Information Sciences 581 (2021) 60–72

Contents lists available at ScienceDirect

Information Sciences

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

Context-aware seq2seq translation model for sequential recommendation

Ke Sun [a], Tieyun Qian [a,*], Xu Chen [a], Ming Zhong [a]

[a] School of Computer Science, Wuhan University, Hubei, China

ARTICLE INFO

Article history:
Received 19 November 2020
Received in revised form 31 August 2021
Accepted 4 September 2021
Available online 08 September 2021

Keywords:
Sequential recommendation
Context information
Seq2seq translation model

ABSTRACT

Context information, such as product category, plays a vital role in sequential recommendations. Recently, there has been a growing interest in context-aware sequential recommender systems. However, in previous studies, contexts have often been treated as auxiliary information without the consideration of the *inter-sequence dependency between the item sequence and the context sequence*. Such a dependency provides valuable details for predicting a user's future behavior. For example, a user may buy electronic accessories after buying an electronic product.

In this paper, we propose a context-aware seq2seq translation model to capture the inter-sequence dependency for sequential recommendations. The key component in our model is a tripled seq2seq translation architecture with an injected variational autoencoder (VAE). The tripled architecture, consisting of forward and backward translation, naturally encodes bi-directional inter-sequence dependency. Moreover, the injected VAE enables the translation process to redress the semantic imbalance between context and item. We conduct extensive experiments on four real-world datasets. The results show the superior performance of our model over the state-of-the-art baselines. The code and datasets are available at https://github.com/NLPWM-WHU/CAST.

1. Introduction

The core concept of sequential recommendation is the capture of the latent transition patterns in users' historical records. Markov chains were initially employed to build a transition matrix for sequential recommendations [24]. Recently, owing to the prevalence of deep learning techniques, many deep neural networks, such as recurrent neural networks (RNNs), convolutional neural networks (CNNs), and attention mechanisms, have been incorporated into sequential recommender systems [33,40,30,10]. However, these methods focus on item sequences without considering the rich context information.

The context information can provide new perspectives for understanding users' intrinsic intentions, and it has proved to be essential for sequential recommendations. Contexts can be viewed from the perspectives of both users and items. The user's context mainly consists of user profiles or situation information, such as age or profession, and user actions, such as clicks [38,44,12]. It is usually difficult to access user profiles because of privacy protection issues; thus, researchers pay more attention to users' situations, such as position and time [2,15,13,39], and item contexts [38,9,2], such as category, brand, and descriptions. Category and time are two widely used contexts [5,1,8,13,39], as they are easily accessible from

* Corresponding author.
E-mail address: qty@whu.edu.cn (T. Qian).

https://doi.org/10.1016/j.ins.2021.09.001
0020-0255/

most real-world e-commerce websites or online social networks. We use category as an item's context and time as a user's context in this study.

Previous studies conducted to model contexts for sequential recommendations have two limitations. First, most of the existing methods [17,18,38,1,44] focus on the single item sequence, S_i, and ignore the latent transition patterns in contexts, that is, they do not treat contexts as a sequence. Second, although in two recent studies [12,23], the context sequence, S_c, was utilized, contexts were regarded as auxiliary information to better predict the next item, that is, only the $S_c \rightarrow S_i$ relation was considered and the $S_i \rightarrow S_c$ relation was ignored.

In this study, we argue that *the inter-sequence dependency, which consists of* $S_c \rightarrow S_i$ *and* $S_i \rightarrow S_c$ *relations* provides valuable details for predicting users' future intentions. In particular, it would be helpful to exploit the $S_i \rightarrow S_c$ relation. This is because users often have a pre-determined target category before they begin shopping or visiting, indicating that predicting the category will help to predict the items. We present an example of an item sequence, S_i, a context sequence, S_c, and the inter-sequence dependency between S_i and S_c in Fig. 1. A user purchases a *phone case* once after buying a *cellphone*, and there is an item-level transition from *cellphone* to *cellphone case*. Such a pattern might not be very informative when the user buys a *pad*. However, if we can predict the next category in advance by learning the category-level transition from the *electronic product* to *electronic accessories*, the system will recommend *pad case* to the user.

To capture the inter-sequence dependency between S_i and S_c, we propose a context-aware seq2seq translation (CAST) model for sequential recommendation. Our idea is inspired by recent advancements in neural machine translation (NMT). Intuitively, we can treat the item sequence and context sequence as two sentences in different languages and model their relations in a translational manner. However, there are two significant gaps between the scenarios in NMT and those in the sequential recommendation. First, we need to capture two-way relations, but a vanilla NMT contains only one way translation. Second, in NMT, each symbol (word or phrase) from the source corpus is semantically parallel with one corresponding symbol in the target corpus. In our case, a category, such as "fruit," often contains many items, such as "apple and orange," showing a subsidiary relationship.

