMF model to predict $\boldsymbol{Y}_{T+1:T+\tau}$, taking $\boldsymbol{Y}_{1:T}$ as input, and obtain output $\boldsymbol{P}, \boldsymbol{Q}_{1:T}$ and $\mathbf{w}_r$.
- Use $\mathbf{w}_r$ and $\boldsymbol{Q}_{1:T}$ to generate future time patterns $\boldsymbol{Q}_{T+1:T+\tau}$.
- Concatenate $\boldsymbol{Q}' = [\boldsymbol{Q}_{1:T}, \boldsymbol{Q}_{T+1:T+\tau}]$, $\boldsymbol{P}$ with covariates $\boldsymbol{X}_{1:T+\tau}$ to get the final input of TED.
- Feeding the input to TED model, we get output $\hat{\boldsymbol{Y}}_{T+1:T+\tau}$. With the output, loss function that is based on mean square error (MSE) can be calculated:

$$\mathcal{L}(\boldsymbol{Y}, \hat{\boldsymbol{Y}}) = \frac{1}{M\times\tau} \sum_{m=1}^M \sum_{t=T+1}^{T+\tau} (y_t^m - \hat{y}_t^m)^2. \tag{12}$$

  The parameters of the model TED are then learned via Adam optimizer based on this loss function.

Finally, we get model TRMF to extract global trends, and model TED to give forecasting results. By combining these two models into TEDGE, global trends across multiple time series and sequential patterns of each time series are combined for accurate prediction.

IV. Experiments

In this section, we evaluate our model on three real-world datasets with rolling-window forecasting tasks against different kinds of baseline methods.

A. Datasets Description

The three datasets we used are summarized in Table II:

- **Live600** is a dataset provided by a live streaming platform. The target is to predict the daily total watched durations of 623 anchors in December 2017. This dataset

TABLE II
SUMMARIZATION OF DATASETS

Datasets	Length	Series	Covariates	Window	Horizon
Live600	31	623	54	10	7
PM2.5	43824	1	10	72	24
Electricity	26304	321	0	72	24

contains rich covariates (such as tags provided by watchers, anchor occupations and live categories, etc.), These anchors may follow some similar global trends.

- **PM2.5** is a public dataset provided by [4], including hourly observed PM2.5 value series from 2010 to 2014, together with meteorological conditions (such as temperature, pressure, and wind, etc.) as covariates. It totally contains 43824 time steps.
- **Electricity** contains hourly electricity consumption of 321 clients from 2012 to 2014, and is used in [2]. There are no covariates in this dataset.

B. Methods for Comparison

We compare our proposed model with the following baseline models:

- **ARIMAX** autoregressive integrated moving average (ARIMA) model taking covariates into consideration [5]. It is a linear baseline.
- **DSSM** [12] is a probabilistic forecasting model, using RNN to model parameters of linear state space model.
- **IMV-LSTM** [18] adopts tensorized LSTM and attention mechanism to give interpretable forecasting results.
- **TRMF** [19] is a temporal regularization based matrix factorization (MF) method, which uses autoregression to forecast future time latents features.
- **LSTNet** [2] uses CNN to extract global features and uses RNN to model temporal dependency. A skip strategy is designed to capture periodic patterns.
- **DSANet** [8] applies two CNN architectures to extract both global and local temporal patterns, and self-attention mechanism to model dependencies among series.
- **DeepGLO** [16] uses temporal convolution network (TCN) to model temporal regularization in MF, and employs another TCN to forecast future time series values, taking prediction of MF, true value at previous step and covariates as inputs.
- **TCN-MF** is the matrix factorization part in [16].

