Knowledge and Information Systems (2020) 62:2139–2167
https://doi.org/10.1007/s10115-019-01404-8

REGULAR PAPER

Online sales prediction via trend alignment-based multitask recurrent neural networks

Tong Chen[1] · Hongzhi Yin[1] · Hongxu Chen[1] · Hao Wang[2] · Xiaofang Zhou[1] · Xue Li[1,3]

Received: 19 March 2019 / Revised: 11 September 2019 / Accepted: 13 September 2019 /
Published online: 9 October 2019

Abstract
While business trends are constantly evolving, the timely prediction of sales volume offers precious information for companies to achieve a healthy balance between supply and demand. In practice, sales prediction is formulated as a time series prediction problem which aims to predict the future sales volume for different products with the observation of various ***influential factors*** (e.g. brand, season, discount, etc.) and corresponding historical sales records. To perform accurate sales prediction under the offline setting, we gain insights from the encoder–decoder recurrent neural network (RNN) structure and have proposed a novel framework named TADA (Chen et al., in: ICDM, 2018) to carry out trend alignment with dual-attention, multitask RNNs for sales prediction. However, the sales data accumulates at a fast rate and is updated on a regular basis, rendering it difficult for the trained model to maintain the prediction accuracy with new data. In this light, we further extend the model into TADA$^+$, which is enhanced by an online learning module based on our innovative similarity-based reservoir. To construct the data reservoir for model retraining, different from most existing random sampling-based reservoir, our similarity-based reservoir selects data samples that are "hard" for the model to mine apparent dynamic patterns. The experimental results on two real-world datasets comprehensively show the superiority of TADA and TADA$^+$ in both online and offline sales prediction tasks against other state-of-the-art competitors.

Keywords Online sales prediction · Recurrent neural networks · Attention mechanism · Time series analysis

1 Introduction

For various retail businesses, keeping a balance between supply and demand is crucial to retailers, and the accurate prediction of sales volume is becoming indispensable for commercial success [6]. Overestimated sales can result in excessive inventory, unhealthy cash flow and even bankruptcy, while the underestimated sales may lead to unfulfilled orders, decreased business reputation and profit [26]. In practice, sales prediction is formulated as a time series forecasting problem, which aims to predict future sales volume based on the observed mul-

Extended author information available on the last page of the article

tivariate time series data which consists of historical sales volume and ***influential factors*** (e.g. brand, season, discount, etc.). Thus, a reasonable modelling of the influential factors and historical sales information should be performed to successfully predict sales volume.

In recent years, time series prediction algorithms are widely adopted in many areas such as financial market prediction [37,49], recommender systems [11,54] and medical research [5,29]. Among these techniques, the discovery of trending events or repeating patterns based on the clues from historical observations has inspired some interesting applications like traffic modelling [28], solar intensity prediction [35] and argument discovery [23]. Undoubtedly, the discovery of recurring trends will greatly benefit the forecast of sales by aligning relative contextual information learned from the influential factors, and this insight is referred to as ***trend alignment*** in this paper. However, both traditional autoregressive-based methods [4,19] and recent trend mining models [28,40] are ineffective for the trend alignment in sales prediction. This is because these methods assume that the trend in time series data recurs periodically (i.e. distributes with a fixed time period), thus requiring domain knowledge for every application area and carefully chosen parameters based on the data. Hence, existing techniques are unable to align similar trends in sales time series where the sales patterns are much more subtle and irregular due to the effect from complicated real-world situations, and the difficulty increases when there are a large number of different products.

The formation of a trend in sales time series has specific contexts which can be modelled from the interaction among various influential factors. In regard to contextual information learning from raw time series, recurrent neural network (RNN) models have been intensively studied and applied to learn vector representations from sequential inputs [17,28,42]. Compared with previous efforts on time series prediction like kernel methods and Gaussian process [24,48] which are limited by their predefined nonlinear form, RNNs show their advantages in flexible yet discriminative nonlinear relationship modelling. Moreover, two variants of RNN, namely long short-term memory (LSTM) [21] and gated recurrent unit (GRU) [13], further advance the performance in tasks related to neural machine translation [1] and image captioning [50]. Among these applications, the encoder–decoder RNN architecture leverages two independent RNNs to encode sequential inputs into latent ***contextual vectors*** and decode these contexts into desired interpretations [1,43,50]. After showing its superiority in recent time series modelling tasks [28,37], it is natural to consider encoder–decoder RNNs for sales prediction by leveraging its capability to fully capture the nonlinear relationship between the influential factors and the sales volume.

However, even with the state-of-the-art encoder–decoder RNN models, sales prediction is still a challenging research problem because when multiple influential factors interact with each other, they have different influences on different products. For instance, the temperature has more impact on the sales of down jackets than shirts because shirts are intrinsically cheaper and can be worn all year round. Furthermore, the influential factors are dynamic and unpredictable in many cases, so it is impractical to assume their future availability. For example, though the environmental policy significantly affects electrical car sales, and the fashion trend dominates the clothing industry, we have very limited prevision on these influential factors. To make things worse, when performing trend alignment using contexts learned from the past, the decoder cannot generate rich contexts with the unknown states of influential factors.

In real-life scenarios, sales data is always updated on a regular basis, e.g. on each business day or at the end of each month. The aforementioned time series prediction methods may show the promising results under the offline setting, but they lack adequate ability to adapt to the updated data when performing sales prediction under the online setting. Thus, it is crucial to extend the offline prediction scheme to an online sales prediction scheme. On the one hand,

the incoming data stream is helpful for the verification of the correctness of the predicted sales; on the other hand, the updated sales time series data enriches the data samples that can be used to better tune the original model in order to better model the most recent sales trends to ensure the prediction accuracy. In a broad range of existing online models that deals with streaming data [7,14,45,46], a widely adopted strategy for online model parameter update is the reservoir-based online learning. With the new incoming data, a data reservoir is utilized to store a small portion of previously used data samples, and the stored data samples are further leveraged to retrain the model along with the updated data. The motivation of keeping both new and used data samples to retrain the model is to adapt the model parameters to emerging dynamic patterns within the new data while retaining as much information of used data as possible. Unfortunately, existing reservoir-based update methods may fail to help the model memorize the important information contained in the previous training data. This is because the majority of the reservoir-based update methods use the uniform random sampling to select a subset of samples from the used data; hence, a lot of "easy" prediction tasks and outliers will be absorbed by the reservoir for the retraining process. As a result, a random sampling-based reservoir is incapable of selecting representative training samples that substantially contribute to the optimization of model parameters.

Hence, the main challenges in online sales prediction are summarized as follows. The first is how to ***fully capture the dynamic dependencies among multiple influential factors***. Second, without any prior knowledge of mutative variables in the future, how can we possibly ***glean wisdom from the past to compensate for the unpredictability of influential factors***. Third, as different sales trends recur irregularly due to complex real-world situations, it is necessary to ***align the upcoming trend with historical sales trends***, thus selectively gather relative contextual information for accurate prediction of sales volume. Forth, as the sales data is updated on a regular basis, we need to address the way to ***develop an effective online update approach that can adjust to the new data without forgetting useful past information***.

In light of the first three challenges, we have proposed a novel sales prediction framework, namely Trend Alignment with Dual-Attention Multitask Recurrent Neural Network for Sales Prediction (TADA), which is our offline sales prediction model published in [10]. TADA consists of two major components: the multitask LSTM encoder and the dual-attention LSTM decoder, which are illustrated in Fig. 1. Moreover, in this paper, we further extend TADA

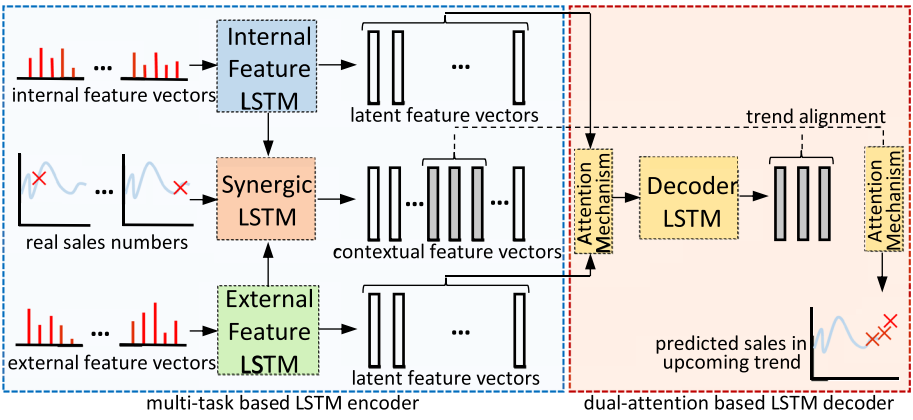


Fig. 1 Workflow of the offline sales prediction model TADA

with an online update mechanism, namely ***TADA***$^+$ to ensure the effectiveness under a more practical online sales prediction setting.

In order to solve the first challenge, we make our own observation on the characteristics of sales time series based on previous discussions. The semantics of influential factors in sales prediction are diverse, which, however, has been ignored by the conventional time series prediction methods. Specifically, for each product, its influential factors come with its intrinsic properties which are directly related to customers' subjective preference, e.g. brand, category, price, etc. Meanwhile, there are also many factors that objectively affects the sales, e.g. weather, holiday, promotion, etc. In this paper, we categorize the intrinsic properties of a product as its ***internal feature*** and the other influential factors as the ***external feature***. While internal features and external features express different semantic meanings, they both contribute to the fluctuations of the product sales volume at the same time. Hence, compared with predictive models that treat all kinds of features in a unified way [28,37,53], we propose a multitask LSTM encoder to learn contextual vector representations of historical sales time series. As shown in Fig. 1, to solve the first challenge, we novelly model the internal feature and external feature in parallel via two individual LSTM layers. Then, we use a synergic LSTM layer to simultaneously join these two learned latent representations at each time step. The insight of a multitask encoder structure is to comprehensively leverage all available resources by modelling internal and external features separately first and then pose a dynamic interaction between different features to generate contextual representations of historical sales time series.

To address the challenges of trend alignment and unknown influential factors, we propose an innovative dual-attention LSTM decoder to tackle the difficulties. Grasping intuitions from existing attention mechanisms [1,18,44] which aim to select relevant parts of hidden states learned by the encoder to attend, we develop our simple yet effective attention mechanisms which perfectly blend into the neural network for accurate sales prediction. As illustrated in Fig. 1, in the decoding stage, the first attention models the effect of unknown influential factors using relevant contextual vectors from the encoder. After new sales contexts are generated within the look-ahead time interval, the second attention gathers contextual information of this upcoming trend and then actively aligns the new trend with historical ones. Eventually, we combine the representation from the aligned trends to produce a sequence of estimated sales volume in the future.

Finally, to grant TADA the ability of online learning, in TADA$^+$, we present a novel similarity-based reservoir for online model update. Intuitively, when constructing a data reservoir using previous data samples, we aim to seek "hard" tasks and discard "easy" training samples based on the intermediate results. To select the most relevant contexts for the decoder as well as perform trend alignment, the dual-attention mechanisms in TADA utilize the similarity score (e.g. probability distribution) to pick the most important contextual information. To this end, we propose to fully leverage the similarity score within the attention mechanisms. In short, when computing different similarity scores with the attention mechanism, if the distribution of the similarity scores for different contexts is similar to the uniform distribution, it implies that the model is having difficulties in differentiating important contexts from irrelevant ones. Such cases are relatively challenging for the model because they do not show obvious dynamic patterns that can be mined by the existing model. So, in TADA$^+$, we design a similarity-based reservoir to select useful past data samples with a cut-off threshold for the distribution of the similarity scores and then combine the selected past data samples with the new ones to update the model and enable effective online sales prediction.