To address the aforementioned challenges, we first propose a *a tripled seq2seq translation (TST) module* to capture the bi-directional inter-sequence dependency, including a forward translation for $S_i \rightarrow S_c$ and a backward translation for $S_c \rightarrow S_i$. We then enhance the basic seq2seq NMT model with *an injected variational autoencoder (VAE)*, such that the semantics of S_i are broadened and match those of S_c in the $S_i \rightarrow S_c$ relation extraction. Our framework also contains a personal fusion and prediction (PFP) module to combine users' static preferences with the dynamic preference encoded in the TST module. We conduct extensive experiments on four public datasets to validate the effectiveness of the proposed model. The results show that our model outperforms the state-of-the-art sequential recommendation methods.

2. Related work

2.1. Sequential recommendation without considering contexts

Classic sequential recommenders often adopt a Markov chain to model a user's sequential behaviors [24]. Owing to page limitations, we omit the research on this topic and pay more attention to the deep-learning-based techniques that have shown improvements over traditional methods. The RNN was first employed in this field [6,40], with the goal of mining dynamic sequential patterns. Over the last few years, with the success of attention and self-attention mechanisms, researchers have developed many attention-based methods, such as NARM [14], SASRec [10], and ATTRec [43]. These methods can leverage the user's entire history and usually achieve better performance than RNNs.

More recently, the language model named Bert has been employed in sequential recommendation [26], and it has shown competitive performance. Other neural networks, such as memory networks [3,29], gating networks [21], CNNs [30,42,4], non-local neural networks [27], and graph neural networks [22,35,36], have also been explored in the literature. In general, these deep learning-based methods mainly concentrate on modeling item sequences without considering context information.

2.2. Context-aware sequential recommendation

In addition to the interaction (item) sequence, researchers are now paying more attention to the additional context information to achieve more accurate recommendations. However, most existing context-aware sequential recommendation (CASR) methods do not treat the context as a sequence. For clarity, we divided these methods into two categories according to the context type.

The first type focuses on the user's context, such as location, time, and weather. Among these contexts, time is the most widely used, and it is mainly adopted to adjust the influence of past interaction records [17,41,34,31,19,32]. For example, a larger time interval usually represents less influence from history. Some recent methods [39,13] aim to mine implicit temporal patterns from the user's context. Other uncommon contexts, such as text [46] and social network [47], are also proved to be helpful for recommendation.

The second type deploys the items' context, such as category, brand, and shop. The attention mechanism is commonly used to handle the context of items. For example, Bai et al. [1] propose a hierarchical attention architecture for next-

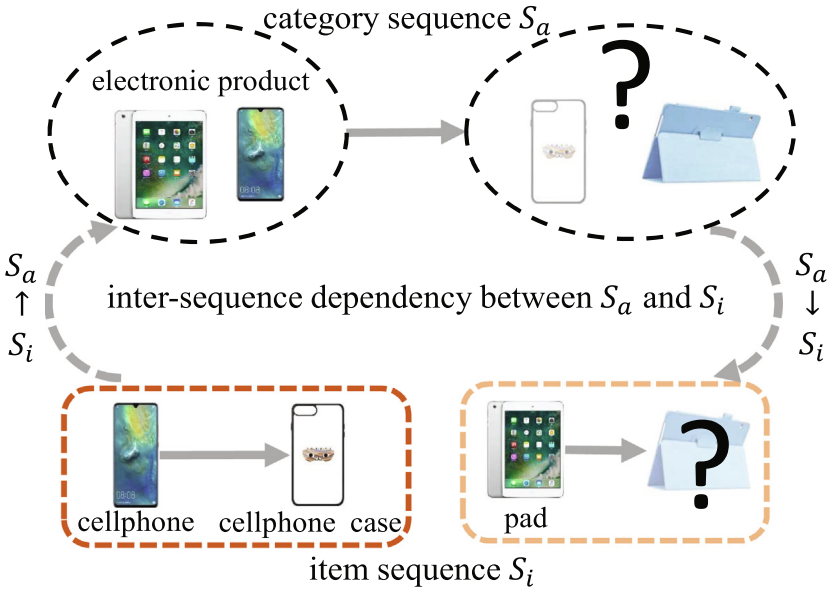

Fig. 1. Item sequence, S_i, category sequence, S_a, and the inter-sequence dependency between S_i and S_a.

basket recommendation, and three attention-based frameworks are proposed by Zhou et al. [44], Huang et al. [9], and Zhou et al. [45] to model various context information. Other architectures or techniques [8,5,38,15], such as RNN, reasoning networks, and listwise ranking models, have also been introduced to model the context of the items in CASR.