Among these baselines, TRMF, LSTNet, DSANet and TCN-MF only take all time series $\boldsymbol{Y}$ as input and do not consider covariates, and $\hat{\boldsymbol{Y}}$ is generated together as a matrix. we call this group of models as global models. ARIMAX, DSSM, IMV-LSTM and TED consider both time series and covariates, but ignore global patterns. We call them local models. For local models, $\boldsymbol{X}$ and $\mathbf{y}$ in each series are fed into model individually, and $\hat{\mathbf{y}}$ is also generated per series. DeepGLO combines both global trends and local properties of time series. Our proposed model TEDGE considers both global trends and long- and short-term dependency of times series. We call these two models as mixed models. Since DSSM and IMV-LSTM are probabilistic forecasting models, we use the estimated mean value as prediction results.

C. Experiment Setting

We focus on rolling-window forecasting on these datasets. Taking dataset Live600 as an example, we feed the series in a 10-time step window as input, and predict the time series in the following horizon of 7 steps. In another word, we use data of previous 10 days to predict the next week. As for PM2.5 and Electricity datasets, we want to forecast data on next day with data 3 days before, so we use 72 as window size and 24 as horizon size, as shown in Table II.

By rolling the window, we get the forecasting results of all steps in the test set. For one-step baselines *e.g.* IMV-LSTM, LSTNet and DSANet, we feed the hidden state obtained at the last step into a feed-forward network to predict all results in next seven days. Results of other multipie-step baselines are generated naturally.

Live600 dataset is split into training set (80%), validation set(13.3%) and test set(6.7%) because of dataset length limit, the other two datasets are split as 60%, 20% and 20%. All the inputs are normalized into range $[-1, 1]$. Except for the covariates provided in the datasets, we also take time features and target value at previous step as inputs of each time step. Time features include month of year, day of month, day of week, is weekend, hour of day and is business hour (from 9:00 to 17:00).

Hyper-parameters of our model and baselines are well tuned on the validation set. As for our proposed model, learning rate is set as 0.001, and Adam is used as optimization method to learn the parameters of TED. The hidden state dimension d is set as 12, and dropout rate is set as 0.2. For global trends extraction, d_k is set as 5. We run 100 epochs and use the best epoch on validation set to perform predictions on the test dataset. For other baselines, we use the same learning rate and optimization method.

We adopt two metrics to measure each model's performance for time series forecasting tasks:

- Root Relative Squared Error (RRSE):

$$\text{RRSE} = \frac{\sqrt{\sum_{m=1}^{M}\sum_{t=T+1}^{T+\tau}(y_t^m - \hat{y}_t^m)^2}}{\sqrt{\sum_{m=1}^{M}\sum_{t=T+1}^{T+\tau}(y_t^m - \text{mean}(\mathbf{y}^m)^2}} \tag{13}$$

- Normalized Mean Absolute Error (NMAE):

$$\text{NMAE} = \frac{\sum_{m=1}^{M}\sum_{t=T+1}^{T+\tau}|y_t^m - \hat{y}_t^m|}{\sum_{m=1}^{M}\sum_{t=T+1}^{T+\tau}|y_t^m|} \tag{14}$$

RRSE and NMAE are the scaled version of widely used RMSE and MAE, for RMSE and MAE become unreadable with varying value ranges of $\boldsymbol{Y}$ and $\hat{\boldsymbol{Y}}$.

TABLE III
PERFORMANCE COMPARISON WITH BASELINES ON THREE DATASETS.

Method		Live600		PM2.5		Electricity	
		NMAE	RRSE	NMAE	RRSE	NMAE	RRSE
Global models	TRMF	0.4538	2.5910	–	–	0.0884	0.0152
	TCN-MF	0.4520	3.0365	–	–	0.2785	0.0755
	LSTNet	0.3834	1.8422	–	–	0.0997	0.0166
	DSANet	0.4111	1.9615	–	–	0.1167	0.0175
Local models	ARIMAX	0.4853	3.8637	0.8337	7.8195	0.5102	2.5531
	DSSM	0.2994	0.5515	**0.1175**	**0.1919**	**0.0508**	0.0162
	IMV-LSTM	0.4164	1.6058	0.4836	2.2092	0.0951	0.0182
	TED	0.3077	0.6654	**0.1177**	**0.1939**	0.0623	0.0085
Mixed models	DeepGLO	0.4111	1.9615	0.3470	1.7998	0.0998	0.0412
	TEDGE	**0.2324**	**0.4105**	–	–	0.0570	**0.0083**

