Large-scale User Visits Understanding and Forecasting with Deep Spatial-Temporal Tensor Factorization Framework

Xiaoyang Ma
xiaoyangma@tencent.com
Tencent

Lan Zhang*
zhanglan03@gmail.com
University of Science and Technology of China

Lan Xu
lanxu@tencent.com
Tencent

Zhicheng Liu
zhichengliu@tencent.com
Tencent

Ge Chen
gechen@tencent.com
Tencent

Zhili Xiao
tomxiao@tencent.com
Tencent

Yang Wang
reganwang@tencent.com
Tencent

Zhengtao Wu
wzt@mail.ustc.edu.cn
University of Science and Technology of China.

ABSTRACT

Understanding and forecasting user visits is of great importance for a variety of tasks, e.g., online advertising, which is one of the most profitable business models for Internet services. Publishers sell advertising spaces in advance with user visit volume and attributes guarantees. There are usually tens of thousands of attribute combinations in an online advertising system. The key problem is how to accurately forecast the number of user visits for each attribute combination. Many traditional work characterizing temporal trends of every single time series are quite inefficient for large-scale time series. Recently, a number of models based on deep learning or matrix factorization have been proposed for high-dimensional time series forecasting. However, most of them neglect correlations among attribute combinations, or are tailored for specific applications, resulting in poor adaptability for different business scenarios. Besides, sophisticated deep learning models usually cause high time and space complexity. There is still a lack of an efficient highly scalable and adaptable solution for accurate high-dimensional time series forecasting. To address this issue, in this work, we conduct a thorough analysis on large-scale user visits data and propose a novel deep spatial-temporal tensor factorization framework, which provides a general design for high-dimensional time series forecasting. We deployed the proposed framework in Tencent online guaranteed delivery advertising system, and extensively evaluated the effectiveness and efficiency of the framework in two different large-scale application scenarios. The results show that our framework outperforms existing methods in prediction accuracy. Meanwhile, it significantly reduces the parameter number and is resistant to incomplete data with up to 20% missing values.

CCS CONCEPTS

• **Information systems** → **Computational advertising**;
• **Theory of computation** → **Models of learning**.

KEYWORDS

User Visits Forecasting; High-dimensional Time Series Forecasting; Guaranteed Delivery Advertising

ACM Reference Format:
Xiaoyang Ma, Lan Zhang, Lan Xu, Zhicheng Liu, Ge Chen, Zhili Xiao, Yang Wang, and Zhengtao Wu. 2019. Large-scale User Visits Understanding and Forecasting with Deep Spatial-Temporal Tensor Factorization Framework. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330728

1 INTRODUCTION

Deep understanding and accurate forecasting of user visits can significantly benefit most online systems, such as online advertising, recommendation, etc. We take user visits prediction for online guaranteed delivery advertising (also known as agreement-based advertising) as our motivating application, which is one of the most important sources of income for the Internet industry. A typical payment model is that advertisers are charged for stationary advertising spaces with specific user visits guarantees, including both visitor volume and attributes (e.g., "Female" visitors from "New York", who are visiting "News" pages) guarantees. Publishers sell advertising spaces and user visits in advance, as a consequence, publishers' revenue relies heavily on the ability of predicting the number of user visits under different

*Lan Zhang is the corresponding author.

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330728

attribute combinations. Inaccurate prediction would cause under-deliver or over-deliver, which results in economic loss of both advertisers and publishers. Even one percent accuracy improvement matters for the revenue. Formally, user visits are time series with different attribute combinations. There are usually tens of thousands of attribute combinations in an online advertising system. In this work, we try to achieve a deep understanding of large-scale user visits, as well as accurate forecasting of high-dimensional time series, which is also a fundamental but very challenging issue in many other areas, e.g., finance and traffic management.

Various models have been designed to forecast time series. Many traditional work model temporal trends of each time series separately, e.g., ARIMA [3], GARCH [4] and TBATS with regressors method [11].Those model are inefficient to process large-scale time series. Besides, they fall short of handling long-term properties and incomplete data with a large portion of missing values. Recently, state-of-the-art methods based on matrix factorization [17, 21] or deep learning [14, 20, 22] have been developed for high-dimensional time series forecasting. However, most of those models focus on temporal properties but neglect spatial correlations among attribute combinations. Some models employing CNN layers consider spatial dependencies, but neglect long-term temporal features like yearly trend and period, e.g., the CTR prediction model [9]. Moreover, most models are designed for specific applications and make assumptions on the attribute dimension and spatial correlations, that results in poor adaptability for different scales and spatial correlations of diverse business scenarios. For example, traffic flow prediction models [22, 23] are only suitable for two-dimensional spatial data (e.g., the GPS data), and not applicable to user visits prediction. In addition, sophisticated deep learning models usually cause high time and space complexity. Therefore, there is still a lack of an efficient highly scalable and adaptable framework for accurate high-dimensional time series forecasting.

It is very challenging to achieve such a framework due to the following complex factors:

1. The number of attribute combinations is large and dynamic. Most online systems use a combination of attributes to describe a user visit (e.g., platform, content, area, gender and age group), which results in tens of thousands of time series to be predicted. Moreover, the attribute combinations are very dynamic and usually continuously increasing due to the change of business requirements and system design. It thus requires the forecasting model to be highly efficient and scalable.

2. Implicit correlations among diverse attributes. As we will present in Section 3, there are significant correlations among different attributes and their combinations. Failing to capture these correlations restricts the prediction accuracy as shown in the evaluation section. Considering diverse forms and meanings, how to model the implicit correlations among various attributes remains a challenging issue.