It can be seen that great effort has been made to incorporate contextual information to achieve better performance in sequential recommendation. However, the aforementioned approaches are all based on a single-item sequence without modeling context as a sequence. Recently, two multi-sequence-based context-aware methods have been proposed for sequential recommendations. Le et al. [12] developed a contemporaneous basket sequence (CBS) framework with twin network structures to capture the synergies between the support and target sequences. Rakkappan and Rajan [23] proposed a stacked temporal-context aware RNN (STAR) method to make use of stacked RNNs to model item and context sequences, respectively. Both CBS and STAR view the context sequence as auxiliary information. They explored only the context-to-item relationship, and ignored the item-to-context relation. In contrast, our proposed model can capture two-way relations with the tripled seq2seq translation architecture, and thus achieves significantly better performance.

3. Problem formulation and preliminaries

Problem Formulation. Let $U = \{u_1, u_2, \ldots, u_{|U|}\}$ denote a set of users, and let $I = \{i_1, i_2, \ldots, i_{|I|}\}$ denote a set of items. For the CASR problem, we have a set of contexts C denoting user or item contexts, such as time, weather, and category. Each dataset contains its own context. In this study, we choose time and category (shop) as the contexts, because they are easily accessible and have been widely used in the literature. Specifically, each item i belongs to a category context A, and the user interacts with the item at a specific time τ. Note that τ is mapped to a 24-h categorical feature $t_\tau \in T$, denoting the corresponding time context. Hence, we have the context set as $C = \{A, T\}$. For each user $u \in U$, we can obtain an interaction sequence (often called an item sequence) sorted by the absolute time $S_i = \{i_1, i_2, \ldots, i_n\}$ and two types of context sequences: $S_a = \{a_1, a_2, \ldots, a_n\}$ $(a_i \in A)$, and $S_t = \{t_1, t_2, \ldots, t_n\}$ $(t_i \in T)$. We use a general symbol S_c to denote S_a and S_t.

Given the historical item sequence S_i of user u, and two context sequences, S_a and S_t, our goal is to predict the next item i_{n+1} that u is most likely to interact with, at time-step t_{n+1}.

Preliminaries. In recent years, seq2seq NMT [28,20] has achieved great success at translating texts from the source language into those in the target language. NMT models mainly contain three components: an encoder for source sentence reading, a decoder for target sentence generation, and middleware for relation modeling.

It can be seen that NMT methods not only model sequential patterns in the two corpora, they also exploit the relations between these corpora. This is inherently similar to our problem, where we treat context and item sequences as sentences from different corpora. Intuitively, the context sequence can be directly translated into an item sequence for modeling the $S_c \rightarrow S_i$ relation. However, the context in such a translation model cannot be informed of any item information, that is, $S_i \rightarrow S_c$ dependency is ignored. Thus, we further employ an item sequence layer at the bottom, and translate the item into context. This helps generate a better context representation before it is translated into an item. Nevertheless, because of the *semantic imbalance problem* between context and item, that is, context is an abstract concept and it may contain many items, it is dif-

ficult to directly translate the low-level item sequence S_i into a high-level context sequence S_c. To tackle this problem, we propose injecting a VAE [11] into the vanilla seq2seq NMT.

VAE enables the model to learn a compressed representation $\boldsymbol{z}$ from the input picture or sentence [11]. The conventional loss function of the VAE [11] is formulated as

$$\mathscr{L}_{vae} = D_{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) - \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log(p(\boldsymbol{x}|\boldsymbol{z}))], \tag{1}$$

where $\boldsymbol{x}$ is the observed input data, $\boldsymbol{z}$ is the latent factor variable, $q(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{x}|\boldsymbol{z})$ are the inference and generative models, respectively, and $D_{KL}(q||p)$ is the Kullback-Leibler divergence between the two distributions. In Eq. 1, the expectation $\mathbb{E}$ can be viewed as the reconstruction loss, and KL as regularization.