We summarize the primary contributions of our research as follows:

- We are the first to categorize the influential factors in sales time series into internal features and external features and innovatively model these two aspects with the multitask LSTM encoder. We also adopt a synergic LSTM layer to model the dynamic interaction between different types of influential factors.
- To obtain optimal sales prediction performance under the offline setting, we present the dual-attention multitask recurrent neural network to tackle the aforementioned challenges in sales prediction. The novel encoder–decoder structure can comprehensively model variables with different semantic meanings, and the dual-attention increases both the interpretability and accuracy of the model by simulating unknown states of future contexts and aligning the upcoming sales trend with the most relevant one from the past.
- To achieve better real-life practicality under the online prediction setting, we extend our conference version TADA [10] to TADA$^+$ with an online update scheme. The online update module uses a similarity-based reservoir to store new data as well as keep the most important training samples from the used training data, which ensures the robustness of the prediction results under an online setting.
- We conduct extensive experiments on two real-life commercial datasets. The results showcase the superiority of our approach in sales prediction by outperforming a group of state-of-the-art predictive models. We validate the vigorous contribution of each component in TADA via ablation tests and visualizations. Additional experiments on training efficiency further show promising scalability of TADA.

The rest of this paper is organized as follows. Section 3 formulates the sales prediction task and explains our proposed TADA in detail. We outline the related research backgrounds in Sect. 2. Section 4 verifies that the asymptotic time complexity of TADA is linearly associated with the scale of the data. After reporting the experimental results of our model in comparison with state-of-the-art baselines in Sect. 5, we conclude our findings with Sect. 6.

2 Related work

With our motivation stated, we review relevant literatures in order to clearly position our proposed method against different existing approaches. Specifically, our work is related to time series prediction, trend modelling and multitask learning.

2.1 Time series prediction

When performing sales prediction using multivariate time series, the techniques can be divided into linear models and nonlinear models. While linear models like autoregressive integrated moving average (ARIMA) [4], support vector machine (SVM) [38] and robust regression [39] mostly aim at finding parameterized functions from statistics, and nonlinear models like Gaussian process [48,51] and gradient boosting machines [9,16] can better model complicated dependencies by leveraging machine learning techniques. However, due to the high computational cost and unsatisfying scalability in real applications [28,58], these approaches are not ideal for sales time series which usually carries high dimensionality and long time range. In addition, these methods mainly rely on carefully designed mapping functions, so sufficient domain knowledge of the data is a prerequisite. To address this issue, recurrent neural network (RNN) [30], along with its two popular variants, namely long

short-term memory (LSTM) [21] and gated recurrent unit (GRU) [13], have been proposed to dynamically capture long-range dependencies among the sequential data via a flexible nonlinear mapping from the inputs to the outputs.

Attempts on time series modelling using RNNs have demonstrated the efficacy of RNNs in various time series prediction tasks, such as dynamic location prediction [52,57] and user satisfaction prediction [33]. In the aforementioned applications, a single RNN is leveraged to learn discriminative hidden states from the raw sequential inputs, and the last hidden state in a sequence is used to generate the desired output. As real-life tasks get more complex, the one-step prediction result generated from the last hidden state of a single RNN no longer suits the demand. Consequently, the encoder–decoder network is first proposed in neural machine translation scenarios [13,43], which further inspires relevant researches on multi-step ahead time series prediction [2,3,41].

2.2 Trend modelling

With the repetitive patterns in different time series, the discovery of recurring trends in time series is worth more investigations [12,34,35,40]. Unfortunately, these methods are either too rough to capture the subtle trend in sales time series or can only be applied to periodic trends within the time series. Besides, it is doubtful whether these approaches can be effectively embedded into the network structure. On the basis of encoder–decoder RNN structure, several attention mechanisms are designed to align the output state with relevant encoded hidden states, thus selectively picking valuable contextual information to enhance the model's performance [1,37,50]. However, these attention mechanisms are incompatible with the requirement of trend alignment in sale time series because of the unknown state of influential factors and the timely interaction between semantically different influential factors (i.e. internal and external features). Recently, a framework incorporating a regular RNN layer and a recurrent layer with skipping schemes is developed in [28] to capture repetitive trends in the time series. However, the skipping step size in [28] needs to be either observed from the data or obtained with a manual tuning process, which lacks enough flexibility to tackle the irregular patterns in sales time series.

2.3 Multitask learning

Machine learning models, especially deep neural network-based models, rely on large numbers of labelled samples to fully optimize their parameters in order to achieve optimal performance [56]. However, in many applications, data insufficiency problem inevitably arises, rendering the deep neural networks hard to generate rich representations from the inputs. In this light, multitask learning (MTL) is proposed as an important machine learning paradigm. The main purpose of MTL is to enhance the model performance via multiple learning sources, or to improve model generalizability on a specific task using other related tasks [32].

In the context of deep neural networks, MTL has shown the promising results in various applications, such as textual representation learning [31,32], graph-based recommendation [55] and speech modelling [20,22]. In [15], the idea of MTL is first brought to sequence-to-sequence learning, where one RNN-based encoder is used to extract sentence representation while multiple decoders are deployed to generate translations in different languages simultaneously. Apart from this one-to-many MTL approach, [32] further extends the MTL scheme with the many-to-many and many-to-one encoder–decoder architectures. The many-to-one

architecture consists of multiple encoders for different sequential inputs from different tasks, and one decoder to compute the outputs for all tasks. However, since there are various objective functions to be optimized at the same time, the decoder is forced to achieve a performance trade-off among all tasks. In contrast, our multitask encoder contains a synergic LSTM that combines the learned features from different contexts into a unified representation, and the decoder only focuses on the estimation of upcoming sales. On the one hand, the multitask encoder holds expressive power [56] by learning representations from different sources. On the other hand, the decoder is dedicated to only one objective, which ensures that the sales prediction performance can be fully maximized.

3 TADA: the model

In this section, we first mathematically formulate the definition of sales prediction, and then, we present the technical details of our proposed model TADA. Finally, we introduce the loss function and optimization strategy.

3.1 Problem formulation

The objective of sales prediction is to predict future sales volume according to multivariate observations (e.g. previous sales, weather, price, promotion, etc.) from the past. The formulation of sales prediction is similar to, but different from, multivariate time series forecasting and autoregressive models (AR). Formally, for an arbitrary product, the input is defined as its fully observed feature vector set $\{\mathbf{x}_t\}_{t=1}^{T} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T\}$ and the corresponding sales volume $\{y_t\}_{t=1}^{T} = \{y_1, y_2, ..., y_T\}$ at time step t. Here, $\mathbf{x}_t \in \mathbb{R}^n$, $y_t \in \mathbb{R}$ and n is variable according to the feature dimension, while T is the amount of total time steps. The output of sales prediction is the estimated sales volume of following Δ time steps after T, denoted as $\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = \{\hat{y}_{T+1}, \hat{y}_{T+2}, ..., \hat{y}_{T+\Delta}\}$, where Δ is adjustable according to the business goal. In this paper, we assume $\Delta \ll T$ to ensure the prediction accuracy because $\{\mathbf{x}_t\}_{t=T+1}^{T+\Delta}$ is non-available in the prediction stage.

Importantly, compared with multivariate time series forecasting and AR, sales prediction models behave differently. This is because our target is to acquire the one-dimensional scalar representing the sales volume without prior knowledge of the features in the future. Meanwhile, in multivariate time series forecasting, the output is specifically $\{\mathbf{x}_t\}_{t=T+1}^{T+\Delta}$, which has the same form and contextual meaning of its input [28]. Also, the AR assumes $\{\mathbf{x}_t\}_{t=T+1}^{T+\Delta}$ is available when predicting $\{\hat{y}_t\}_{t=T+1}^{T+\Delta}$ [37] because it is designed to model a mapping function between conditions and consequences.

Hence, we formulate sales prediction as a nonlinear mapping from time series features $\{\mathbf{x}_t\}_{t=1}^{T}$ and real sales $\{y_t\}_{t=1}^{T}$ in the history to the estimation of sales volume $\{\hat{y}_t\}_{t=T+1}^{T+\Delta}$ with Δ time steps ahead:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F\Big(\{\mathbf{x}_t\}_{t=1}^{T}, \{y_t\}_{t=1}^{T}\Big), \tag{1}$$

where $F(\cdot)$ is the nonlinear mapping function to learn.

3.2 Multitask encoder structure

Taking a time series $\{\mathbf{x}_t\}_{t=1}^{T}$ as input, recurrent neural network (RNN) encodes $\{\mathbf{x}_t\}_{t=1}^{T}$ into hidden states $\{\mathbf{h}_t\}_{t=1}^{T}$ via $\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$, where $f(\cdot)$ is a nonlinear mapping function.

To capture the long-range dependency, we leverage RNNs with long short-term memory architecture (LSTM) via the following formulation [21]:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f),\\
\mathbf{o}_t &= \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o),\\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c),\\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t),
\end{aligned} \tag{2}$$

where $\odot$ denotes element-wise multiplication and the recurrent activation σ is the *Logistic Sigmoid* function. $\mathbf{i}$, $\mathbf{f}$, $\mathbf{o}$ and $\mathbf{c}$ are, respectively, the input gate, forget gate, output gate and cell state vectors. When updating each of them, there are corresponding trainable input-to-hidden and hidden-to-hidden weights $\mathbf{W}$ and $\mathbf{U}$ along with the bias vectors $\mathbf{b}$.

For sales prediction, ***internal feature*** and ***external feature*** are two kinds of features with different semantic meanings in sales time series. We use $\{\mathbf{x}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{x}_t^{\text{ext}}\}_{t=1}^{T}$ to denote the feature vectors of internal and external information in sales time series, respectively. As we discussed in previous sections, internal features carry information of intrinsic attributes directly linked with the product like store location and item category, while the external features store information of extrinsic attributes viewed as external influential factors like weather condition and holiday. As a result, a single LSTM structure may suffer from loss of contextual information as it maps all raw features into one unified space, as we will reveal in Sect. 5. Hence, we use two LSTMs in parallel to effectively capture the different semantics by treating internal and external feature modelling as two sub-tasks. Correspondingly, we extend the problem formulation in Eq. (1) as:

$$\{\hat{y}_t\}_{t=T+1}^{T+\Delta} = F\Big(\{\mathbf{x}_t^{\text{int}}\}_{t=1}^{T}, \{\mathbf{x}_t^{\text{ext}}\}_{t=1}^{T}, \{y_t\}_{t=1}^{T}\Big). \tag{3}$$

Figure 2 demonstrates our proposed encoder architecture. We use $\{\mathbf{h}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{h}_t^{\text{ext}}\}_{t=1}^{T}$ to denote the latent representations learned from $\{\mathbf{x}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{x}_t^{\text{ext}}\}_{t=1}^{T}$. After the hidden states are learned from both sub-tasks, we simultaneously feed those hidden states into a synergic LSTM layer to learn a joint representation, namely ***contextual vectors*** denoted by $\{\mathbf{h}_t^{\text{con}}\}_{t=1}^{T}$ at all T time steps in the sales time series. Furthermore, to enhance the expressive ability of the encoder, instead of adopting $\{y_t\}_{t=1}^{T}$ to calculate the prediction loss, we fuse $\{y_t\}_{t=1}^{T}$ with hidden states from both internal and external encoding LSTMs to calculate the input $\{\mathbf{x}_t^{\text{syn}}\}_{t=1}^{T}$ for the synergic layer:

$$\mathbf{x}_t^{\text{syn}} = \mathbf{W}_{\text{syn}}[\mathbf{h}_t^{\text{int}}; \mathbf{h}_t^{\text{ext}}; y_t] + \mathbf{b}_{\text{syn}}, \tag{4}$$

where $[\mathbf{h}_t^{\text{int}}; \mathbf{h}_t^{\text{ext}}; y_t]$ represents the concatenation of $\mathbf{h}_t^{\text{int}}$, $\mathbf{h}_t^{\text{ext}}$ and y_t while $\mathbf{W}_{\text{con}}$ and $\mathbf{b}_{\text{con}}$ are weights and biases to be learned. For notation convenience, we format the multitask encoder structure into the following equations:

$$\begin{aligned}
\mathbf{h}_t^{\text{int}} &= LSTM^{\text{int}}(\mathbf{x}_t^{\text{int}}, \mathbf{h}_{t-1}^{\text{int}}),\\
\mathbf{h}_t^{\text{ext}} &= LSTM^{\text{ext}}(\mathbf{x}_t^{\text{ext}}, \mathbf{h}_{t-1}^{\text{ext}}),\\
\mathbf{h}_t^{\text{con}} &= LSTM^{\text{syn}}(\mathbf{x}_t^{\text{syn}}, \mathbf{h}_{t-1}^{\text{con}}),
\end{aligned} \tag{5}$$

where $\text{LSTM}^{\text{int}}(\cdot)$, $\text{LSTM}^{\text{ext}}(\cdot)$ and $\text{LSTM}^{\text{syn}}(\cdot)$ denote internal, external and synergic LSTM encoders, respectively. Note that the trainable weights are not shared across different LSTM layers in our multitask encoder structure.