3. Long-term temporal properties and missing values. Based on our data analysis in Section 3, temporally, user visits have long-term properties and short-term properties. In many business scenarios, there are years of historical data. How to utilize those historical data and model long-term and short-term properties seamlessly is non-trivial. Besides, in practice, there are usually some missing values caused by networking or exceptions. It also raises a challenge to the robustness of the forecasting model.

In this work, we aim to solve aforementioned challenges and avoid task-specific design by proposing a novel deep spatial-temporal tensor factorization framework. Our main contributions are summarized as follows:

•We conduct a thorough analysis on large-scale real user visits data and explore both spatial and temporal properties, which provide a deep understanding of user visits and guide the design of user visit forecasting model.

•Our framework provides a general design for high-dimensional time series forecasting. It is capable of capturing implicit correlations among diverse attributes, characterizing long-term and short-term temporal properties, and resistant to missing values. It is highly scalable, adaptable and efficient, and can be adopted by various applications, e.g, online advertising and traffic management.

•We have deployed the proposed framework in Tencent online advertising system for more than 5 months, which has brought considerable revenue growth. To compare with other methods, we implemented 6 diverse state-of-the-art models and conducted extensive evaluations using two large-scale real-world datasets. The larger one has three year anonymized user visits data with billions of user visits per day. The smaller one has more than one year traffic data generated by 963 detectors. Our framework outperforms all existing models on both datasets. On the larger one, our method provides a 8.7% relative improvement over the CNN based model, and a 5.8% relative improvement over the matrix factorization based method [21]. There is only a slight accuracy loss (0.001 normalized deviation decrease), when there are 20% missing values. Our model performs even better on the smaller dataset. It also significantly reduces the parameter number, which has only 35% parameters of the CNN based model and 3.4% parameters of TRMF.

2 RELATED WORK

2.1 Time Series Forecasting

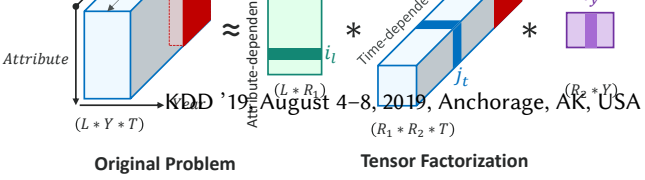
Traditional Time Series Prediction. Many efforts have been devoted to predict time series [4]. The ARIMA (Auto Regressive Integrated Moving Average) [3] model is one of the most popular and flexible linear models for stationary processes. TBATS with regressors method [11] has been proved to be effective in forecasting time series with multiple seasonal periods. A family of OMKR algorithms [15] were proposed for kernel based regression to capture non-linear patterns in an online and scalable way. Those traditional models are not good at capturing both long-term (e.g., yearly trend) and short-term properties, or handling noisy data with a large portion of mistakes or missing values. Most importantly, they model temporal changes of each time series separately, as

a result it would take them expensive computatio
process large-scale time series.

Matrix Factorization based Prediction. Furt
matrix factorization models [1, 2]have been proposec
time series problems for recommendation and spati
ral modeling. Matsubara et al. proposed CompCub
capture some time series patterns(e.g., non-linear dynamics, seasonalities) and then Takahashi et al. proposed AutoCyclone [17] using matrix factorization for better performance. Based on tensor factorization and multilinear dynamic systems, Facets[5] addressed three prominent challenges in mining a network of high-order time series data with contextual constraints and temporal smoothness. Recently, Yu et al. [21] designed a temporal regularized matrix factorization (TRMF) framework for high-dimensional, noisy data forecasting. Most of those models, however, focus on temporal properties but neglect to characterize correlations among attributes. Besides, their prediction results mainly rely on optimizing the factorization, neglecting utilizing any pre-knowledge or pre-processing of the input data. Facing a large number of correlated attribute combinations and years of historical data in different business scenarios, it is hard for them to further improve prediction accuracy or reduce the time-space complexity.

Deep Learning based Time Series Prediction Models Recently, with the remarkable development of deep learning techniques, a number of deep learning based models for time series prediction have been developed. There are two main streams of models, one is based on RNN and the other is based on CNN. Many efforts have been devoted to time series prediction based on RNN or its variants [6, 24],e.g., ST-RNN[12], SFM recurrent networkcite[24] and DA-RNN [14]. Some other researchers pay attention to extend CNN or even combine RNN and CNN to achieve further improvement, e.g., DeepST[20] and ST-ResNet [22]. Compared with traditional time series prediction, those deep learning based models achieve accuracy improvement. However, most of RNN based models focus on characterizing temporal features but neglect correlations among attribute combinations. Some CNN based models consider spatial dependencies, e.g., the CTR prediction model [9], but neglect long-term temporal features. Moreover, most of them are designed for specific applications (e.g., traffic flow prediction [22, 23]) and make assumptions on the attribute dimension and spatial correlations, which results in poor adaptability for different scales and spatial correlations of diverse business scenarios. For example, ST-ResNet [22, 23] is only suitable for two-dimensional spatial data (e.g., the GPS data).

As a summary, there is still a lack of an efficient highly scalable and adaptable framework for high-dimensional time series forecasting, which should be capable of characterizing both spatial and temporal features and handling both long-term and short-term prediction as well as missing values.



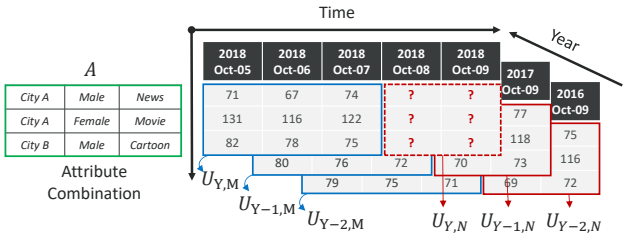


Figure 1: Example of user visits records and prediction problem.