While existing studies adopt VAE to reflect the distribution in a single sequence [25], we explore its capability to address the semantic imbalance problem between the item and the context. Specifically, the VAE can generate the hidden representation of the next item from a distribution instead of directly taking the output item embedded from the bottom S_i layer as input. In this way, the context sequence layer receives different item representations when $S_i \rightarrow S_c$ translation, and the semantic imbalance problem is solved accordingly.

4. The proposed method

In this section, we elaborate on our context-aware seq2seq translation (CAST) model.

4.1. Overall architecture

The proposed CAST model consists of two modules. One is the tripled seq2seq translation (TST) module. The other is the PFP module. The overall architecture of CAST is shown in Fig. 2.

The TST module is the main module responsible for modeling the bi-directional inter-sequence dependency between one type of context sequence and the interaction sequence, where TSTa and TSTt denote the category and time context, respectively. Because we need to stack the TST module for multiple context sequences, it matters which context sequence is first translated into the item sequence. Correspondingly, our CAST model has two variants, CASTat and CASTta, where *at* and *ta* denote the category-time and time-category stacking order, respectively. In the experiment, we show that the varying scenarios in the datasets require different stacking orders.

4.2. Tripled seq2seq translation module

The TST module consists of three layers. The first layer $L^{(1)}$ encodes the item sequence S_i, then the second layer $L^{(2)}$ forward translates S_i into context sequence S_c, and the third layer $L^{(3)}$ back translates S_c into S_i. Note that we use the superscript number to denote the layer, and use subscripts i, c, and k for the item, context, and time-step throughout the paper.

First Layer of TST: We feed the item embedding sequence $(\boldsymbol{i_1}, \boldsymbol{i_2}, \ldots, \boldsymbol{i_n})$ to $L^{(1)}$, and output the hidden state $\boldsymbol{h}^{(\boldsymbol{1})}_{i_k} \in \mathbb{R}^{d\times 1}$ for each item i_k at the k^{th} time step:

$$\boldsymbol{h}^{(\boldsymbol{1})}_{i_k} = \mathrm{RNN}(\boldsymbol{h}^{(\boldsymbol{1})}_{i_{k-1}}, \boldsymbol{i_k}) \tag{2}$$

Next, to address the aforementioned semantic imbalance problem, we introduce an injected VAE structure between S_i in $L^{(1)}$ and S_c in $L^{(2)}$. Specifically, at each time step, we infer a latent variable $\boldsymbol{z_k} \in \mathbb{R}^{d/2\times 1}$ that follows the Gaussian distribution, and is conditioned on the previous actions of the sequence:

$$q(\boldsymbol{z_k}|\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}) = \mathscr{N}\left(\mu(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}), \mathrm{diag}\left\{\sigma^2(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k})\right\}\right), \tag{3}$$

where $\mu(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k})$ and $\sigma^2(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k})$ denote the parameters of a Gaussian distribution generated from $\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}$, and can be simply defined as

$$\mu\left(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}\right) = \boldsymbol{W}^{(\boldsymbol{1})i_k}_{\boldsymbol{1h}}, \quad \sigma^2\left(\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}\right) = \exp^{\boldsymbol{W}^{(\boldsymbol{1})i_k}_{\boldsymbol{2h}}}, \tag{4}$$

where $\boldsymbol{W_1} \in \mathbb{R}^{d/2\times d}$ and $\boldsymbol{W_2} \in \mathbb{R}^{d/2\times d}$ are the two trainable weight matrices. Next, the latent factor $\boldsymbol{z_k}$ is sampled from the above posterior inference distribution with a widely used reparameterization trick to avoid a non-differential problem [11]. Such a procedure enhances the representation of item i_k from the fixed $\boldsymbol{h}^{(\boldsymbol{1})}_{i_k}$ to a varying $\boldsymbol{z_k}$, and the semantics of the item are broadened accordingly. Furthermore, $\boldsymbol{z_k}$ is employed in the loss function of $L^{(1)}$ to maintain its capability for predicting the next item:

$$\mathscr{L}^{(1)}_{TSTc} = -\log(p(i_{k+1}|\boldsymbol{z_k})) \tag{5}$$

where TSTc ($c \in \{a, t\}$) can be either TSTa or TSTt, depending on the context of the category or time.