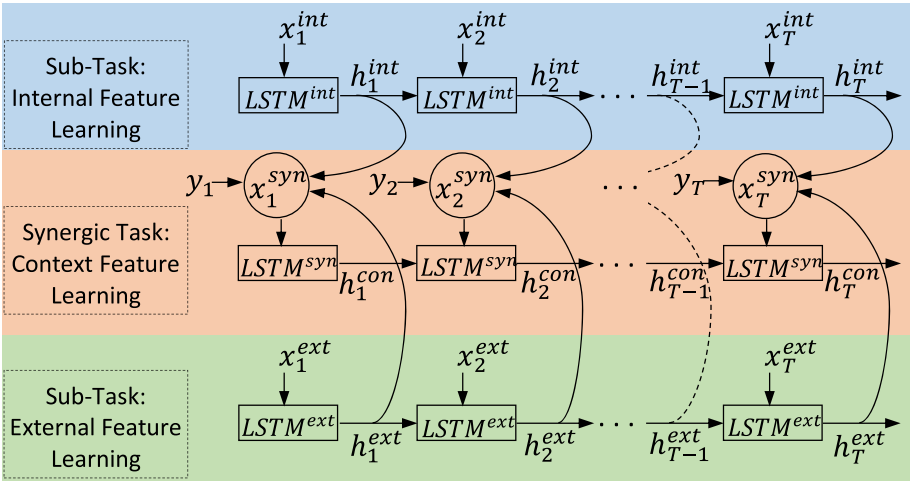


Fig. 2 Unfolded structure of our proposed multitask LSTM encoder. Two sub-tasks consist of internal feature learning and external feature learning LSTMs, denoted by LSTM^{int} and LSTM^{ext}, respectively. After latent representations of both internal and external features are generated, they are combined with the real sales number $\{\mathbf{y}_t\}_{t=1}^{T}$ to compute the contextual vectors $\{\mathbf{h}_t^{\text{con}}\}_{t=1}^{T}$ via the synergic task LSTM (LSTM^{syn})

3.3 Dual-attention decoder structure

After encoding the entire historical sales time series with the multitask encoder, we have the ***contextual vectors*** $\{\mathbf{h}_t^{\text{con}}\}_{t=1}^{T}$ where each $\mathbf{h}_t^{\text{con}}$ carries contextual information of the sales time series at time step t. The latent representations, $\{\mathbf{h}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{h}_t^{\text{ext}}\}_{t=1}^{T}$ for internal and external features are also learned. To predict the desired sales volume $\{\hat{y}_t\}_{t=T+1}^{T+\Delta}$, we adopt a LSTM decoder to mimic the ***contextual vectors*** in the following Δ time steps. Similar to Eq. (5), when $T < t \leq T + \Delta$, we have:

$$\mathbf{d}_t^{\text{con}} = LSTM^{\text{dec}}(\mathbf{x}_t^{\text{dec}}, \mathbf{d}_{t-1}^{\text{con}}), \tag{6}$$

where $\mathbf{d}_t^{\text{con}} \in \{\mathbf{d}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ is the contextual vector to learn in the decoding stage at time step t, $LSTM^{\text{dec}}(\cdot)$ is the decoder with the same formulation as Eq. (2), $\mathbf{x}_t^{\text{dec}}$ is the ***attention-weighted*** input for the decoder and $\mathbf{d}_{t-1}^{\text{con}}$ is the previous decoder hidden state.

3.3.1 Attention for weighted decoder input mapping

According to the problem formulation, we assume that both $\{\mathbf{x}_t^{\text{int}}\}_{t=T+1}^{T+\Delta}$ and $\{\mathbf{x}_t^{\text{ext}}\}_{t=T+1}^{T+\Delta}$ are non-available in the decoding stage because both of them contain attributes unknown to the future, such as price as an internal feature and weather as an external feature. Thus, to formulate the decoder input at time $t > T$, we propose an attention mechanism to dynamically select and combine relevant contextual vectors from $\{\mathbf{h}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{h}_t^{\text{ext}}\}_{t=1}^{T}$ with:

$$\mathbf{x}_t^{\text{dec}} = \mathbf{W}_{\text{dec}} \left[\sum_{t'=1}^{T} \alpha_{tt'}^{\text{int}} \mathbf{h}_{t'}^{\text{int}}; \sum_{t'=1}^{T} \alpha_{tt'}^{\text{ext}} \mathbf{h}_{t'}^{\text{ext}} \right] + \mathbf{b}_{\text{dec}}, \tag{7}$$

where $\alpha_{tt'}^{\text{int}}$ and $\alpha_{tt'}^{\text{ext}}$ denote the attention weights mapped to t'th hidden states of internal and external feature encoders, respectively. We use Fig. 3 to illustrate the attention for weighted

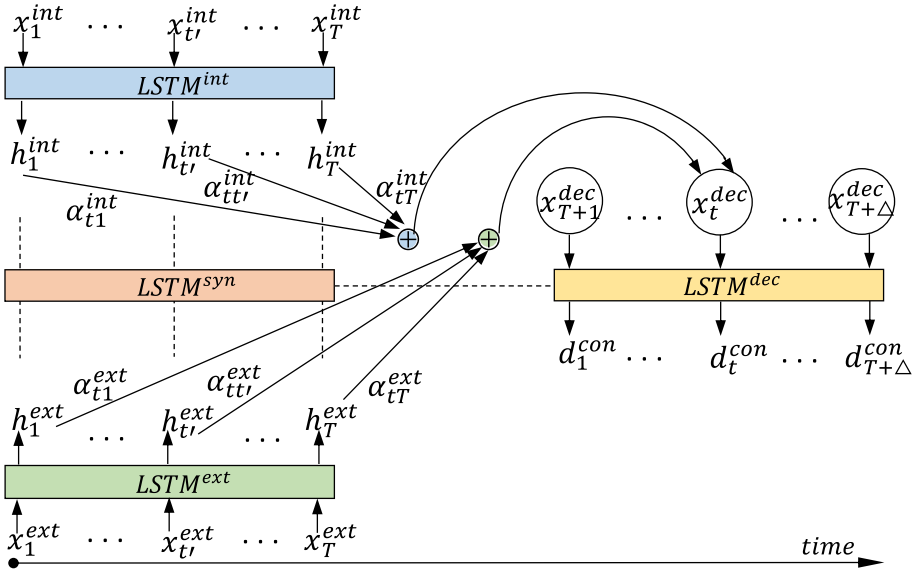


Fig. 3 Demonstration of proposed attention mechanism for weighted input mapping. The details of LSTM^{syn} are omitted for a clearer view. With the calculated attention weights $\alpha_{tt'}^{\text{int}}$ and $\alpha_{tt'}^{\text{ext}}$, the latent representations generated by LSTM^{int} and LSTM^{ext} are mapped into the input vectors $\{\mathbf{x}_t^{\text{dec}}\}_{t=T+1}^{T+\Delta}$ for the decoder LSTM^{dec}

decoder input mapping process. We enforce $\sum_{t'=1}^{T} \alpha_{tt'}^{\text{int}} = \sum_{t'=1}^{T} \alpha_{tt'}^{\text{ext}} = 1$, so that $[\cdot]$ in Eq. (7) can be viewed as the concatenation of two probability expectations from $\{\mathbf{h}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{h}_t^{\text{ext}}\}_{t=1}^{T}$. The rationale is that we simulate $\mathbf{x}_t^{\text{dec}}$ by summarizing varied influences from all $2T$ historical hidden states of both internal and external features. The influences are computed through quantifying the relevance between $\mathbf{d}_{t-1}^{\text{con}}$ and each $\mathbf{h}_{t'}^{\text{int}}$, $\mathbf{h}_{t'}^{\text{ext}}$:

$$\begin{aligned} e_{tt'}^{\text{int}} &= \mathbf{v}_{\text{int}}^{\top} \tanh\left(\mathbf{M}_{\text{int}} \mathbf{d}_{t-1}^{\text{con}} + \mathbf{H}_{\text{int}} \mathbf{h}_{t'}^{\text{int}}\right), \\ e_{tt'}^{\text{ext}} &= \mathbf{v}_{\text{ext}}^{\top} \tanh\left(\mathbf{M}_{\text{int}} \mathbf{d}_{t-1}^{\text{con}} + \mathbf{H}_{\text{ext}} \mathbf{h}_{t'}^{\text{ext}}\right), \end{aligned} \tag{8}$$

where $e_{tt'}^{\text{int}}$ and $e_{tt'}^{\text{ext}}$ are the relevance scores mapped to t'th hidden states in $\{\mathbf{h}_t^{\text{int}}\}_{t=1}^{T}$ and $\{\mathbf{h}_t^{\text{ext}}\}_{t=1}^{T}$ for the decoder input at time t, while $\mathbf{v}_{\text{int}}$, $\mathbf{v}_{\text{ext}}$, $\mathbf{M}_{\text{int}}$, $\mathbf{v}_{\text{ext}}$, $\mathbf{H}_{\text{int}}$ and $\mathbf{H}_{\text{ext}}$ are parameters to learn. In particular, Eq. (8) compares two hidden states with different semantic meanings. Intuitively, this is a scoring scheme that shows how well two vectors are correlated by projecting them into a common space. Afterwards, we apply $SoftMax$ on both attention weights:

$$\begin{aligned} \alpha_{tt'}^{\text{int}} &= \frac{\exp(e_{tt'}^{\text{int}})}{\sum_{s=1}^{T} \exp(e_{ts}^{\text{int}})}, \\ \alpha_{tt'}^{\text{ext}} &= \frac{\exp(e_{tt'}^{\text{ext}})}{\sum_{s=1}^{T} \exp(e_{ts}^{\text{ext}})}, \end{aligned} \tag{9}$$

which enforces $\sum_{t'=1}^{T} \alpha_{tt'}^{\text{int}} = \sum_{t'=1}^{T} \alpha_{tt'}^{\text{ext}} = 1$.