3 UNDERSTANDING LARGE-SCALE USER VISITS

3.1 Problem Definition

In this work, we focus on high-dimensional time series forecasting, specifically, user visits forecasting for online guaranteed delivery advertising. As the example shown in Fig.1, there are years of daily records of user visits for each attribute combination (e.g., "City A", "Male" and "News"). The cornerstone is forecasting the number of user visits of each attribute combination in the next few days (e.g., the numbers represented by question marks in Fig.1) accurately based on the historical data. Inaccurate prediction will cause under-deliver or over-deliver, which both result in economic loss. Even one percent accuracy improvement matters for the revenue. The highly changing advertising scenarios (e.g., different applications and advertising space categories) require the prediction model to efficiently adapt to different data volumes, attributes and time spans.

3.2 Insights From Real Data Analysis

First, we analyze the anonymized user visits data of Tencent. The dataset includes user visits for diverse advertising spaces in different applications, such as news, music, video and sports, from February 2014 to August 2018. Each day, there were billion level user visits in total. The attributes of each visit include platform, area, content, gender, age group and so on. Based on the extensive analysis, we have the following insights which provide a deeper understanding of user visits as well as inspire the design of forecasting framework.

•**Insight #1: High-dimensional attribute combinations and spatial distribution of user visits**: The number of attribute combinations is very large. Even only consider the most commonly used three attributes (platform, area, content), there are about 40 thousand attribute combinations. Nevertheless, a large portion of attribute combinations have few user visits. When considering three attributes, there are 21,483 valid combinations in our dataset, which have at least one user visit, and 18,270 combinations are illegal with zero user visit. We calculated the whole user visit number for each combination. As presented in Fig.2, the summation of user visits of top 5,000 (20% of 21,483) attribute combinations accounted for nearly 95% of total user visits. The spatial distribution of user visits follows a power-law distribution,

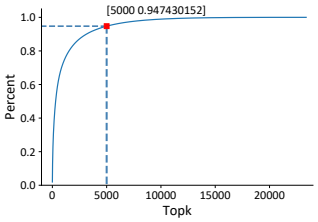


Figure 2: Spatial distribution of user visits.

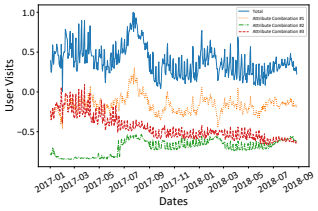
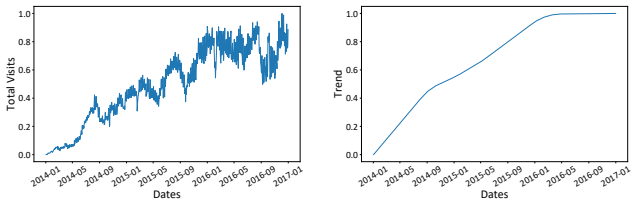
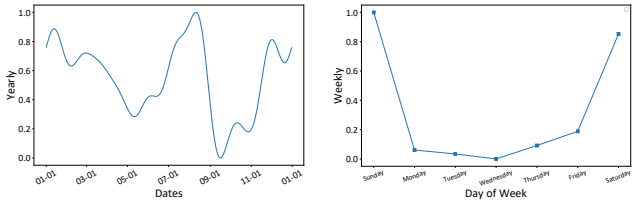


Figure 3: User visits of different attribute combinations.

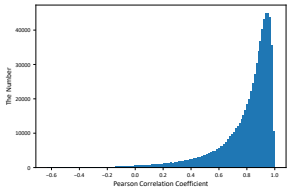


(a) Long-term total user visits. (b) Long-term trend of user visits.

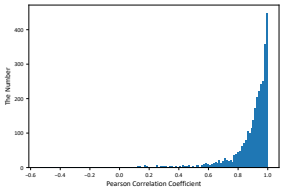


(c) Yearly property of user visits. (d) Weekly property of user visits.

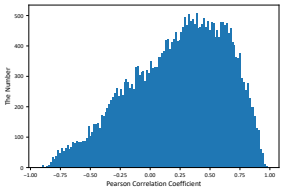
Figure 4: Temporal properties of large-scale user visits. All values are normalized to $[0, 1]$.



(a) Histogram of Pearson correlation of user visit time series for any combinations of platform and content type when fixing area value.



(b) Histogram of Pearson correlation of user visit time series for any combinations of area and content type when fixing platform value.



(c) Histogram of Pearson correlation of user visit time series for any combinations of platform and area when fixing content value.

Figure 5: Pearson correlation of time series with different attribute combinations, considering three attributes of user visits, platform, area and content type.

which is also known as the Pareto distribution or "80-20 rule". Moreover, we notice that there could be some missing/incorrect values in the dataset, which may be caused by errors during data collecting and processing. Thus, a highly scalable and robust forecasting model is necessary but challenging.

•**Insight #2: Diversity and Correlation among attribute combinations**: User visits across different attribute combinations are highly diverse. For example, patterns of user visits of big cities and small cities could be quite different. Fig.3 illustrates user visits of three different attribute combinations. Further, we explore the correlation among user visits with different attribute combinations. As shown in Fig.5, there is a strong correlation between attributes platform and content (Fig.5(a)), as well as area and content (Fig.5(b)). Attributes platform and area are also correlated to some extent (Fig.5(c)). Those correlations among attributes are neglected by existing work. Failing to capture correlations, however, could cause accuracy loss for user visits forecasting.