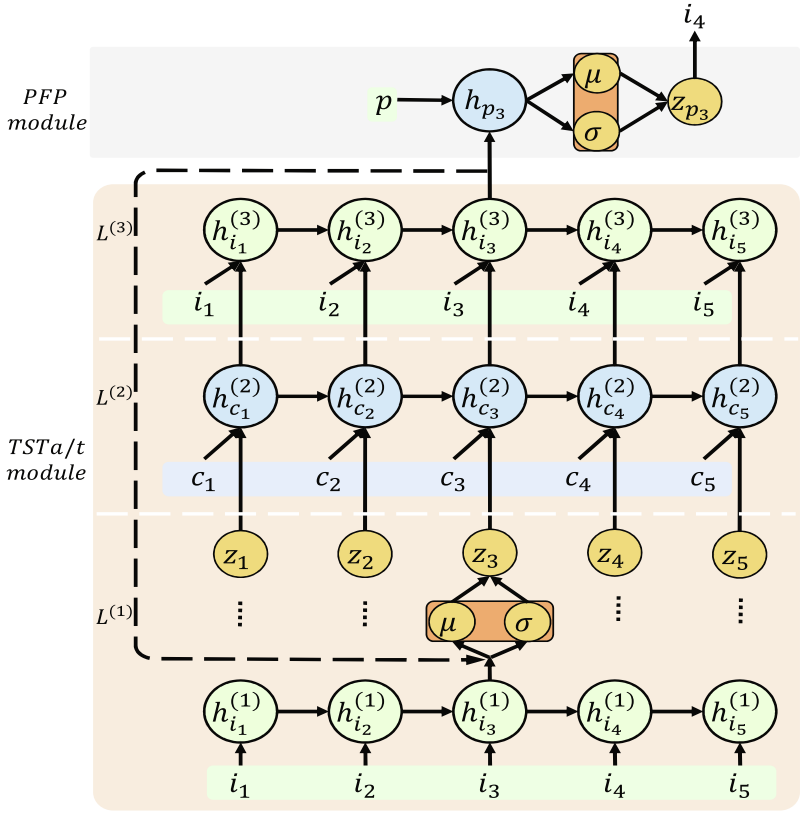

Fig. 2. Architecture of our CAST model.

Second Layer of TST: We forward translate item sequence S_i into context sequence S_c in $L^{(2)}$. First, we incorporate $\boldsymbol{z_k}$ into the generation of the hidden context state $\boldsymbol{h}_{\boldsymbol{c_k}}^{(\boldsymbol{2})} \in \mathbb{R}^{d\times 1}$:

$$\boldsymbol{h}_{\boldsymbol{c_k}}^{(\boldsymbol{2})} = \mathrm{RNN}(\boldsymbol{h}_{\boldsymbol{c_{k-1}}}^{(\boldsymbol{2})}, \boldsymbol{c_k}, \boldsymbol{W_3 z_k}), \tag{6}$$

where $\boldsymbol{W_3} \in \mathbb{R}^{d\times d/2}$ is a trainable weight matrix.

We then modify the standard VAE loss function to serve as the loss for the $S_i \rightarrow S_c$ translation in the second layer:

$$\mathscr{L}_{TSTc}^{(2)} = \mathscr{L}_{c}^{(2)} + KL_{c}^{(2)} \tag{7}$$

$\mathscr{L}_{c}^{(2)}$ has different settings for the two types of context sequences, S_a and S_t. When the TST module deals with the items' category sequence S_a ($c = a$), $\mathscr{L}_{c=a}^{(2)} = -\log(p(c_{k+1}|\boldsymbol{h}_{c_k}^{(\boldsymbol{2})}))$ is used for predicting the item's next context (e.g., category and shop). In addition, $\mathscr{L}_{c=t}^{(2)}$ is set to zero for the user context sequence S_t (e.g., time). This is because the model is already informed of the user's specific circumstance information at the $k+1^{th}$ time step. The expression $KL_c^{(2)} = D_{KL}(q(\boldsymbol{z_k}|\boldsymbol{h}_{i_k}^{(\boldsymbol{1})})||p(\boldsymbol{z_k}))$ is the Kullback-Leibler (KL) divergence regularization term from the VAE loss function, where $q()$ is defined in Eq. 3, and $p()$ is the prior over the latent variable $\boldsymbol{z_k}$, which is commonly set to a Gaussian distribution.