D. Comparison with Baselines

The results of all models on all datasets are summarized in Table III. TEDGE is our proposed model, and TED is a simplified version without the global trends extraction component. First, as can be seen from Table III, LSTNet shows better performance than other global baselines. Since there is only one single series in PM2.5 dataset, global models and mixed models become unsuitable for such situation. Second, among local models, TED and DSSM consistently yield better performance than ARIMAX and IMV-LSTM on three datasets. What's more, with same inputs, TED is able to explain the forecasting results via temporal importance and variable importance. IMV-LSTM can provide explainable results but fails to give such accurate forecasting. Third, TEDGE enhances the performance of TED by importing global trends, and interpretability remains. It provides the best forecasting on Live600 dataset, and similar accuracy with DSSM on Electricity dataset. DeepGLO shows accuracy results inferior to TEDGE and has low interpretability.

To summarize, TED, the simplified model of TEDGE, can give accurate forecasting results as well as interpretable ability. TEDGE, being able to combine both global trends cross time series with matrix factorization and long-short temporal dependency remaining in the tensorized LSTM, achieves the best performance in terms of accuracy and interpretability.

E. Ablation Study

1) Effect of Global Trends Component: In our proposed model, we introduce global trends component through matrix factorization method. To evaluate the effect of the global trends component, we conduct an ablation study by removing this component from TEDGE. As described in Section III-D, we factorize historical time series matrix $\boldsymbol{Y}_{1:T} \in \mathbb{R}^{M\times T}$ into two factors, $\boldsymbol{P} \in \mathbb{R}^{M\times d_k}$ and $\boldsymbol{Q}_{1:T} \in \mathbb{R}^{d_k\times T}$, where $\boldsymbol{P}$ contains variable latent features and $\boldsymbol{Q}$ contains global change patterns. We implemented three simplified models: TEDGE-V and TEDGE-G remove variable latent features $\boldsymbol{P}$ and global change patterns $\boldsymbol{Q}$ from the inputs of TEDGE respectively. And TED remove both $\boldsymbol{P}$ and $\boldsymbol{Q}$ from the inputs of TEDGE.

TABLE IV
EXPERIMENT RESULTS OF THE EFFECT OF GLOBAL TRENDS COMPONENT ON LIVE600 DATASET.

	NMAE	RRSE
TEDGE	0.2324	0.4105
TEDGE-V	0.2630	0.5726
TEDGE-G	0.2417	0.4880
TED	0.3077	0.6654

TABLE V
EXPERIMENT RESULTS OF THE EFFECT OF ATTENTION COMPONENT ON LIVE600 DATASET.

	NMAE	RRSE
TEDGE	0.2324	0.4105
TEDGE-TA	0.2341	0.4730
TEDGE-VA	0.2421	0.4291
TEDGE-NA	0.2530	0.4759

The results on the live600 dataset are shown in Table IV. We can see from the table that both variable features and global trends contribute to forecasting results. Moreover, global trends are more important than variable features. This might be because that global trends contain common trends over time across all time series, which is hard to be captured through dealing with individual time series.

2) Effect of Attention Component: We first examine the effect of attention component. In our proposed model we have applied two kind of attention mechanisms in decoder part, namely temporal and variable attention. Here, we consider three simplified models:

- TEDGE-TA: TEDGE with only temporal attention (TA), which uses hidden state matrix at current step $\boldsymbol{H}_t$ to calculate $\mathbf{g}_t$ without previous hidden state matrices $\mathbf{H}_{1:T}$.
- TEDGE-VA: TEDGE with only variable attention (VA), which gives predictions by averaging $\boldsymbol{\mu_t}$ with no weights.
- TEDGE-NA: TEDGE without any attention.