•**Insight #3: Temporal properties in an additive perspective**:

Fig.4(a) plots the total user visits from February 2014 to February 2017. In an additive perspective, we understand the temporal properties of total user visits by exploring three major factors: trend, period and temporal closeness factor. Fig.4(b) presents the long term growth trend, which is similar to the population growth in natural ecosystems [10]. For the period, via the frequency-domain analysis we found two notable periods, i.e., year and week. For a year (Fig.4(c)), in general, there are two peaks around the summer vacation and winter vacation, and the trough is reached after the summer vacation. For a week (Fig.4(d)), the number of user visits reaches the peak on the weekend and drops sharply on Monday. It decreases slowly before Wednesday and gradually recovers after Wednesday. Besides, user visits are also affected by latest events, e.g., the national holiday. These observation inspire us that, to improve prediction accuracy, we should not only use temporally close data but also data from the same period in the past few weeks and years, which help to capture long-term trend and weekly/yearly periodic properties.

4 DEEP SPATIAL-TEMPORAL FACTORIZATION FRAMEWORK

According to aforementioned insights, we propose a tensor factorization framework to forecast user visits considering both spatial and temporal features, as well as handle the high dimension and missing value issues efficiently.

4.1 Design Overview

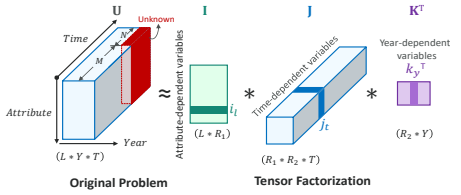


Figure 6: Concept of 3rd-order tensor factorization. The red part of the 3-rd order tensor U is unknown.

Fig.7 illustrates our proposed forecasting framework. Instead of treating user visits of each attribute combination as a separate time series, our basic idea is to employ tensor factorization to formalize the forecasting problem, which is efficient and highly scalable to handel high-dimensional time series. Both spatial latent vector and temporal latent vector are extracted by well-designed networks and input to the tensor factorization component to predict the results.

Specifically, let $U \in \mathbb{R}^{L*Y*T}$ be the 3-rd order tensor with three modes including attribute combination, time and year (Fig.1). Here L is the number of attribute combinations. For each attribute combination, the time span of its time series is T days. Y means there are Y year data in total. As presented in Fig.6, based on the equivalent representation of the PVD (Population Value Decomposition) model, the tensor U can be decomposed to the factor matrices $I \in \mathbb{R}^{L*R_1}$, $K \in \mathbb{R}^{Y*R_2}$ and a core tensor $J \in \mathbb{R}^{R_1*R_2*T}$. Intuitively, I, J and K represent attribute-dependent variables, time-dependent variables and year-dependent variables respectively. Let u_{lyt} be an element of U, which indicates the count of user visits of the l-th attribute combination at the t-th day in the y-th year. u_{lyt} can be estimated by the inner product $i_l * j_t * k_y^T$. Here, i_l is a *spatial latent embedding* for the l-th attribute combination, which is the l-th row of I. j_t is a *temporal latent embedding* matrix for the t-th day, which is the t-th slice of J. k_y is a temporal latent embedding for the y-th year, which is the y-th row of K. Then, to forecast the unknown u_{lyt} (i.e., the red part of the tensor U), we should explore the optimal factorization by minimizing the estimation error:

$$\min_{I,J,K} \sum_{u_{lyt} \in \Omega} (u_{lyt} - i_l * j_t * k_y^T)^2 + \alpha * R_1(I) + \beta * R_2(J) + \gamma * R_3(K).$$

Here Ω is the set of known elements of U, i.e., the light blue part of U. $R_1(I)$, $R_2(J)$ and $R_3(K)$ are regularizers. Specially, comparing to existing matrix factorization machine based methods like TRMF [21], our model adds an extra mode "Year" considering data of the same period in the past few years. It not only provides a more general framework, but also becomes more capable of capturing the long-term periodic properties of user visits.

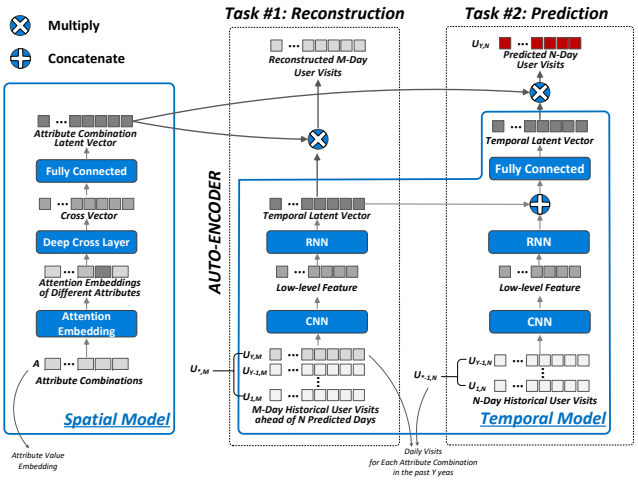


Figure 7: Deep spatial-temporal tensor factorization forecasting framework.

Taking the above tensor factorization as the main structure, our forecasting model includes the rest following components.

•**Spatial model**: in this component, there are graph attention and deep cross layers used to characterize implicit attribute interactions, followed by a fully connected layer to generate the attribute combination latent vector.

•**Temporal model**, in this component, we combine CNN, RNN and DNN to capture information about the input time series at different scales and produce the temporal latent vector. Considering the temporal closeness factor as well as yearly trend and periodic properties of user visits, we input not only data of several weeks before the days to be predicted, but also data of the same period in the past few years.

•**Multi-task**: in our framework, we design multiple tasks for the training process, i.e., multiple optimization objectives. Specifically, based on tensor factorization and spatial-temporal feature modelling, we develop one auto-encoder structure to reconstruct input data and one prediction structure to predict future user visits. Multiple tasks share some common layers to reduce parameter number as well as jointly achieve better prediction results.