Third Layer of TST: After obtaining the hidden context state $\boldsymbol{h}_{\boldsymbol{c_k}}^{(\boldsymbol{2})}$ in $L^{(2)}$, we can back translate S_c into S_i in the third layer. This is done by feeding $\boldsymbol{h}_{\boldsymbol{c_k}}^{(\boldsymbol{2})}$ to $L^{(3)}$ to generate the final hidden item state $\boldsymbol{h}_{\boldsymbol{i_k}}^{(\boldsymbol{3})}$:

$$\boldsymbol{h}_{\boldsymbol{i_k}}^{(\boldsymbol{3})} = \mathrm{RNN}(\boldsymbol{h}_{\boldsymbol{i_{k-1}}}^{(\boldsymbol{3})}, \boldsymbol{i_k}, \boldsymbol{h}_{\boldsymbol{c_k}}^{(\boldsymbol{2})}) \tag{8}$$

Currently, a single TSTa/TSTt module is presented for handling one category or time context sequence. TST captures the bidirectional inter-sequence dependency via the $S_i \rightarrow S_c \rightarrow S_i$ translation procedure. With the back-translated S_i as the bridge, the TST modules can be stacked to exploit multiple context sequences. Assuming that we first translate S_a, the output $\boldsymbol{h}_{\boldsymbol{i_k}}^{(\boldsymbol{3})}$ from TSTa is directly used as the input of the VAE in $L^{(1)}$ of TSTt, as shown in Fig. 2). Such a stacking order results in the CASTat model with the $S_i \rightarrow S_a \rightarrow S_i \rightarrow S_t \rightarrow S_i$ translation procedure. We can obtain the CASTta model in a similar way.

4.3. Personal fusion and prediction module

The stacked TST modules incorporate multifaceted context sequences with the interaction sequence, and thus capture the user's dynamic sequential patterns well. Note that a user's static preference is also important for the prediction of the next item. For example, different users may visit different POIs, even if they arrive at the same location. Therefore, we further propose combining users' static preferences in PFP.

For each user $u \in U$, we allocate a randomly initialized personal vector $\boldsymbol{u} \in \mathbb{R}^{d\times 1}$ to reflect the general preference of u. Then, we produce a fusion vector $\boldsymbol{h}_{\boldsymbol{p}_k} \in \mathbb{R}^{2d\times 1}$ that concatenates the user's dynamic preference $\boldsymbol{h}_{\boldsymbol{i}_k}^{(3)}$ from the TST and the user's static preference $\boldsymbol{u}$:

$$\boldsymbol{h}_{\boldsymbol{p}_k} = \boldsymbol{u} \oplus \boldsymbol{h}_{\boldsymbol{i}_k}^{(3)} \tag{9}$$

During the fusing process, if we simply make use of $\boldsymbol{h}_{\boldsymbol{p}_k}$, the model will gradually pay more attention to the user's personal preference $\boldsymbol{u}$, and ignore the impacts of dynamic preference $\boldsymbol{h}_{\boldsymbol{i}_k}$, as $\boldsymbol{u}$ is at the top layer of the model, and it is easier to back-propagate the gradient. To address this problem, we employ a VAE again to generate the final representation $\boldsymbol{z}_{\boldsymbol{p}_k} \in \mathbb{R}^{d\times 1}$ from the approximate posterior $q(\boldsymbol{z}_{\boldsymbol{p}_k}|\boldsymbol{h}_{\boldsymbol{p}_k})$.

$$q(\boldsymbol{z}_{\boldsymbol{p}_k}|\boldsymbol{h}_{\boldsymbol{p}_k}) = \mathcal{N}(\mu(\boldsymbol{h}_{\boldsymbol{p}_k}), \mathrm{diag}\{\sigma^2(\boldsymbol{h}_{\boldsymbol{p}_k})\}), \tag{10}$$

$$\mu(\boldsymbol{h}_{\boldsymbol{p}_k}) = \boldsymbol{W}_{\boldsymbol{4}}\boldsymbol{h}_{\boldsymbol{p}_k}, \quad \sigma^2(\boldsymbol{h}_{\boldsymbol{p}_k}) = \exp^{\boldsymbol{W}_{\boldsymbol{5}}\boldsymbol{h}_{\boldsymbol{p}_k}}, \tag{11}$$

where $\boldsymbol{W}_{\boldsymbol{4}} \in \mathbb{R}^{d\times 2d}$ and $\boldsymbol{W}_{\boldsymbol{5}} \in \mathbb{R}^{d\times 2d}$ are two trainable weight matrices.