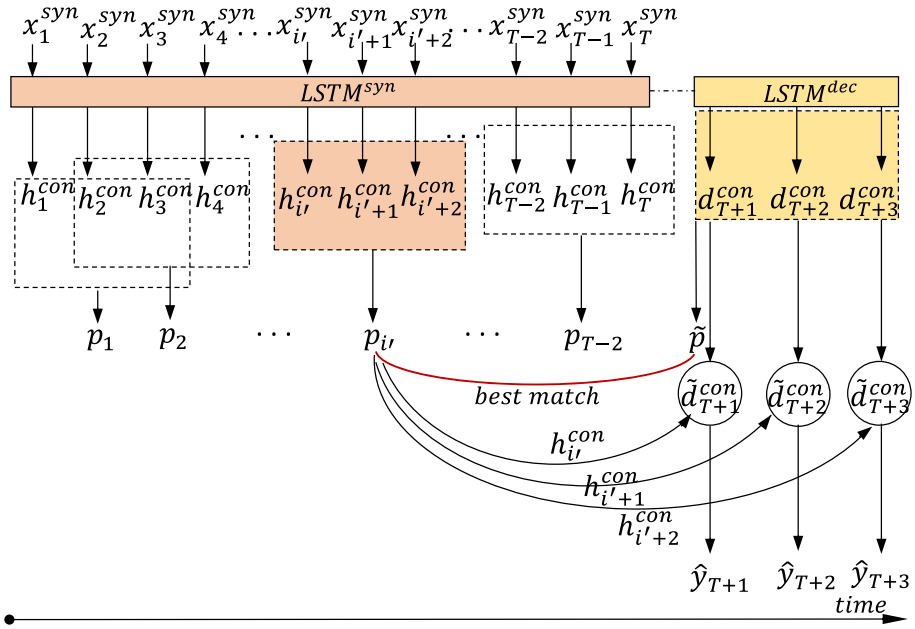


Fig. 4 Demonstration of proposed attention mechanism for trend alignment. The process for generating output label is included as well. We omit LSTM^{int} and LSTM^{ext} to be succinct. Note that we assume $\Delta = 3$ in this figure for better readability. The essence is to find a best match denoted by $\mathbf{p}_{i'}$ for the current trend $\widetilde{\mathbf{p}}$. Afterwards, we sequentially join the aligned contextual vector pairs within two trends to produce the final contextual vectors $\{\widetilde{\mathbf{d}}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ and then predict the upcoming sales $\{\hat{y}_t\}_{T+1}^{T+\Delta}$

3.3.2 Attention for trend alignment

Ideally, at time t, each acquired contextual vector in $\{\mathbf{h}_t^{\text{con}}\}_{t=1}^{T}$ and $\{\mathbf{d}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ carries contextual information of both time t and previous time steps. However, as discussed in [13,37], the performance of the encoder–decoder networks decreases significantly when the length of time series grows. To alleviate the problem, traditional attention mechanisms have been designed to align the current output with the targeted input by comparing the current hidden state with the ones generated at previous time steps. Meanwhile, these methods are not applicable as we aim to match similar trends for the prediction period Δ, and we propose a novel attention mechanism for trend alignment. Mathematically, we represent a Δ-step trend in sales time series as the concatenation of Δ successive contextual vectors in $\{\mathbf{h}_t^{\text{con}}\}_{t=1}^{T}$:

$$\mathbf{p}_i = \left[\mathbf{h}_i^{\text{con}}; \mathbf{h}_{i+1}^{\text{con}}; ...; \mathbf{h}_{i+\Delta-1}^{\text{con}}\right], \quad 1 \le i \le T - \Delta + 1 \tag{10}$$

where $\mathbf{p}_i$ denotes the ith trend in T with a time span of Δ. Similarly, we represent the upcoming trend $\widetilde{\mathbf{p}}$ in the $[T+1, T+\Delta]$ time interval via the concatenation of all contextual vectors in $\{\mathbf{d}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$:

$$\widetilde{\mathbf{p}} = \left[\mathbf{d}_{T+1}^{\text{con}}; \mathbf{d}_{T+2}^{\text{con}}; ...; \mathbf{d}_{T+\Delta}^{\text{con}}\right]. \tag{11}$$

We explain the workflow of attention for trend alignment in Fig. 4. As demonstrated in Fig. 4, when the trend index i increases from 1 to $T - \Delta + 1$, $\mathbf{p}_i$ can be viewed as a sliding window that dynamically captures temporary contextual informa-

tion learned from existing sales time series with respective step and window size as 1 and Δ. Hence, we compute the relevance score between $\widetilde{\mathbf{p}}$ and each $\mathbf{p}_i \in \{\mathbf{p}_i\}_{i=1}^{T-\Delta+1}$, with:

$$e_i^{\text{trd}} = \mathbf{p}_i^\top \widetilde{\mathbf{p}} \tag{12}$$

and then find out the best match of $\widetilde{\mathbf{p}}$:

$$i' = \text{argmax}\left(e_i^{\text{trd}}, e_{i+1}^{\text{trd}}, ..., e_{T+\Delta-1}^{\text{trd}}\right), \tag{13}$$

where e_i^{trd} denotes the relevance between $\widetilde{\mathbf{p}}$ and $\mathbf{p}_i$, while i' indicates the i'th trend in $\{\mathbf{p}_i\}_{i=1}^{T-\Delta+1}$ is the most relevant to $\widetilde{\mathbf{p}}$. Because $\widetilde{\mathbf{p}}$ and $\mathbf{p}_i$ express similar contextual semantics with the same dimensionality, we do not use the scoring scheme in Eq. (8) but adopt the dot product to be computational efficient. Intuitively, the closer $\widetilde{\mathbf{p}}$ and $\mathbf{p}_i$ are, a larger e_i^{trd} will be generated and vice versa ($e_i^{\text{trd}} = 0$ when orthogonal), so we can align the upcoming trend $\widetilde{\mathbf{p}}$ with its best match $\mathbf{p}_{i'} = [\mathbf{h}_{i'}^{\text{con}}; \mathbf{h}_{i'+1}^{\text{con}}; ...; \mathbf{h}_{i'+\Delta-1}^{\text{con}}]$.

More importantly, now the contextual vectors within both trends, i.e. $\{\mathbf{d}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ and $\{\mathbf{h}_t^{\text{con}}\}_{t=i'}^{i'+\Delta-1}$ are also aligned as trend components instead of individual hidden states. With the upcoming sales trend $\widetilde{\mathbf{p}}$ aligned with the i'th historical trend, we merge each pair of contextual vector in $\{\mathbf{d}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ and $\{\mathbf{h}_t^{\text{con}}\}_{t=i'}^{i'+\Delta-1}$ into the aligned representation of contextual vectors:

$$\begin{aligned} &\widetilde{\mathbf{d}}_t^{\text{con}} = \mathbf{W}_{\text{ali}}[\mathbf{d}_j^{\text{con}}; \mathbf{h}_k^{\text{con}}] + \mathbf{b}_{\text{ali}}, \\ &T+1 \le j \le T, \quad i' \le k \le i'+\Delta-1, \end{aligned} \tag{14}$$

where $\widetilde{\mathbf{d}}_t^{\text{con}}$ is the aligned contextual vectors at time t, $\mathbf{W}_{\text{ali}}$ and $\mathbf{b}_{\text{ali}}$ are parameters to learn and $[\mathbf{d}_j^{\text{con}}; \mathbf{h}_k^{\text{con}}]$ is the concatenation of aligned contextual vector pair. We use the following algorithm to acquire the full set of aligned contextual vectors for sales prediction:

Algorithm 1 Generating Aligned Contextual Vectors

1: **Input:** prediction time steps Δ; aligned trend index i'; encoded time length T; sales contextual vectors $\{\mathbf{d}_t^{con}\}_{t=T+1}^{T+\Delta}$ and $\{\mathbf{h}_t^{con}\}_{t=i'}^{i'+\Delta-1}$
2: **Output:** aligned representations of contextual vectors $\{\widetilde{\mathbf{d}}_t^{con}\}_{t=T+1}^{T+\Delta}$
3: initialize with $j = T+1, k = i'$;
4: **while** $j \le T+\Delta$ **and** $k \le i'+\Delta-1$ **do**
5: update $\widetilde{\mathbf{d}}_t^{con}$ via Eq. (14);
6: $j++$;
7: $k++$;
8: **end**

Here, $\{\widetilde{\mathbf{d}}_t^{\text{con}}\}_{t=T+1}^{T+\Delta} = \{\widetilde{\mathbf{d}}_{T+1}^{\text{con}}, \widetilde{\mathbf{d}}_{T+2}^{\text{con}}, ..., \widetilde{\mathbf{d}}_{T+\Delta}^{\text{con}}\}$ contains the final latent representation at each upcoming time step in the simulated sales context.

3.4 Sales prediction and model optimization

With the aligned contextual vectors $\{\widetilde{\mathbf{d}}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$ generated, we approximate the future sales with regression:

$$\hat{y}_t = \mathbf{v}_y^\top \widetilde{\mathbf{b}}_t^{\text{con}} + b_y, \tag{15}$$

where $\hat{y}_t \in \{\hat{y}_t\}_{t=T+1}^{T+\Delta}$ denotes the predicted sales at time t and $\mathbf{v}_y^\top$ and b_y are parameters to learn.

For model learning, we apply the simple yet effective mean squared error coupled with *L2* regularization (to prevent overfitting) on model parameters:

$$\mathcal{L}_F = \frac{1}{N}\left(\sum_{n=1}^{N}\sum_{t=T+1}^{T+\Delta}(\hat{y}_{nt} - y_{nt})^2\right) + \lambda\sum_{l}^{L}\theta_l^2, \tag{16}$$

where $n \leq N$ is the number of training samples, $l \leq L$ is the index of model parameters, y_{nt} is the actual label of sales at tth time step, θ_l is the model parameter and λ is the weight decay coefficient that needs to be tuned.

In the training procedure, we leverage a mini-batch stochastic gradient descent (SGD) algorithm, namely Adam [25] optimizer. Specifically, we set the batch size as 128 according to device capacity and the start learning rate as 0.001 which is reduced by 10% after every 10,000 iterations. We iterate the whole training process until the loss converges.

3.5 Similarity-based reservoir for online model update

In this part, we extend our proposed offline model TADA to an online setting, which is named TADA$^+$. As discussed in Sect. 1, as the sales companies usually track their timely sales for many purposes, sales data may arrive in a streaming manner, thus offering new training samples for TADA. On the one hand, a desirable model should be able to promptly adapt to the new incoming data by updating the model using the latest data, so as to provide timely and precise predictions. On the other hand, a desirable model should be able to automatically retain useful information in the past sales time series, such that the important trend patterns would be retained during the retraining process. Though it is a straightforward way to directly combine the new data with existing data to construct a new training set to update the parameters in the model, the growing size of the training data will soon become unmanageable. Hence, instead of storing all of all the training samples, we propose to leverage the data reservoir to keep a portion of the training samples which are the most useful from the past.

We use $\mathcal{S}$ to denote the reservoir that contains all previous training samples. When the new training samples in $\mathcal{S}_{\text{new}}$ arrive, our target is to select m samples from $\mathcal{S}$ denoted by $\mathcal{S}_{\text{hard}} \in \mathcal{S}$ to construct an updated reservoir $\mathcal{S}' = \mathcal{S}_{\text{hard}} \cup \mathcal{S}_{\text{new}}$. A naive approach for such reservoir construction is shown in Algorithm 2. Then, the updated reservoir $\mathcal{S}'$ will be used to resume the training process of TADA and update the parameters.