These components work jointly to achieve better spatial-temporal feature representation, more accurate time series forecasting, as well as high scalability and efficiency. We present detailed design in the following subsections.

4.2 Spatial Feature Modeling

This model captures implicit correlations among different values of the same attribute and different attributes. Given attribute combinations as the input, we first employ an attention embedding [18] module to capture correlation among different value embeddings of the same attribute. More importantly, the attention mechanism allows focusing on the most relevant parts of the input to make decisions. In our framework, the attention mechanism is a single-layer feed-forward neural network, and for each attribute, its attention

embedding is the weighted sum of all attribute value embeddings. Obtaining the attention embedding of each attribute, we concatenate them into a whole vector and feed it into a DeepCross [19] layer to learn the bounded-degree interactions among different attributes. By crossing different attributes, it generates the cross vector which is converted to the attribute combination latent vector via a fully connected layer.

4.3 Temporal Feature Modeling

Inspired by CLDNN [16], we combine CNN, RNN and DNN layers into one unified network and train them jointly to extract temporal features of historical data. The CNN block takes raw historical daily user visits for each attribute combination as input and reduces frequency variance in the input. Then, the CNN block feds the output low-level feature into the RNN block to reduce temporal variation. The RNN block models the time series temporally by converting the low-level feature to temporal latent vector. The output of RNN is fed into a fully connected DNN block, which maps features to a more separable space. CNN, RNN and DNN blocks are complementary in modeling capability and work as a whole to capture temporal properties at different scales. In details, the input data are divided into two parts. As shown in Fig.6, the light blue part is the input historical user visits, while the red part is unknown and to be predicted. Let the set of N-day unknown user visits be $U_{Y,N} = \{u_{lyt} | 1 \leq l \leq L, y = Y, M < t \leq T\}$, here $M + N = T$. The first part of input is $U_{*,M} = \{u_{lyt} | 1 \leq l \leq L, 1 \leq y \leq Y, 1 \leq t \leq M\}$, which is the set of M-day time series before the predicted N-day period in all Y years. This part is used to capture the temporal closeness factor and periodic property. The second part is $U_{*-1,N} = \{u_{lyt} | 1 \leq l \leq L, 1 \leq y \leq Y-1, M < t \leq T\}$, which is the data from the same N-day period of $U_{Y,N}$ in the past $Y-1$ years. This part is used to capture the long-term trend and periodic properties. Fig.7 presents the network design. Both parts of input are processed first by the CNN block and then by the RNN block. Their outputs are concatenated and fed into the DNN block to produce the temporal latent vector for the tensor factorization based forecasting. Note that, when $Y = 1$, i.e., the time span of historical data is shorter than one year, the second part of the input can be omitted. Specially, to minimize the information loss caused by encoding the input data into a single latent vector, we leverage an auto-encoder structure to reconstruct the input $U_{*,M}$ with minimal loss.

4.4 Multi-task

We propose a multi-task design as shown in Fig.7. There are two tasks: reconstructing historical user visits data using an auto-encoder structure and predicting future user visits via tensor factorization model. The auto-encoder structure minimizes reconstruction loss to retain as much information as possible during the encoding process. These shared layers, including CNN, RNN, attention embedding, cross and fully connected layers, are used by both tasks. During the training process, we measure the combined loss of both auto-encoder and prediction by $Loss_{total} = \eta Loss_{AE} + (1-\eta) Loss_{Pred}$, where η is the weight parameter to balance two tasks. Usually, we set $\eta < 0.5$ since prediction is our main objective. This multi-task design achieves the following benefits: (1) minimizing information loss during spatial and temporal latent vector embedding; (2) sharing layers as well as parameters among models to reduce parameter number and cost; (3) improving prediction accuracy through multi-objective optimization. Our evaluation results prove the effectiveness of our design.

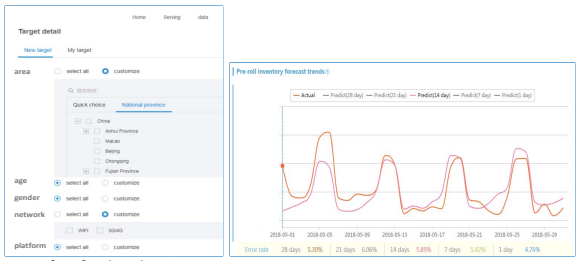


Figure 8: Real system interfaces for advertiser and publisher.

5 EXPERIMENTAL RESULTS

The proposed framework has been deployed in the Tencent online guaranteed delivery advertising system for more than 5 months. It has brought a considerable revenue growth thanks to more accurate user visits prediction results. Fig.8 shows the real system interfaces for advertiser to specify target users' attribute and for publisher to view the forecasted user visits. Hereafter, we use **ST-TF** to denote our framework for simplicity. In this section, extensive evaluations are conducted to verify the effectiveness and efficiency of ST-FT.

5.1 Experiment Setup

We evaluate our framework using large-scale real user visit data. To compare with related exiting work, meanwhile, to show the scalability and adaptability of our framework, we also conduct evaluation using the PEMS-SF traffic dataset. The details of two datasets are as follows.

Dataset #1: Tencent online video site dataset (referred to as "Video Dataset") contains anonymized daily user visits [1] to the video site's pre-roll advertisements with three attributes, namely, platform, area and type of content. There are 3 platforms, 341 areas, 21 types of content and billions of user visits per day. In total, there are 21,483 valid attribute combinations having at least one user visit, and 11,124 combinations having at least 1000 user visits. We retain data of the 11,124 combinations and filter data of other combinations. Because, for online guaranteed delivery advertising, when the publisher sells user visits to advertiser, the smallest unit is 1000 user visits. The time span of the training data is from January 1, 2016 to August 1, 2017 and that of the testing data is from September 1, 2017 to August 1, 2018. There is no overlap between the training data and testing data.