Finally, we design the following loss function $\mathscr{L}_{PFP}^{(f)}$ in PFP by modifying the standard VAE loss:

$$\mathscr{L}_{PFP}^{(f)} = \mathscr{L}_{i}^{(f)} + KL_{i}^{(f)}, \tag{12}$$

where $\mathscr{L}_{i}^{(f)} = -\log(p(i_{k+1}|\boldsymbol{z}_{\boldsymbol{p}_k}))$ is introduced to consider the user's static interests, and $KL_{i}^{(f)}$ is the regularization term:

$$KL_{i}^{(f)} = D_{KL}(q(\boldsymbol{z}_{\boldsymbol{p}_k}|\boldsymbol{h}_{\boldsymbol{p}_k})||p(\boldsymbol{z}_{\boldsymbol{p}_k})) \tag{13}$$

Model Training. The loss in our CAST model consists of the $\mathscr{L}_{TSTc}^{(1)}$ and $\mathscr{L}_{TSTc}^{(2)}$ losses from the TST, and the $\mathscr{L}_{PFP}^{(f)}$ loss from PFP. Moreover, the TST is stacked for two types of context sequences. Taking the CASTat model with the $S_i \rightarrow S_a \rightarrow S_i \rightarrow S_t \rightarrow S_i$ translation procedure as an example, the total loss function is defined as

$$\mathscr{L}_{CASTat} = \mathscr{L}_{TSTa}^{(1)} + \mathscr{L}_{TSTa}^{(2)} + \mathscr{L}_{TSTt}^{(1)} + \mathscr{L}_{TSTt}^{(2)} + \mathscr{L}_{PFP}^{(f)} \tag{14}$$

Furthermore, we would like to balance the *KL* term and the prediction term. As shown in [7], a high weight on the *KL* term helps the model to learn disentangled representations of independent data factors, and can improve performance. Thus, we expand $\mathscr{L}_{TSTc}^{(2)}$ and $\mathscr{L}_{PFP}^{(f)}$ using Eq. 7, 12, and then the joint training objective is defined as

$$\mathscr{L}_{CASTat} = \mathscr{L}_{TSTa}^{(1)} + \mathscr{L}_{a}^{(2)} + \mathscr{L}_{TSTt}^{(1)} + \mathscr{L}_{i}^{(f)} + \lambda\left\{KL_{a}^{(2)} + KL_{t}^{(2)} + KL_{i}^{(f)}\right\} \tag{15}$$

where λ is a hyperparameter, and we will examine its impacts in the experiment. Note that we remove $\mathscr{L}_{t}^{(2)}$ from TSTt because it is equal to zero in the context of time. To optimize all model parameters, including the trainable weight matrices, we employ the widely used Adam optimizer.

5. Experiments

5.1. Settings

Datasets and Metrics We conduct experiments on four public real-world datasets:

- A movie rating dataset **MovieLens-1 M**.[1] This is a widely used dataset for recommendation tasks, containing 1,000,209 rating records of 3,900 movies from 18 categories by 6,040 users. Each record is associated with a specific timestamp.
- A POI recommendation dataset **Gowalla**.[2] This is a public location-based social networking dataset for point-of-interest recommendation tasks collected from February 2009 to October 2010. Each record in Gowalla consists of a user ID, a POI ID, a corresponding GPS location, and a timestamp. The category information is not provided in the original dataset, and we use the version collected in a previous study [37].

[1] https://grouplens.org/datasets/movielens/1m/.
[2] https://snap.stanford.edu/data/loc-gowalla.html.

- Two e-commerce datasets **Taobao**[3] and **JD**.[4] These two large datasets are from e-commerce websites Taobao and JingDong. Each record in both datasets consists of a user ID, item ID, category ID, and a timestamp. Additional information, such as brands and shops is also provided in the JD dataset.

Due to the large scale of the Taobao and JD datasets, we randomly sampled 100,000 users so that they are computationally tractable [34]. Following the settings in prior work [30,10,21], we removed inactive users who visited less than m items, and unpopular items that interacted with fewer than m users. Furthermore, we retained users who have more than l records on Gowalla and JD to ensure the sequence length [15]. We employed the item category and the user's temporal information as the contexts. For JD, we regarded shop ID as category information. For each user, the most recently visited item was considered as the test item, while the second one was used for validation, and all other items were used for training [10]. Note that if the sequence was too long, we limited its length, such that it is longer than 95% of the users' historical sequences in the corresponding dataset. The statistics of the above four datasets after preprocessing are listed in Table 1.

For evaluation, we adopted four widely used metrics: hit ratio (*Hit@K*), normalized discounted cumulative gain (*NDCG@K*), F-measure (*F1@K*), and area under the curve (*AUC*), where K is from $\{1, 5, 10\}$ in our study. The first three metrics evaluate the top K-ranking performance, while the last one assesses the overall performance. Following previous studies [10,8], we rank the ground truth item and other 500 items randomly sampled from unvisited items by a specific user.