However, updating model parameters with Algorithm 2 can hardly help the model achieve its optimal performance. This is because the random sampling strategy for reservoir construction treats all the past training samples equally, and a lot of "easy" prediction tasks and outliers might be selected to retrain the model. Eventually, the model will be subject to degraded performance over time. As shown in Eqs. (9) and (13), to generate accurate prediction results, our model relies on the dual-attention mechanism scheme that leverages historical contextual

Algorithm 2 Online Model Update with Random Sampling-based Reservoir

1: **Input:** the current model $\mathcal{M}$, the current reservoir $\mathcal{S}$, new data $\mathcal{S}_{new}$
2: **Output:** the updated reservoir $\mathcal{S}'$, the updated model the current model $\mathcal{M}_{new}$
3: **if** the last training epoch has finished **then**
4: fetch m samples from $\mathcal{S}_{new}$ with uniform random sampling, denoted as $\mathcal{S}_{rand}$;
5: $\mathcal{S}' = \mathcal{S}_{rand} \cup \mathcal{S}_{new}$;
6: retrain model $\mathcal{M}$ with $\mathcal{S}'$ and Adam optimizer, return $\mathcal{M}_{new}$;
7: **end**

Algorithm 3 Online Model Update with Similarity-based Reservoir

Input: the current model $\mathcal{M}$, the current reservoir $\mathcal{S}$, new data $\mathcal{S}_{new}$
Output: the updated reservoir $\mathcal{S}'$, the updated model the current model $\mathcal{M}_{new}$
initialize with $\mathcal{D} = \varnothing$;
while in the last training epoch **do**
  set $n = 1$;
  while data sample $s_n \in \mathcal{S}$ **do**
    set $t = T + 1$;
    while $t \leq T + \Delta$ **do**
      set $d'_n = 0$
      compute the internal feature attention scores $\alpha^{int}_{t1}, \alpha^{int}_{t2}, ..., \alpha^{int}_{tT}$ and external feature attention scores $\alpha^{ext}_{t1}, \alpha^{ext}_{t2}, ..., \alpha^{ext}_{tT}$ with Eq. (9), denoted by $\boldsymbol{\varrho}^{int}_t$ and $\boldsymbol{\varrho}^{ext}_t$, respectively;
      compute $D_{KL}(\boldsymbol{\varrho}^{int}_t|\boldsymbol{\xi}) + D_{KL}(\boldsymbol{\varrho}^{ext}_t|\boldsymbol{\xi})$ with Eq. (17), where $|\boldsymbol{\varrho}| = |bm\xi|$ and $\boldsymbol{\xi} = [\frac{1}{|bm\xi|}; \frac{1}{|bm\xi|}; ...; \frac{1}{|bm\xi|}]$, denoted by d^{sim}_t;
      $d'_n = d^{sim}_t + d'_n$;
    compute trend scores $\alpha^{trd}_1, \alpha^{trd}_2, ..., \alpha^{trd}_{T+\Delta-1} = softmax(e^{trd}_1, e^{trd}_2, ..., e^{trd}_{T+\Delta-1})$, denoted by $\boldsymbol{\varrho}^{trd}_n$;
    compute $D_{KL}(\boldsymbol{\varrho}^{trd}_n|\boldsymbol{\xi})$, where $|\boldsymbol{\varrho}| = |bm\xi|$ and $\boldsymbol{\xi} = [\frac{1}{|bm\xi|}; \frac{1}{|bm\xi|}; ...; \frac{1}{|bm\xi|}]$, denoted by d^{trd}_n;
    compute $d_n = d^{trd}_n + d'_n$;
    $d_n \mapsto \mathcal{D}$;
sort each $d_n \in \mathcal{D}$ in ascending order;
fetch the indexes of last m elements in $\mathcal{D}$, denoted by $\mathcal{N}$;
$\mathcal{S}_{hard} = \{s_n\}^{\forall n \in \mathcal{N}}_n$;
$\mathcal{S}' = \mathcal{S}_{hard} \cup \mathcal{S}_{new}$;
retrain model $\mathcal{M}$ with $\mathcal{S}'$ and Adam optimizer, return $\mathcal{M}_{new}$;
end

information to compensate for unknown future inputs as well as match the upcoming trend with existing ones. Hence, to select a fraction of the hardest training cases $\mathcal{S}_{\text{hard}}$ from $\mathcal{S}$, instead of treating all training cases equally with random sampling [14,46], we propose a similarity-based reservoir construction strategy. In general, both attention mechanisms aim to pick up information from historical contextual vectors based on how similar the current contextual vector is to them. As a result, the distribution of similarity scores in Eqs. (9) and (13) reflects how difficult it is for the model to distinguish the importance of difference contextual vectors. Intuitively, if the similarity scores are close to each other, it means the model experiences difficulties in choosing the most relevant contextual information for weighted decoder input mapping or trend alignment. Correspondingly, we utilize the Kullback–Leibler (KL) divergence [27] defined as follows:

$$D_{\text{KL}}(\boldsymbol{\varrho}|\boldsymbol{\xi}) = \sum_{\forall i} \ln\left(\frac{\varrho_i}{\xi_i}\right), \tag{17}$$

where $|\boldsymbol{\varrho}| = |\boldsymbol{\xi}|$ are two probability distribution vectors. Then, if we set $\boldsymbol{\xi} = [\frac{1}{|\boldsymbol{\xi}|}; \frac{1}{|\boldsymbol{\xi}|}; ...; \frac{1}{|\boldsymbol{\xi}|}]$, $D_{KL}(\boldsymbol{\varrho}|\boldsymbol{\xi})$ will represent the closeness between the probability distribution in $\boldsymbol{\varrho}$ and normal distribution. Then, based on the KL divergence, we devise Algorithm 3 to select m used training samples. Afterwards, the selected m samples are fused with the new data samples to construct the updated reservoir $\mathcal{S}'$, which is eventually used to retrain and update the model parameters. Note that in practice, we set m as 20% of the size of the previous training data to achieve a trade-off between storage capacity and model performance.

4 Time complexity analysis

Because the proposed multitask, dual-attention RNN model is heavily associated with multiple parameters, here we discuss the model time complexity in detail. As the majority of time consumption is associated with the training process rather than the reservoir construction, we discuss the time complexity of the offline version, i.e. TADA in this section. We prove that like a standard LSTM system, with the model parameters fixed, the asymptotic time complexity of TADA is linear to the size of data.

For a basic LSTM cell in Eq. (2), we denote the number of hidden dimensions as q (i.e. $\mathbf{h} \in \mathbb{R}^{q\times 1}$). According to [21,36], ignoring the biases, a single-task LSTM with T time steps has the complexity of $O(q^2T)$. Similarly, we formulate the time complexity for our encoder–decoder structure. Assuming all LSTMs in TADA have q hidden dimensions, and the multitask encoder structure with LSTM^{int}, LSTM^{ext} and LSTM^{syn} are deployed in parallel, the time complexity is $O\big(q^2(T+\Delta)\big)$, which is identical to a basic encoder–decoder LSTM structure.

Then, we focus on the dual-attention mechanism. Since Eq. (8) can be viewed as two parallel feed-forward networks, the complexity is $O(q^2)$ for each time step. Coupled with Eq. (7), the time complexity of attention mechanism in Sect. 3.3.1 is $O(q^2T\Delta + q^2\Delta) = O\big(q^2(T+1)\Delta\big) \simeq O(q^2T\Delta)$. According to [44], dot product-based attention mechanism in Eq. (12) has the complexity of $O\big(q\Delta(T-\Delta+1)\big) \simeq O(qT\Delta - q\Delta^2)$. Combining with Eq. (14), the overall complexity of attention mechanism in Sect. 3.3.2 is $O(q^2\Delta + qT\Delta - q\Delta^2)$.

With the complexity of encoder–decoder and dual-attention mechanism sorted, we aggregate the complexity for generating the aligned contextual vectors $\{\widetilde{\mathbf{d}}_t^{\text{con}}\}_{t=T+1}^{T+\Delta}$. Note that the complexity of Eq. (4) throughout time T is $O(q^2T)$, and the complexity of Eq. (15) throughout time Δ is $q\Delta^2$. Finally, the overall complexity of TADA comes to $O\big(2q^2(T+\Delta) + qT(q+\Delta)\big)$. In practice, we have $\Delta \ll T$ and $\Delta \ll q$, so T and q are dominating in dimensionality. Therefore, we simplify the final time complexity as $O(3q^2T) \to O(q^2T)$. For a dataset with N samples (time series), it takes $O(Nq^2T)$ to go through the entire dataset once. In summary, when the hidden dimension q and total time step T are fixed, the time complexity of TADA is linearly associated with the scale of the data.

5 Experiments

In this section, we conduct experiments on real commercial datasets to showcase the advantage of TADA in the task of sales prediction. In particular, we aim to answer the following research questions via the experiments:

- (RQ1) Under the offline setting, how effectively and accurately TADA can predict continuous sales volume with observed sales time series from the past.
- (RQ2) How far into the future can TADA generate accurate predictions.
- (RQ3) Under the online setting, how the online update scheme helps TADA^+ deal with incoming data for sales prediction.
- (RQ4) How TADA and TADA^+ benefits from each component of the proposed structure for sales prediction.
- (RQ5) How efficiently our proposed model can be trained when handling training data with different sizes.

Table 1 Statistics of datasets in use

Dataset	Time series	Granularity	Time range	Variables
Favorita	11,536	1 day	365 days	13
OSW	1585	1 week	106 weeks	11

5.1 Datasets and features

To validate the performance of both TADA and TADA$^+$, we use two real-life commercial datasets shown in Table 1, namely **Favorita** and **OSW**. Here, we briefly introduce the properties of these two datasets below:

- *Favorita* It contains the daily features and sales volume of all products in 56 Ecuadorian-based grocery stores. Note that the original Favorita dataset covers the time range from 1 January 2013 to 15 August 2017, but we only use the portion from 15 August 2016 to 15 August 2017 (365 days) due to two reasons: (1) a magnitude 7.8 earthquake struck Ecuador on 16 April 2016, which exerted abnormal sales patterns in the following few weeks[1]; (2) shorter time series suits the real-life conditions better as it is faster for the model to learn.
- *OSW* One Stop Warehouse[2] is one of the largest solar energy appliance suppliers in Australia. The dataset covers 12 warehouses' weekly sales volume of various products (e.g. solar panels, batteries, etc.) from 22 February 2016 to 4 March 2017 (106 weeks). Empirically, sales prediction on OSW dataset is more challenging from two perspectives: (1) the sales volume of solar energy appliances is more dependent on external causes (e.g. policy, electricity price, promotion, etc.), which are unavailable in this dataset; (2) the sales volume in OSW dataset fluctuates more significantly than Favorita.

The features we used from the datasets are listed in Table 2. Features consist of binary (represented as 1 or 0), categorical (represented via one-hot vectors) and numerical data, which are marked by superscripts of b, c and n, respectively. To accelerate the training process, we process all the numerical features by performing $\log_{10}$ transfer (a small bias of 0.001 was added to all numerics to avoid the case of 0). In addition, we leverage embedding to reduce the original dimensionality of categorical data and combine all these features as the model input. As suggested by the Tensorflow research team from Google,[3] we set the embedding dimension of each categorical feature by taking the 4th root of the total amount of categories. In Table 2, numbers with "$*$" mean the dimension of embedding for categorical features.

In both datasets, each time series is actually a log file for a specific product. Hence, we do not split different products up for training and test because it means many products are totally new to the model during the test, which is not realistic in real business. So, we first randomly take 3000 and 400 time series out of Favorita and OSW dataset for validation. Then, given time series with the total time steps of M (365 for Favorita and 106 for OSW) and Δ steps to predict, we apply the "walk-forward" split strategy on the remaining data. For training, we encode the information with $t \in [1, M-2\Delta]$ and predict sales with $t \in [M-2\Delta+1, M-\Delta]$. For evaluation, we encode the information with $t \in [\Delta+1, M-\Delta]$ and predict sales with $t \in [M-\Delta+1, M]$ to test the accuracy. This test strategy has more practical meaning in the

[1] https://www.kaggle.com/c/favorita-grocery-sales-forecasting/data.

[2] https://www.onestopwarehouse.com.au.

[3] https://developers.googleblog.com/2017/11/introducing-tensorflow-feature-columns.html.

Table 2 Summarization of features extracted from datasets

Dataset	Type	Feature	Dimension	
Favorita	Internal feature	City of store[c]	3∗	17
		State of store[c]	2∗	
		Store type[c]	2∗	
		Store group[c]	2∗	
		Item family[c]	3∗	
		Item class[c]	5∗	
	External feature	Promotion state[b]	1	11
		Date[c]	5∗	
		Store transaction[n]	1	
		Oil price[n]	1	
		Local holiday[b]	1	
		National holiday[b]	1	
		Pay day[b]	1	
OSW	Internal feature	Item index[c]	5∗	16
		City of store[c]	2∗	
		Item category[c]	5∗	
		Battery type[c]	3∗	
		Item price[n]	1	
	External feature	Week number[c]	4∗	9
		Discontinued state[b]	1	
		Solar exposure[n]	1	
		Temperature[n]	1	
		Week(s) after last holiday[n]	1	
		Week(s) to next holiday[n]	1	

real world, where most businesses tend to predict future sales volume according to previous records.