[1] There is no personally identifiable information in this dataset.

Dataset #2: PEMS-SF dataset[8] (referred to as 'Traffic Dataset') gives the occupancy rate (from 0 to 1) of different car lanes of San Francisco bay area freeways from January 1, 2008 to March 30, 2009. There are 963 time series, each of which was generated by a station/detector. The time span is 440 days in total. For the experiment, we use the data of the first 365 days as the training data, and the data of the rest 75 days as the testing data. For the spatial latent vector, since there is no attribute describing each time series in this dataset, we simply embed IDs of time series. Note that, there is only one year training data, i.e., $Y = 1$, so the second part of the input $U_{*-1,N}$ is omitted.

Training and testing samples. For both datasets, as described in Section 4.3, let M (days) be the time span of the training data and N (days) mean the prediction is N days in advance. For example, to generate a training sample for the Video Dataset, we first select a random date d during January 1, 2017 to July 1, 2017 as "today" and a random attribute combination. Then, we use the time series between $[d-k*365-M+1, d-k*365+N]$, here $k \in 0, 1$, as a training sample. Testing samples are generated from the testing data in the same way. In the following experiments, we set $M \in \{7, 28, 56, 112\}$ due to the weekly periodic property of user visits, and set $1 \leq N \leq 28$ according to real business requirements.

Accuracy evaluation metrics. Let Ω_{test} be the testing set. For each value $u_{lyt} \in \Omega_{test}$, the predicted value is $\hat{u}_{lyt}$. To measure the accuracy of prediction results, we use normalized deviation (ND) and normalized root mean square error (NRMSE) as metrics, which are respectively defined as

$$\frac{\left(\frac{1}{|\Omega_{test}|}\sum_{u_{lyt}\in\Omega_{test}}|\hat{u}_{lyt}-u_{lyt}|\right)}{\left(\frac{1}{|\Omega_{test}|}\sum_{u_{lyt}\in\Omega_{test}}|u_{lyt}|\right)}, \frac{\sqrt{\frac{1}{|\Omega_{test}|}\sum_{u_{lyt}\in\Omega_{test}}(\hat{u}_{lyt}-u_{lyt})^2}}{\left(\frac{1}{|\Omega_{test}|}\sum_{u_{lyt}\in\Omega_{test}}|u_{lyt}|\right)}.$$

Models for comparison. We compare accuracy and time-space efficiency of our framework with the following state-of-the-art time series forecasting models:

•**Mean**: we take the "Mean" method as a baseline. For each attribute combination, it averages the user visits of the past M days as the future prediction.

•**ARIMA[3] and TBATS with regressors [11]**: they are popular traditional time series prediction methods, which model time series of each attribute combination separately.

•**LSTM**: it is a typical LSTM network. In our experiment, we set the hidden units size to 32.

•**CNN**: inspired by the well-known Google WaveNet [7], we use convolutional nets with size [3 * 1] to capture single dimension time series feature and then using two-layer convolutional nets to build the residual unit. In total, we stack 3 residual units. The final predicted data are generated by one dense layer with size 28.

•**TRMF [21]**: it is a popular matrix factorization based framework,which can be considered as a special case of our ST-TF framework. In the TRMF model, we set the hidden units size as 64 and $L = \{1, 2, \cdots, M\}$ to capture the correlation with the previous M days which is consistent with our design.

	Video Dataset (ND/NRMSE)	Traffic Dataset (ND/NRMSE)
ST-TF	**0.179 / 1.093**	**0.157 / 0.284**
TRMF	0.19 / 1.122	0.17 / 0.287
CNN	0.196 / 1.12	0.174 / 0.291
ARIMA	0.212 / 1.191	0.235 / 0.379
TBATS	0.208 / 1.294	0.268 / 0.418
LSTM	0.238 / 1.279	0.201 / 0.323
Mean	0.251 / 1.321	0.241 / 0.363

Table 1: 28-day average forecasting error for different models on two datasets.

Parameter settings of ST-TF. For the rest experiments, we set the parameters of our framework ST-TF as follows. The batch size is 2048 and epoch num is 5. Embedding size is $6 * (category\ cardinality)^{1/4}$ according to [19]. To extract lower feature as well as keep the network structure simple, we set depth of the cross layer to 2, depth of CNN module to 1 and kernel size to [3,1]. We use bi-direction RNN and the RNN hidden size is 64. Dense layer size for spatial model is 64 and for prediction model we set it to 28*64 which can generate the future 28-day feature according to the business requirement. All activation functions are RELU. For the multi-task, we set $\eta = 0.2$, which means we focus more on the prediction model which is the final goal of our work.

5.2 Forecasting Accuracy

5.2.1 Prediction of user visits N days in advance. When predicting user visits N days in advance, N usually varies for different business requirements. Here, let $N = \{1, 2, \cdots, 28\}$ and the training window size $M = 56$, which is the value adopted by our real system.

28-day average accuracy. Fig. 9 and Fig. 10 compare the average ND and NRMSE values of ST-TF with that of other existing methods. Table.1 presents the detailed ND and NRMSE values. On the Video Dataset, all six methods perform better than the baseline method "Mean". Among the six methods, the traditional time series prediction methods (ARIMA and TBATS) and classic LSTM perform worst. The reason is that they focus on characterizing the temporal trends of separated time series but fail to capture the correlation among them. The latest CNN-based deep learning method (CNN) and factorization based method (TRMF) achieve better performance than those traditional methods. Our framework (ST-TF) outperforms all existing methods and achieves average 0.179 ND and 1.093 NRMSE. ST-TF provides a 8.7% relative improvement over CNN, and a 5.8% relative improvement over TRMF. The improvements are achieved by a better spatial correlation and temporal properties characterizing and the multi-task design. On the Traffic Dataset, similarly, ST-TF, TRMF and CNN outperform other methods significantly. In this case, ST-TF also achieves the best accuracy and the accuracy is much better than that on the Video Dataset. ST-TF provides a 9.8% relative improvement over CNN, and a 7.6% relatively improvement over TRMF. The reason is that, the attribute number of the Traffic Dataset is much smaller than that of the Video Dataset. Note that, since the Traffic Dataset has no attribute information (only the IDs of time series), the improvements are mainly contributed by our well-designed temporal model.