Experiment results on Live600 dataset are shown in Table V. It can be seen that each kind of attention improve the accuracy

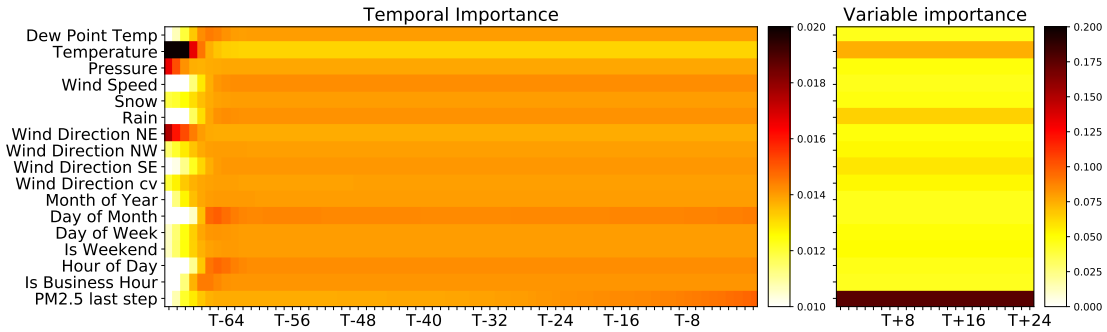


Fig. 3. Average temporal and variable importance on prediction of PM2.5 dataset.

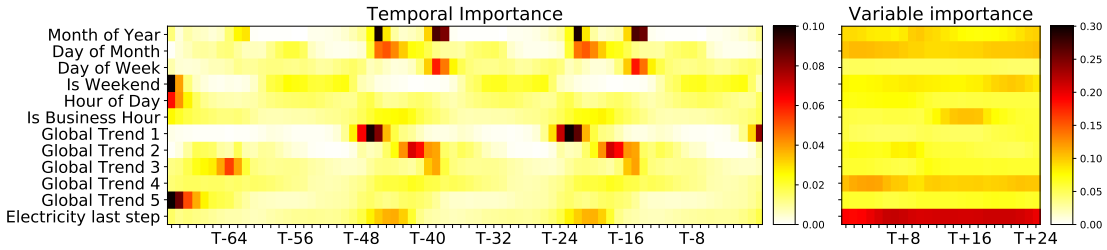
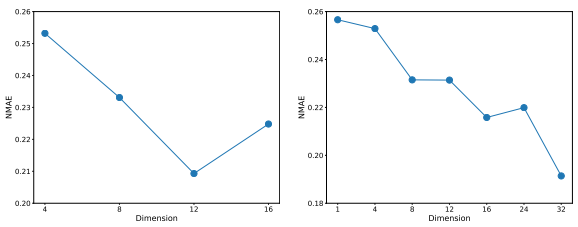
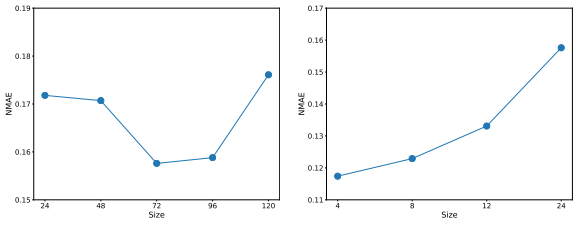


Fig. 4. Average temporal and variable importance on prediction of electricity dataset.



(a) Dimension of hidden states (b) Dimension of global trends

Fig. 5. NMAE performence with different dimension on Live600 dataset.



(a) Encoder window (b) Decoder horizon

Fig. 6. NMAE performence with different size on PM2.5 dataset.

and utilizing two attentions improves further.