5.2 Baseline methods

We conduct experiments against the following state-of-the-art predictive methods:

- *Random Forest (RF)* We implement a widely used, predictive decision tree model, namely random forest to predict sales from the observed features.
- *XGBoost* It stands for extreme gradient boosting, proposed by Chen et al. [9]. It is a state-of-the-art, gradient boosted regression tree approach based on the gradient boosting machine (GBM) [16].
- *SAE-LSTM* From the cutting edge of economics research, we adpot the stacked autoencoder with LSTM (SAE-LSTM) [2] which is a neural network-based model proposed for financial time series prediction.
- *A-RNN* Attention RNN (A-RNN) was originally designed by Bahdanau et al. for machine translation tasks [1], with the output of a probability distribution over the word dictionary.

We modify the output layer by mapping the learned hidden states into scalar values and use the loss function in Eq. (16) for the sales prediction task.

- *DA-RNN* This is a nonlinear autoregressor (AR) with attentions in both encoder and decoder RNNs [37]. Compared with A-RNN, the proposed encoder attention in DA-RNN assumes the inputs must be correlated along the time, which is not always true in sales time series.
- *LSTNet* It is a deep learning framework (long- and short-term time series network) designed for multivariate time series prediction [28]. This method combines a convolutional neural network with a recurrent-skip network to capture both short-term and long-term trending patterns of the time series.

Furthermore, to fully study the performance gain from each component of our proposed model, we implement three degraded versions of TADA and conduct ablation tests under the offline setting:

- *TADA-SE* We replace the multitask encoder with a single-task, 1-layer LSTM encoder. The internal and external feature vectors are concatenated as the input of the single-task encoder.
- *TADA-SA*$_1$ We remove the first attention mechanism in Sect. 3.3.1 for decoder input mapping to build a single-attention variant.
- *TADA-SA*$_2$ We remove the second attention mechanism in Sect. 3.3.2 for trend alignment to build another single-attention variant.

Note that when conducting experiments under the online setting, for a fair comparison we adopt the random sampling-based reservoir in Algorithm 2 to update the parameters in baseline models and TADA. In addition, we also test the prediction accuracy of TADA without retraining, abbreviated as **TADA(w/o retrain)**.

5.3 Parameters and experimental settings

In TADA, we apply the same size to the hidden states of all LSTM systems to maintain the consistency of the contextual feature dimension. That is to say, there are only two hyperparameters in TADA to be determined, namely the size of hidden states and the weight decay penalty λ. We conduct grid search for the number of hidden states and λ over $\{32, 64, 128, 256, 512\}$ and $\{0.001, 0.01, 0.1, 1, 10\}$, respectively. The settings with the best performance on the validation set ($\lambda = 0.01$ on Favorita, $\lambda = 0.1$ on OSW, and 128 hidden states for both datasets) are used in the test.

To measure the overall effectiveness of all the methods in sales prediction under both online and offline settings, we adopt two evaluation metrics, namely mean absolute error (MAE) and symmetric mean absolute percentage error (SMAPE), which are widely used in relevant tasks [8,47]. Mathematically, they are defined as follows:

$$\begin{aligned} \textbf{MAE} &= \frac{1}{N \times \Delta} \sum_{n=1}^{N} \sum_{t=T+1}^{T+\Delta} |y_t - \hat{y}_t|, \\ \textbf{SMAPE} &= \frac{100\%}{N \times \Delta} \sum_{n=1}^{N} \sum_{t=T+1}^{T+\Delta} \left(\begin{array}{c} 0, \quad if \ y_t = \hat{y}_t = 0 \\ \frac{|y_t - \hat{y}_t|}{(|y_t| + |\hat{y}_t|)/2}, otherwise \end{array} \right), \end{aligned} \tag{18}$$

where y_t and $\hat{y}_t$ denote real and predicted sales volume, respectively. We choose them because MAE is scale-dependent while SMAPE is not, so MAE is suitable for comparison of different

Table 3 Sales prediction results under the offline setting

Dataset	Method	$\Delta = 2$		$\Delta = 4$		$\Delta = 8$	
		MAE	SMAPE (%)	MAE	SMAPE (%)	MAE	SMAPE (%)
Favorita	RF	32.483	200 (max)	35.507	200 (max)	41.329	200 (max)
	XGBoost [9]	16.705	87.433	19.833	91.230	22.547	158.461
	SAE-LSTM [2]	7.364	39.447	8.033	44.384	8.116	46.932
	A-RNN [1]	11.610	60.781	12.226	62.397	13.005	65.812
	DA-RNN [37]	7.816	43.859	8.234	44.704	8.566	46.281
	LSTNet [28]	7.419	43.523	7.982	45.662	8.729	48.469
	TADA-SE	9.995	58.715	11.076	60.332	10.955	60.257
	TADA-SA_1	8.152	46.732	8.273	43.951	8.968	49.079
	TADA-SA_2	7.635	42.883	8.247	44.942	8.626	48.609
	TADA	**6.955**	**38.770**	**7.323**	**40.588**	**7.422**	**43.675**
OSW	RF	29.147	89.482	35.576	137.892	43.096	200(max)
	XGBoost [9]	21.496	49.556	24.916	53.243	30.322	82.633
	SAE-LSTM [2]	17.828	44.241	19.805	46.887	20.823	49.873
	A-RNN [1]	17.391	44.635	18.823	44.603	22.129	49.180
	DA-RNN [37]	17.634	44.215	19.578	47.139	20.693	48.365
	LSTNet [28]	16.625	42.317	18.989	45.782	21.246	49.191
	TADA-SE	19.635	53.017	20.884	49.370	21.687	51.685
	TADA-SA_1	16.585	42.620	18.624	44.331	21.699	51.195
	TADA-SA_2	17.087	42.199	18.643	45.219	21.190	49.825
	TADA	**15.418**	**41.354**	**17.572**	**43.265**	**19.618**	**47.782**

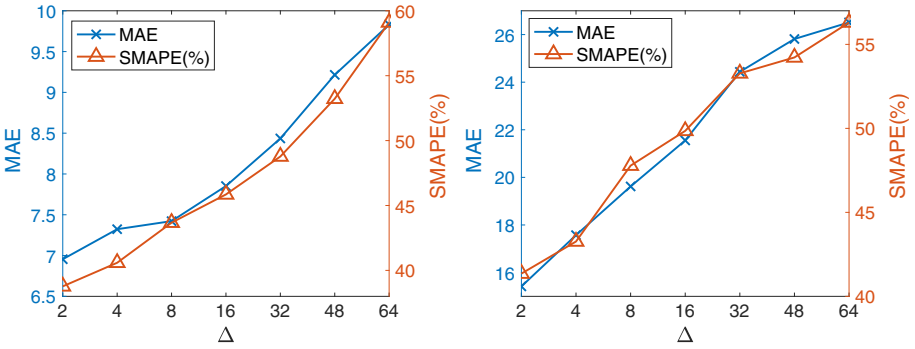
Numbers in boldface are the best results within each column

methods on the same dataset and SMAPE suits comparison across different datasets. In terms of online sales prediction with TADA$^+$, we adopt the uniform random sampling in Algorithm 2 to construct the data sample reservoir and update all the models used for comparison.

5.4 Discussion on offline sales prediction effectiveness (RQ1)

To thoroughly evaluate the predictive capability of TADA under the offline setting, we test all methods on two datasets with $\Delta \in \{2, 4, 8\}$ to showcase their robustness in multiple sales prediction scenarios. We report the results of all tested methods on all Δ settings in Table 3, where the best performance is highlighted with boldface. MAE measures the error with the deviation between predicted and real sales volume, and SMAPE quantifies such error with a proportional perspective. Noticeably, in both datasets, a small-scale MAE of predicted sales volume causes a relatively large percentage error (SMAPE), which indicates a high potential economic loss for the incorrect prediction results. This observation reflects the significance and necessity of ensuring the accuracy of sales prediction.

It is as expected that all neural network-based predictive models outperform decision tree-based models (RF and XGBoost) by a significant margin in both datasets. Hence, we can empirically suggest that deep neural networks better suit the task of sales prediction in the real-world scenario. Apparently, the performance of all methods starts to drop when we gradually increase the time range for sales prediction with $\Delta \in \{2, 4, 8\}$. However,



(a) Prediction results on Favorita. **(b)** Prediction results on OSW.

Fig. 5 Sales prediction effectiveness on both datasets w.r.t. different Δ values. Generally, the sales performance decreases as Δ increases, while TADA shows promising robustness when tackling longer prediction intervals

among this observation, TADA demonstrates the least negative impact from the increasing Δ and presents the dominating prediction performance against all state-of-the-art baselines. In other words, the trend alignment scheme from TADA can practically meet the requirement of sales prediction when merchants are trying to look ahead at more upcoming time steps. When comparing with other deep neural network-based approaches (SAE-LSTM, A-RNN, DA-RNN and LSTNet), the results also support the superiority of TADA. This is because: (1) the multitask encoder in TADA is better at capturing the interactive effect from both internal and external features to the real sales than modelling all influential factors in the unified way; (2) the dual-attention architecture in TADA successfully captures latent trends from the past which are similar to the upcoming one, especially when comparing with existing attention mechanisms (A-RNN and DA-RNN) and periodic trend modelling method (LSTNet). The effectiveness of each proposed component in TADA is initially revealed in Table 3 by its degraded versions, which we will further discuss in the following section.

5.5 Discussion on model sensitivity to prediction interval length (RQ2)

In Sect. 5.4, we verify the prediction effectiveness of TADA with a relatively small interval length Δ. While a small Δ can be practical and realistic for retail businesses to actively predict sales and adjust inventory, in some cases, it would also be beneficial for a retailer to foresee the sales volume in a longer prediction interval (e.g. monthly or quarterly sales) in order to devise long-term sales strategies. To further investigate the effect of a larger Δ on the prediction accuracy of TADA, we aggressively set $\Delta = \{2, 4, 8, 16, 32, 48, 64\}$ for both datasets and verify how the predictive performance can possibly deteriorate as Δ increases (Fig. 5).

We visualize the performance fluctuations of MAE and SMAPE on both the Favorita and OSW datasets. Note that as Δ increases, the time span of training data shortens accordingly. At the first glance, TADA yields a gradually decreasing prediction performance on both datasets, which is shown by the increasing MAE and SMAPE scores. Meanwhile, when $\Delta \geq 16$, the prediction performance on Favorita drops slightly quicker than that on OSW. The reason could be highly relevant to the properties of these two datasets. The sales volume in Favorita is updated on a daily basis, which is more sensitive to the timely contexts within different influential factors. As a result, a larger prediction interval length sets challenges for

Table 4 The range of t for online sales prediction

Dataset	Subset	Train		Test	
		Input t range	Output t range	Input t range	Output t range
Favorita	Set 1	[1, 333]	[334, 341]	[9, 341]	[342, 349]
	Set 2	[9, 341]	[342, 349]	[17, 349]	[350, 357]
	Set 3	[17, 349]	[350, 357]	[25, 357]	[358, 365]
OSW	Set 1	[1, 74]	[75, 82]	[9, 82]	[83, 90]
	Set 2	[9, 82]	[83, 90]	[17, 90]	[91, 98]
	Set 3	[17, 90]	[91, 98]	[25, 98]	[99, 106]

Table 5 Sales prediction results under the online setting

Dataset	Method	Test set 1		Test set 2		Test set 3	
		MAE	SMAPE (%)	MAE	SMAPE (%)	MAE	SMAPE (%)
Favorita	SAE-LSTM [2]	8.180	**42.806**	8.873	45.104	7.891	44.977
	A-RNN [1]	12.185	61.116	13.027	60.591	11.793	62.130
	DA-RNN [37]	8.254	45.943	9.352	44.864	8.006	46.608
	LSTNet [28]	8.153	45.968	8.845	44.720	7.913	46.085
	TADA (w/o retrain)	**7.845**	43.622	9.265	44.128	8.282	47.046
	TADA	**7.845**	43.622	8.839	42.168	7.621	44.466
	TADA$^+$	**7.845**	43.622	**8.418**	**40.788**	**7.448**	**44.100**
OSW	SAE-LSTM [2]	21.920	49.433	21.584	48.365	21.988	49.473
	A-RNN [1]	21.983	49.027	21.969	47.724	22.262	48.753
	DA-RNN [37]	22.759	51.942	21.956	50.110	22.117	49.303
	LSTNet [28]	21.604	48.952	21.099	49.498	21.076	48.599
	TADA (w/o retrain)	**21.196**	**48.802**	21.643	47.774	21.892	49.735
	TADA	**21.196**	**48.802**	21.290	49.393	21.247	48.894
	TADA$^+$	**21.196**	**48.802**	**20.546**	**46.762**	**20.624**	**48.260**

Numbers in boldface are the best results within each column

the model to precisely capture the future effects of unknown influential factors, impeding the accuracy of predicted sales. In contrast, OSW records sales volume in a weekly granularity, which tends to be more stabilized in terms of uncertainties, and the trend alignment scheme can assist TADA to gather more seasonal information to ensure prediction effectiveness.