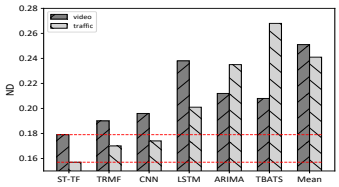


Figure 9: 28-day average ND on both datasets.

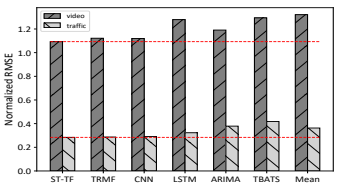


Figure 10: 28-day average N-RMSE on both dataset.

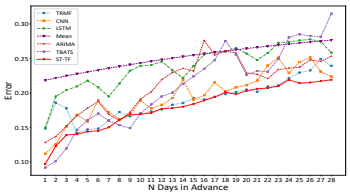


Figure 11: ND against different N on the Video Dataset.

Accuracy against different N. Fig. 11 further illustrates the forecasting error with different N values. Overall, for all methods, the error increases as N gets larger, and our ST-TF method achieves the smallest error. Thus, ST-TF is able to provide accurate short-term and long-term high-dimensional time series forecasting. Specially, the error of ST-TF increases much more smoothly than that of other state-of-the-art methods. It implies that, ST-TF performs more stably in different business scenarios.

Accuracy against different date. Fig.12 plots the detailed deviations for different predicted dates in a whole year. The deviation varies in different periods of a year. When N is larger, the variance is greater. The greatest deviations occur around September and February. The reason is that the number of user visits usually has a significant change around the summer vacation and winter vacation, as shown in Fig.4(c). However, in China the time of winter vacation varies from year to year. For such case, it is difficult to achieve accurate prediction by directly using data from past years. Fortunately, after a suddenly increase, the deviation drops quickly within a few days. In our real system, to further mitigate the deviation caused by time-shifted events, we shift the data from past years before feeding it to the network.

Accuracy for small cities. We notice that, 20% attribute combinations account for nearly 95% of total user visits, and for many combination with small user visit numbers, the variance is usually larger. Lager variance incurs more difficulty to prediction. Table 2 presents the average forecasting error for 10 small cities, each of which has less than 50,000 daily user visits. ST-TF achieves 0.185 ND, which is slightly worse than 0.179 ND for all combinations. The errors of other methods also increase to different extent. But ST-TF still outperforms other methods, providing a 18.8% relative improvement over CNN and a 6% relatively improvement over TRMF.

Method	ST-TF	TRMF	CNN	LSTM	ARIMA	TBATS
ND	0.185	0.197	0.228	0.261	0.229	0.221

Table 2: 28-day average forecasting error for 10 small cities (each has less than 50,000 daily user visits).

5.2.2 Different training data window size M. Fig. 13 presents accuracy changes against different training data window size M. Generally, for Mean and LSTM, larger training data window size leads to worse accuracy, while for other methods, different window size doesn't change accuracy significantly. Overall, ST-TF achieves best accuracy only except when $M = 7$. When $M = 56$, our framework achieves best performance, where the 28-day average ND is 0.179. Here 56 is a magic number because an 8-week (nearly two months) data provides just the proper amount of information for characterizing recent weekly and monthly trends. When window size increases to 112, the performance drops slightly due to the fact that a larger time span is more complicated to characterize, and more noises may be introduced.

Component	Situation	ND	NRMSE
Complete Framework	-	**0.179**	**1.093**
Input Data	No Last Year Data	0.181	1.099
	No Attribute Embedding	0.185	1.139
Temporal Model	Simple NN	0.184	1.129
	No RNN	0.187	1.158
	No CNN	0.183	1.114
Spatial Model	No Cross Layer	0.182	1.104
	No Attention	0.184	1.118
Framework	No Multi-task	0.182	1.097

Table 3: ND of our framework after removing each single component separately.

5.2.3 Effectiveness of each component. Here, we use control experiments to explore the contribution of each component of our framework. Table 3 presents the framework accuracy after removing each single component separately. The accuracy drop (i.e., the ND increase) compared with the complete framework reveals the effectiveness of each component. Every component contributes to the framework and increases the performance to different extent.

5.2.4 Accuracy with missing data. To evaluate the performance when there are missing data, during the training phase, we drop 20% input data randomly. Averagely, there is only a slight accuracy loss (the ND decreases from 0.179 to 0.18), which shows that our framework can resist to even 20% missing data in practice.

5.3 Complexity and scalability.

Now, we evaluate the space and time cost for different methods. Traditional methods including ARIMA and TBATS are very time consuming, since they model each series separately. They takes about 1 hour to process one day data. Table 4 presents the parameter number of LSTM, CNN, TRMF and our model, and training time on the Video Dataset. The training time of four models are comparable, however our model has only about 35% parameters of CNN and 3.4% parameters of TRMF. Although, LSTM has the lest parameters due to its simple structure, its accuracy is much worse than other three models as presented in Table1. The results show that our sharing weights and multi-task design greatly reduces the parameter number, thus significantly saves space.