F. Hyper-parameter Analysis

We conduct experiments to study the impact of different hyper-parameter values. In each group of the following experiments, we only change one parameter and fix the others. Metric NMAE is used here to compare the results.

1) Impact of different dimensions: There are two dimension hyper-parameters of our proposed model. One is the dimension d of hidden states in tensorized LSTM. Figure 5(a) shows the normalized mean absolute error (NMAE) as the hidden state dimension increases on Live600 dataset. As we can see from the figure, as the dimension increases from 4 to 12, the error become lower and lower, and then it increases. Therefore, the best dimension is 12 for this dataset. is 12. The other dimension is the latent factor dimension of matrix factorization, d_k. Figure 5(b) shows the normalized mean absolute error as d_k increases on Live600 dataset. As the figure shows that as the dimension becomes larger, the performance becomes better.

2) Impact of window and horizon size: Figure 6 shows the performance of different encoder window and decoder horizon size on PM2.5 dataset, as this dataset has the longest time series. As shown in Figure 6(a), 72 hours (3 Days) is the best encoder window size. A possible explanation is that a too short window may neglect useful information while a too long window may import much information irrelevant with the future. What's more, Figure 6(b) indicates that larger decoder horizon leads to worse prediction results. It is reasonable, as it is more difficult to predict further into the future.

G. Interpretability Analysis

Figure 3 shows the average temporal and variable importance on the PM2.5 dataset among all series. The left figure is temporal importance of all covariates at the first forecasting step. We can find that long history of "Temperature", "Pressure" and "Wind Direction NE" contributes more. Taking covariate "Temperature" as an example, variable importance values at first four steps are much higher than other steps, which means series values at the start of encoder window contribute more to the context vector representing this covariate

as shown in Eq. (6). As for "PM2.5 last step", it's importance becomes higher over time, which means the short history of this covariate is more relevant to forecasting. The right figure shows variable importance at each step in prediction horizon (from $T+1$ to $T+\tau$). As we can see from the figure, variable importance doesn't vary much among different decoder steps. "PM2.5 last step" is the most significant covariate with mean importance of 0.176. That's to say, this covariate contributes the most to the final predictions. Besides, meteorological conditions such as temperature, rain, and wind direction are also much relevant to PM2.5 forecasting.

Figure 4 shows the average temporal and variable importance on the electricity dataset, 5 global trends are introduced as input. As can be seen from the figure, different global trends show different temporal importance, and obvious period patterns are observed with an interval of 24 steps (one day). The 5th global trend is the most significant one among all global trends, which indicates that many series follow this trend. Though there is no covariate contained in the original dataset, we add time features and target value at previous step as inputs. Among these features, "Electricity last step" is the most relevant covariate. Interestingly, "Is Weekend" is more important in the tail of horizon, while "Is Business Hour" contributes more in middle steps.

V. Conclusion

In this paper, we propose a new model named TEDGE for accurate and interpretable time series forecasting. Through a tersorized deep recurrent encoder-decoder model with attention mechanism to capture long- and short term temporal dependency in time series, together with global trends extracted by an autoregression based matrix factorization method, our proposed model can generate both accurate and explainable forecasting results. With experiments on several real-world datasets, our proposed model shows superior performance with good interpretability compared with state-of-the-art models. Ablation study demonstrates the effectiveness of the global trend incorporated into the recurrent encoder-decoder model and the benefits of attention mechanism used.

Acknowledgment

This work was supported by National Natural Science Foundation of China under grant numbers U1711262, 61472428 and 71771131.

References

[1] M. Ceci, R. Corizzo, F. Fumarola, D. Malerba, and A. Rashkovska, "Predictive modeling of pv energy production: How to set up the learning task for a better prediction?" *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 956–966, 2016.

[2] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 95–104.

[3] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, 2017, pp. 2627–2633.