5.6 Discussion on online sales prediction effectiveness (RQ3)

To test the prediction performance of different models under the online setting, we constructed three test sets from the original data. The split strategy for training and evaluation are described in Table 4. Note that we choose $\Delta = 8$ for this task.

The online sales prediction results are presented in Table 5. Because both RF and XGBoost have underperformed with a significant margin compared with deep neural network-based approaches in the offline test, we will not include RF and XGBoost in the online test for brevity. It is worth mentioning that in the online test, we utilize the random sampling-based

reservoir as shown in Algorithm 2 to update the parameters in the peer models (i.e. SAE-LSTM, A-RNN, DA-RNN, LSTNet) as well as TADA. We use **TADA(random)** to denote the retrained TADA model with randomly sampled reservoir and use **TADA(w/o retrain)** to represent the static version without retraining. To validate the effectiveness of our proposed similarity-based reservoir in TADA$^+$, we also implement an online version of TADA with the randomly sampled reservoir. Based on the online sales prediction results, we can draw the following observations.

Obviously, the online sales prediction results show the dominating performance of TADA$^+$ when confronted with new data streams. Apparently, all models are inevitably affected by a slight performance drop as the input time series length for the encoder is shorter than the input used in Sect. 5.4, thus offering less available contextual information for the decoder. Surprisingly, we find that TADA can still outperform several baselines even it is not retrained with the incoming data samples. The online sales prediction performance further verifies that TADA can fully utilize various information sources to ensure the accuracy of sales prediction.

5.7 Discussion on model components (RQ4)

We implement three variants of TADA, namely TADA-SE, TADA-SA$_1$ and TADA-SA$_2$, by removing one of the key components each time. With the degraded versions of TADA, we carry out the ablation study on the performance gain from every proposed component within TADA. As shown in Table 3, the evaluation results on two real datasets indicate that these variants suffer from noticeable drops in the prediction performance. Specifically, TADA-SE shows more obvious infection. This provides evidence for our assertion that by dividing the influential factors in sales time series into semantically different internal and external features, the multitask encoder structure can extract more latent contextual information related to the real sales volume. In TADA-SE, the dynamic interaction of internal and external features is no longer modelled, causing insufficient performance accuracy.

According to Table 3, when we remove each one of the two proposed attention mechanisms in TADA-SA$_1$ and TADA-SA$_2$, the prediction performance both drops. Combining their performance on both datasets, the performance reduction is similar when either part of the dual-attention mechanism is blocked. So, we draw the observation that both attentions contribute positively and almost equally, and they are indispensable to each other for precise sales prediction. Thus, after the contextual vectors are learned from the encoder, it is crucial to leverage the dual-attention decoder to mimic the contextual information in the future as well as aligning the upcoming trend with historical ones to enhance the prediction of sales. Furthermore, as the attention mechanism provides TADA (full version) with better interpretability, we visualize the intermediate results of aligned trends in the predicting (decoding) stage, along with the predicted sales. Fig. 6 visualizes the results of trend alignment from samples selected from both Favorita and OSW datasets by highlighting the sales trend with the highest attention weight. As a result, we find that similar sales contexts lead to similar sales volume, which confirms the rationale of performing trend alignment for sales prediction and the effectiveness of all components in TADA.

From Table 5, we can notice that the similarity-based reservoir in TADA$^+$ helps the model achieve constant and significant improvement over TADA with the random sampling-based reservoir under the online setting. Also, by updating the model with the random sampling-based reservoir, the offline model TADA yields slightly better results compared with TADA (w/o retrain). The results have demonstrated: (1) updating the model parameters is essential

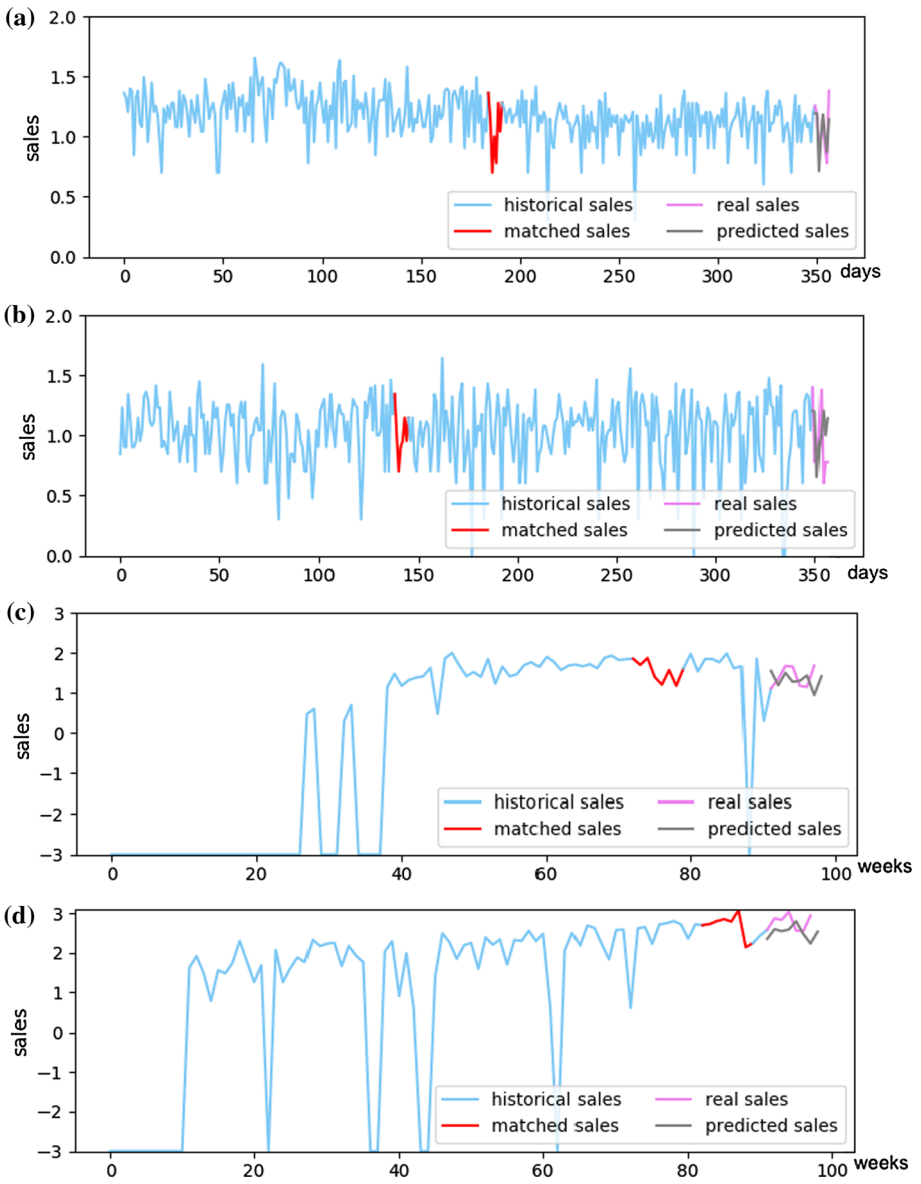


Fig. 6 Demonstration of the proposed trend alignment scheme in TADA with attention mechanism. Among these four visualizations, **a** and **b** are selected from Favorita, while **c** and **d** are selected from OSW. The sales axis is rescaled via $\log_{10}$ transfer on each dataset for better readability. Apparently, there are no obvious recurring trends in all these sales records, but TADA successfully selects the most relevant one to assist the prediction. The figures illustrate that aligned trends in sales time series not only share similar contexts, but also have close sales volume

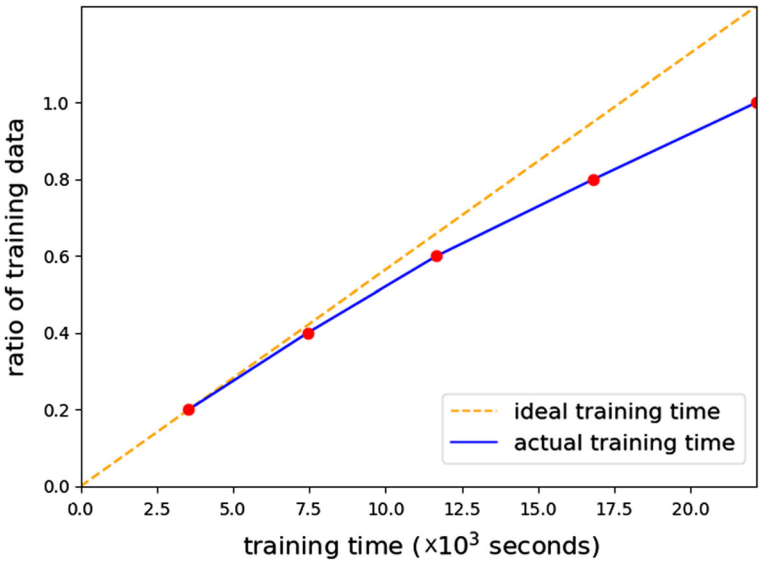


Fig. 7 Training time of TADA with varied proportions of training data

to maintain the sales prediction accuracy; (2) the dedicated similarity base reservoir devised for TADA$^+$ is proved to be effective in terms of both MAE and SMAPE.

5.8 Discussion on training efficiency and scalability (RQ5)

Due to the importance of practicality in real-life applications, we validate the scalability of the proposed model. As we proved in Sect. 4, when all the parameters in the network are fixed (in our case, the dimension for all hidden states is 128, and T is determined according to Δ), the training time for TADA and TADA$^+$ is only associated with the number of training samples. Ideally, the training time should increase linearly as we enlarge the scale of the training data. Note that we set $\Delta = 8$ ($T = 349$ correspondingly) for this validation.

We test the training efficiency and scalability by using different proportions of the whole training set from Favorita and then report the corresponding training time (excluding I/O). The test is conducted under the offline training setting with TADA. The growth of training time along with the data size is shown in Fig. 7. When the ratio of training data gradually extends from 0.2 to 1.0, the training time for TADA increases from 3.54×10^3 seconds to 22.15×10^3 seconds. It shows that the link between training time and the data scale is approximately linear. Hence, we conclude that since its linear time complexity can ensure high scalability, both TADA and TADA$^+$ can be efficiently trained with large-scale datasets.