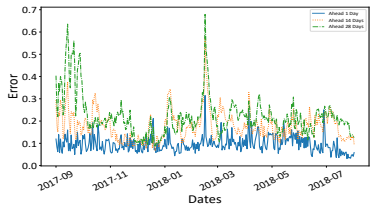


Figure 12: The detailed ND distribution with different N.

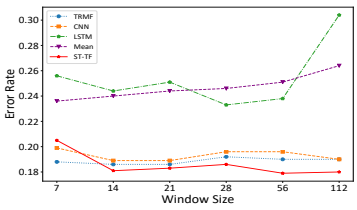


Figure 13: 28-day average ND with different window M.

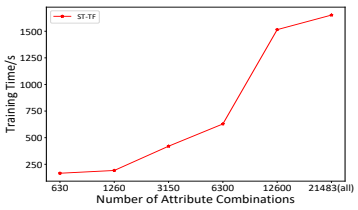


Figure 14: Training time with attribute combination sizes.

Method	# of Parameters	Training Time (s)
ST-TF	61,850	1,652
LSTM	18,716	1,784
CNN	174,748	1,248
TRMF	1,800,879	1,654

Table 4: Parameter number and training time of different networks ($W = 56$).

Fig. 14 shows that the training time of ST-TF increases linearly with the number of combinations. When using two-month data, which contains billions of user visits per day and 21,483 attribute combinations, it takes ST-TF about 27 minutes to train a high-accuracy model. Besides, our model achieves good performance for both user visits forecasting and traffic prediction tasks. The results show the ability of ST-TF to handle large-scale time series prediction, and the adaptability for diverse applications with different attributes.

6 CONCLUSION

In this work, we propose a novel deep spatial-temporal tensor factorization based framework to achieve accurate high-dimensional time series forecasting. We have deployed the framework in Tencent advertising system and extensively evaluated our design using two large real-world datasets, the accuracy of our framework outperforms all existing methods and the space complexity is significantly reduced. The results show that our design is highly scalable, adaptable and efficient, which can be adopted by various applications, such as online advertising, traffic management and public safety.

ACKNOWLEDGMENTS

This work is supported by the National Key R&D Program of China 2017YFB1003003, NSF China under Grants No. 61822209, 61751211, 61572281, 61520106007, and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Mohammad Taha Bahadori, Qi Rose Yu, and Yan Liu. 2014. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Advances in neural information processing systems*. 3491–3499.

[2] Preeti Bhargava, Thomas Phan, Jiayu Zhou, and Juhan Lee. [n.d.]. Who, what, when, and where: Multi-dimensional collaborative recommendations using tensor factorization on sparse user-generated data. In *Proceedings of WWW*.

[3] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

[4] Peter J Brockwell and Richard A Davis. 2016. *Introduction to time series and forecasting*. Springer.

[5] Yongjie Cai, Hanghang Tong, Wei Fan, Ping Ji, and Qing He. 2015. Facets: Fast comprehensive mining of coevolving high-order time series. In *Proceedings of SIGKDD*. ACM, 79–88.

[6] Sakyasingha Dasgupta and Takayuki Osogami. 2017. Nonlinear Dynamic Boltzmann Machines for Time-Series Prediction.. In *AAAI*. 1833–1839.

[7] Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu. 2016. WaveNet: A Generative Model for Raw Audio. *arXiv: Sound* (2016), 125.

[8] Dua Dheeru and Efi Karra Taniskidou. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[9] Hongchang Gao, Deguang Kong, Miao Lu, Xiao Bai, and Jian Yang. 2018. Attention Convolutional Neural Network for Advertiser-level Click-through Rate Forecasting. In *Proceedings of WWW*. ACM, 1855–1864.

[10] George Evelyn Hutchinson. 1978. An introduction to population ecology. (1978).

[11] Rob J Hyndman. 2014. TBATS with regressors. Retrieved September 2017 from https://robjhyndman.com/hyndsight/tbats-with-regressors/

[12] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the Next Location: A Recurrent Model with Spatial and Temporal Contexts.. In *AAAI*. 194–200.

[13] Yasuko Matsubara, Yasushi Sakurai, and Christos Faloutsos. 2016. Non-linear mining of competing local activities. In *Proceedings of WWW*. ACM, 737–747.

[14] Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison Cottrell. 2017. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv:1704.02971* (2017). https://arxiv.org/abs/1704.02971

[15] Doyen Sahoo, Steven CH Hoi, and Bin Li. 2014. Online multiple kernel regression. In *Proceedings of SIGKDD*. ACM, 293–302.

[16] T. N Sainath, O Vinyals, A Senior, and H Sak. 2015. Convolutional, Long Short-Term Memory, fully connected Deep Neural Networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. 4580–4584.

[17] Tsubasa Takahashi, Bryan Hooi, and Christos Faloutsos. 2017. Autocyclone: automatic mining of cyclic online activities with robust tensor factorization. In *Proceedings of WWW*. ACM, 213–221.

[18] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv:1710.10903* 1, 2 (2017). https://arxiv.org/abs/1710.10903

[19] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. 2017. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*. ACM, 12.

[20] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*. 802–810.

[21] Hsiang-Fu Yu, Nikhil Rao, and Inderjit S Dhillon. 2016. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*. 847–855.

[22] Junbo Zhang, Yu Zheng, and Dekang Qi. 2017. Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction. In *AAAI*. 1655–1661.

[23] Junbo Zhang, Yu Zheng, Dekang Qi, Ruiyuan Li, and Xiuwen Yi. 2016. DNN-based prediction model for spatio-temporal data. In *Proceedings of the 24th ACM SIGSPATIAL*. ACM, 92.

[24] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. 2017. Stock Price Prediction via Discovering Multi-Frequency Trading Patterns. In *Proceedings of SIGKDD*. ACM, 2141–2149.