[4] X. Liang, T. Zou, B. Guo, S. Li, H. Zhang, S. Zhang, H. Huang, and S. X. Chen, "Assessing beijing's pm2. 5 pollution: severity, weather impact, apec and winter heating," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 471, no. 2182, p. 20150257, 2015.

[5] J. Durbin and S. J. Koopman, *Time series analysis by state space methods*. Oxford university press, 2012.

[6] R. Hyndman, A. B. Koehler, J. K. Ord, and R. D. Snyder, *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media, 2008.

[7] S.-Y. Shih, F.-K. Sun, and H.-y. Lee, "Temporal pattern attention for multivariate time series forecasting," *Machine Learning*, vol. 108, no. 8-9, pp. 1421–1441, 2019.

[8] S. Huang, D. Wang, X. Wu, and A. Tang, "Dsanet: Dual self-attention network for multivariate time series forecasting," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 2129–2132.

[9] Y.-Y. Chang, F.-Y. Sun, Y.-H. Wu, and S.-D. Lin, "A memory-network based solution for multivariate time-series forecasting," *arXiv preprint arXiv:1809.02105*, 2018.

[10] R. Assaf, I. Giurgiu, F. Bagehorn, and A. Schumann, "Mtex-cnn: Multivariate time series explanations for predictions with convolutional neural networks," in *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2019, pp. 952–957.

[11] D. Salinas, V. Flunkert, J. Gasthaus, and T. Januschowski, "Deepar: Probabilistic forecasting with autoregressive recurrent networks," *International Journal of Forecasting*, 2019.

[12] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *Advances in Neural Information Processing Systems*, 2018, pp. 7785–7794.

[13] L. Li, J. Yan, X. Yang, and Y. Jin, "Learning interpretable deep state space model for probabilistic time series forecasting," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 2019, pp. 2901–2908.

[14] R. Wen, K. Torkkola, B. Narayanaswamy, and D. Madeka, "A multi-horizon quantile recurrent forecaster," *arXiv preprint arXiv:1711.11053*, 2017.

[15] X. Jin, S. Li, Y. Zhang, and X. Yan, "Multi-step deep autoregressive forecasting with latent states," in *URL http://roseyu. com/time-series-workshop/submissions/2019/timeseries-ICML19_paper_19. pdf*, 2019.

[16] R. Sen, H.-F. Yu, and I. S. Dhillon, "Think globally, act locally: A deep neural network approach to high-dimensional time series forecasting," in *Advances in Neural Information Processing Systems*, 2019, pp. 4837–4846.

[17] S. Li, X. Jin, Y. Xuan, X. Zhou, W. Chen, Y.-X. Wang, and X. Yan, "Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting," in *Advances in Neural Information Processing Systems*, 2019, pp. 5243–5253.

[18] T. Guo, T. Lin, and N. Antulov-Fantulin, "Exploring interpretable lstm neural networks over multi-variable data," in *International Conference on Machine Learning*, 2019, pp. 2494–2504.

[19] H. Yu, N. Rao, and I. S. Dhillon, "Temporal regularized matrix factorization for high-dimensional time series prediction," in *Advances in Neural Information Processing Systems*, 2016, pp. 847–855.

[20] C. Fan, Y. Zhang, Y. Pan, X. Li, C. Zhang, R. Yuan, D. Wu, W. Wang, J. Pei, and H. Huang, "Multi-horizon time series forecasting with temporal attention learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2527–2535.

[21] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[22] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, 2017, pp. 4765–4774.

[23] B. N. Oreshkin, D. Carpov, N. Chapados, and Y. Bengio, "N-BEATS: neural basis expansion analysis for interpretable time series forecasting," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020.

[24] Z. He, S. Gao, L. Xiao, D. Liu, H. He, and D. Barber, "Wider and deeper, cheaper and faster: Tensorized lstms for sequence learning," in *Advances in Neural Information Processing Systems*, 2017, pp. 1–11.