6 Conclusion

Sales prediction is a significant yet unsolved problem due to the subtle influential patterns among different factors and the irregular sales trends triggered by complex real-life situations. We first introduce TADA in our conference version [10], a novel model that performs trend alignment with dual-attention, multitask recurrent neural networks to predict sales vol-

ume under the offline setting. The internal and external features within the influential factors are modelled in a multitask fashion, thus maintaining their unique semantic meanings when timely modelling their mutual influences to the sales. Besides, the dual-attention decoder simulates the sales contextual information in the future and then aligns the generated representation of the upcoming trend with the most relevant one from the past. In this paper, to ensure the model's practicality in real-life data streams, we further extend our model TADA into TADA$^+$, which is enhanced by an online learning module with a novel similarity-based data reservoir. Thus, TADA$^+$ can adaptively update the model parameters with both new data and the most challenging data samples from the past. In the future work of sales prediction, it will be appealing to further investigate cold-start predictions and the mutual influence between two products in the retail commerce.

Acknowledgements This work is supported by Australian Research Council (Grant Nos. DP190101985, DP170103954), The University of Queensland (Grant No. 613134) and Natural Science Foundation of China (Grant No. 6167250).

References

1. Bahdanau D, Cho K, Bengio Y (2014) Neural machine translation by jointly learning to align and translate. arXiv:1409.0473
2. Bao W, Yue J, Rao Y (2017) A deep learning framework for financial time series using stacked autoencoders and long-short term memory. PLoS ONE 12(7):e0180944
3. Bengio S, Vinyals O, Jaitly N, Shazeer N (2015) Scheduled sampling for sequence prediction with recurrent neural networks. In: NIPS, pp 1171–1179
4. Box GE, Jenkins GM, Reinsel GC, Ljung GM (2015) Time series analysis: forecasting and control. Wiley, Hoboken
5. Caballero Barajas KL, Akella R (2015) Dynamically modeling patient's health state from electronic medical records: a time series approach. In: SIGKDD, pp 69–78
6. Carbonneau R, Laframboise K, Vahidov R (2008) Application of machine learning techniques for supply chain demand forecasting. Eur J Oper Res 184(3):1140–1154
7. Chen C, Yin H, Yao J, Cui B (2013) Terec: a temporal recommender system over tweet stream. VLDB Endow 6(12):1254–1257
8. Chen H, Yin H, Chen T, Nguyen QVH, Peng WC, Li X (2019) Exploiting centrality information with graph convolutions for network representation learning. In: ICDE, pp 590–601
9. Chen T, Guestrin C (2016) Xgboost: A scalable tree boosting system. In: SIGKDD. ACM, pp 785–794
10. Chen T, Yin H, Chen H, Wu L, Wang H, Zhou X, Li X (2018) Tada: trend alignment with dual-attention multi-task recurrent neural networks for sales prediction. In: ICDM, pp 49–58
11. Chen T, Yin H, Chen H, Yan R, Nguyen QVH, Li X (2019) Air: attentional intention-aware recommender systems. In: ICDE, pp 304–315
12. Chiu B, Keogh E, Lonardi S (2003) Probabilistic discovery of time series motifs. In: SIGKDD. ACM, pp 493–498
13. Cho K, Van Merriënboer B, Bahdanau D, Bengio Y (2014) On the properties of neural machine translation: encoder–decoder approaches. arXiv:1409.1259
14. Diaz-Aviles E, Drumond L, Schmidt-Thieme L, Nejdl W (2012) Real-time top-n recommendation in social streams. In: RecSys, pp 59–66
15. Dong D, Wu H, He W, Yu D, Wang H (2015) Multi-task learning for multiple language translation. In: ACL, pp 1723–1732
16. Friedman JH (2001) Greedy function approximation: a gradient boosting machine. Ann Stat 29:1189–1232
17. Graves A, Jaitly N (2014) Towards end-to-end speech recognition with recurrent neural networks. In: ICML, pp 1764–1772
18. Guo L, Yin H, Wang Q, Chen T, Zhou A, Hung NQV (2019) Streaming session-based recommendation. In: SIGKDD, pp 1569–1577
19. Hamilton JD (1994) Time series analysis, vol 2. Princeton University Press, Princeton

20. Heigold G, Vanhoucke V, Senior A, Nguyen P, Ranzato M, Devin M, Dean J (2013) Multilingual acoustic models using distributed deep neural networks. In: ICASSP, pp 8619–8623
21. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
22. Huang JT, Li J, Yu D, Deng L, Gong Y (2013) Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers. In: ICASSP, pp 7304–7308
23. Hung NQV, Duong CT, Tam NT, Weidlich M, Aberer K, Yin H, Zhou X (2017) Argument discovery via crowdsourcing. VLDB J 26(4):511–535
24. Idé T, Kato S (2009) Travel-time prediction using Gaussian process regression: a trajectory-based approach. In: SDM. SDM, pp 1185–1196
25. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv:1412.6980
26. Kochak A, Sharma S (2015) Demand forecasting using neural network for supply chain management. Int J Mech Eng Robot Res 4(1):96–104
27. Kullback S (1997) Information theory and statistics. Courier Corporation, New York
28. Lai G, Chang WC, Yang Y, Liu H (2018) Modeling long-and short-term temporal patterns with deep neural networks. In: SIGIR, pp 95–104
29. Lawhern VJ, Solon AJ, Waytowich NR, Gordon SM, Hung CP, Lance BJ (2018) Eegnet: a compact convolutional neural network for eeg-based brain-computer interfaces. J Neural Eng 15(5):056013
30. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436
31. Liu X, Gao J, He X, Deng L, Duh K, Wang YY (2015) Representation learning using multi-task deep neural networks for semantic classification and information retrieval. In: NAACL, pp 912–921
32. Luong MT, Le QV, Sutskever I, Vinyals O, Kaiser L (2015) Multi-task sequence to sequence learning. arXiv:1511.06114
33. Mehrotra R, Awadallah AH, Shokouhi M, Yilmaz E, Zitouni I, El Kholy A, Khabsa M (2017) Deep sequential models for task satisfaction prediction. In: CIKM. ACM, pp 737–746
34. Nguyen TT, Duong CT, Weidlich M, Yin H, Nguyen QVH (2017) Retaining data from streams of social platforms with minimal regret. In: IJCAI, pp 2850–2856
35. Papadimitriou S, Sun J, Faloutsos C (2005) Streaming pattern discovery in multiple time-series. In: VLDB, pp 697–708
36. Parikh AP, Täckström O, Das D, Uszkoreit J (2016) A decomposable attention model for natural language inference. In: EMNLP, pp 2249–2255
37. Qin Y, Song D, Cheng H, Cheng W, Jiang G, Cottrell G (2017) A dual-stage attention-based recurrent neural network for time series prediction. In: IJCAI, pp 2627–2633
38. Ristanoski G, Liu W, Bailey J (2013) Time series forecasting using distribution enhanced linear regression. In: PAKDD, pp 484–495
39. Rousseeuw PJ, Leroy AM (2005) Robust regression and outlier detection, vol 589. Wiley, Hoboken
40. Shokouhi M (2011) Detecting seasonal queries by time-series analysis. In: SIGIR. ACM, pp 1171–1172
41. Sordoni A, Bengio Y, Vahabi H, Lioma C, Grue Simonsen J, Nie JY (2015) A hierarchical recurrent encoder–decoder for generative context-aware query suggestion. In: CIKM. ACM, pp 553–562
42. Sun K, Qian T, Yin H, Chen T, Chen Y, Chen L (2019) What can history tell us? Identifying relevant sessions for next-item recommendation. In: CIKM
43. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: NIPS, pp 3104–3112
44. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. In: NIPS, pp 5998–6008
45. Wang Q, Yin H, Hu Z, Lian D, Wang H, Huang Z (2018) Neural memory streaming recommender networks with adversarial training. In: SIGKDD, pp 2467–2475
46. Wang W, Yin H, Huang Z, Wang Q, Du X, Nguyen QVH (2018) Streaming ranking based recommender systems. In: SIGIR, pp 525–534
47. Wang Y, Yin H, Chen H, Wo T, Xu J, Zheng K (2019) Origin-destination matrix prediction via graph convolution: a new perspective of passenger demand modeling. In: SIGKDD, pp 1227–1235
48. Wilson A, Adams R (2013) Gaussian process kernels for pattern discovery and extrapolation. In: ICML, pp 1067–1075
49. Wu Y, Hernández-Lobato JM, Ghahramani Z (2013) Dynamic covariance models for multivariate financial time series. In: ICML, pp 558–566
50. Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhudinov R, Zemel R, Bengio Y (2015) Show, attend and tell: neural image caption generation with visual attention. In: ICML, pp 2048–2057
51. Yan W, Qiu H, Xue Y (2009) Gaussian process for long-term time-series forecasting. In: IJCNN. IEEE, pp 3420–3427
52. Yao D, Zhang C, Huang J, Bi J (2017) Serm: a recurrent model for next location prediction in semantic trajectories. In: CIKM, pp 2411–2414

53. Yin H, Chen H, Sun X, Wang H, Wang Y, Nguyen QVH (2017) Sptf: a scalable probabilistic tensor factorization model for semantic-aware behavior prediction. In: ICDM, pp 585–594
54. Yin H, Cui B, Zhou X, Wang W, Huang Z, Sadiq S (2016) Joint modeling of user check-in behaviors for real-time point-of-interest recommendation. TOIS 35(2):11
55. Zhang S, Yin H, Wang Q, Chen T, Chen H, Nguyen QVH (2019) Inferring substitutable products with deep network embedding. In: IJCAI-19, pp 4306–4312
56. Zhang Y, Yang Q (2017) A survey on multi-task learning. arXiv:1707.08114
57. Zheng X, Han J, Sun A (2018) A survey of location prediction on twitter. TKDE 30(9):1652–1671
58. Zhou J, Tung AK (2015) Smiler: a semi-lazy time series prediction system for sensors. In: SIGMOD. ACM, pp 1871–1886

Tong Chen is currently a computer science Ph.D. candidate at the School of Information Technology and Electrical Engineering, The University of Queensland. His research interests include data mining, machine learning and artificial intelligence. More specifically, he is currently conducting research on commercial time series analysis, sequential recommender systems and user behavior modelling.

Hongzhi Yin received the Ph.D. degree in computer science from Peking University, in 2014. He is a senior lecturer at The University of Queensland. He received the Australian Research Council Discovery Early-Career Researcher Award, in 2015. His research interests include recommendation system, user profiling, topic models, deep learning, social media mining and location-based services.

Hongxu Chen received his M.S. Degree in computer science from The university of Queensland, in 2015. Currently, he is a Ph.D. candidate in Data Science Group at the School of Information Technology and Electrical Engineering, The University of Queensland, Australia. His research interests include social network analysis, network representation learning, deep learning and recommendation systems.

Hao Wang is with Alibaba AI Labs. Before that, he was a chief data scientist in Qihoo 360 Inc and a professor in Chinese Academy of Sciences. He received his Ph.D. degree from University of Tokyo. His research interests include user behavior analysis and intelligent recommendation.

Xiaofang Zhou is a professor of computer science at The University of Queensland. He is the head of the Data and Knowledge Engineering Research Division, School of Information Technology and Electrical Engineering. He is the director of the ARC Research Network in Enterprise Information Infrastructure (EII), and a chief investigator of the ARC Centre of Excellence in Bioinformatics. He has been a fellow of the IEEE since 2017.

Xue Li is with Neusoft. Before that, he was a professor in School of Information Technology and Electrical Engineering at The University of Queensland, Brisbane. He received his Ph.D. degree in information systems from Queensland University of Technology, Brisbane, in 1997. His research interests include data mining and knowledge discovery from databases.

Affiliations

Tong Chen[1] · Hongzhi Yin[1] · Hongxu Chen[1] · Hao Wang[2] · Xiaofang Zhou[1] · Xue Li[1,3]

✉ Hongzhi Yin
h.yin1@uq.edu.au

Tong Chen
tong.chen@uq.edu.au

Hongxu Chen
hongxu.chen@uq.edu.au

Hao Wang
cashenry@126.com

Xiaofang Zhou
zxf@itee.uq.edu.au

Xue Li
xueli@itee.uq.edu.au

1 The University of Queensland, Brisbane, Australia

2 Alibaba AI Labs, Beijing, China

3 Dalian Neusoft University of Information, Dalian, China