Baseline Methods We selected three types of state-of-the-art sequential recommendation methods as our baseline:

(1) *Baselines without considering contexts*

- **SASRec** [10]: This is the first method that employs the self-attention mechanism in the sequential recommendation problem without considering any additional context information. It has been shown to outperform non-neural network models, such as BPR, FPMC, and TransRec.
- **BERT4Rec** [26]: This method is developed from SASRec, which replaces the self-attention mechanism with the popular language model Bert by treating users' behavior sequences as sentences.

(2) *Baselines without considering contexts as sequences*:

- **ANAM** [1]: This method is proposed for next-basket recommendation, which takes the items' attribute information, such as category, into account. A hierarchical attentive RNN was proposed in ANAM to model the users' interests for items.
- **ATRank** [44]: This method aims at generating a more comprehensive user representation by considering users' heterogeneous behaviors, such as buying items, clicking ads, and searching keywords. It directly adopts an attention mechanism to combine different types of behaviors with temporal information before making a final recommendation.
- **DIEN** [45]: This model is proposed for the click-through rate (CTR) prediction task using multiple types of extra information, such as user profile and item category. It incorporates an attentional update gate into the gated recurrent unit (GRU) to capture the evolution of users' interests.
- **SliRec** [41]: This method adaptively integrates both users' long- and short-term preferences based on different types of contexts, such as time interval and category. It uses temporal information to control the state transition and other contexts to help generate user fusion preferences.
- **CTA** [34]: This model utilizes context information to control the influence from historical events with the proposed three-stage weighing pipeline. It mainly concentrates on the time interval context to adjust the importance of previous records.

(3) *Baselines that consider contexts as sequences*:

- **STAR** [23]: This model is based on the stacked RNN, which treats time and time interval as contexts. It sends the context hidden states in the lower RNN layer to the higher-item sequence modeling layer. In this experiment, we use the category to replace time and retain the time interval (which performs better) as the other context for STAR.
- **CBS** [12]: This method models a pair of target and support sequences with a twin network, and we use category or time to build the support sequence. CBS builds the relationship between different sequences by sharing the parameters of sequential modeling networks. Because CBS can model only one support sequence, we report better results between two context sequences, and we will examine the effects of using both contexts with an extended triple network.

Experimental Settings For baseline methods, we adopt the codes provided by the corresponding authors, except that we implemented CBS using PyTorch. For a fair comparison, we set the hidden latent state and embedding dimension d to 50 [10,21]. All other parameters and training settings were consistent with those reported in the original studies. Before training, we used a sliding window with a *Len* (=5) length over the user's history to partition the training sequences [30,21]. We set the batch size to 128, and the learning rate to 0.001. To avoid overfitting, we added an extra dropout layer over all embeddings, and the dropout rate was set to 0.2. The hyperparameter λ is set to 1, 20, 20, and 40 for MovieLens, Gowalla, Taobao, and JD, respectively. Note that we adopt the annealing strategy [16,25] in training; that is, we trained the model for four

[3] https://tianchi.aliyun.com/dataset/dataDetail?dataId=649.
[4] https://www.jd.com/.

Table 1
Statistics of the datasets.

Dataset	#users	#items	#samples	#cate	sparsity	m	l
MovieLens	6,040	3,416	999,611	18	95.16%	5	0
Gowalla	13,989	22,239	896,506	354	99.71%	10	20
JD	23,548	29,027	1,151,117	3461	99.83%	5	20
Taobao	41,910	37,903	2,437,886	1415	99.85%	20	0

epochs without *KL* loss, and then determined λ using the grid search strategy from $\{1, 10, 20, 30, 40\}$ on the validation set. We finally report the average scores by running the models three times.

5.2. Performance comparison

Table 2 shows the performance comparison for all methods on four datasets. We highlight the best results in each row in boldface, and underline the second-best ones. From Table 2, we have the following important observations.

First, it is clear that our model outperforms all baselines on MovieLens, Gowalla, and JD, and it achieves the best results in most cases on Taobao. It is also remarkable that CASTat performs better than CASTta on Movielens and Gowalla, yet worse on JD and Taobao. This phenomenon was caused by the properties of the dataset. JD and Taobao are from e-commerce areas, where the category/shop is more important than time in determining users' purchase behaviors.

Second, the CBS framework, which models context with a separate sequence, can produce the best results among baselines on MovieLens and Taobao, indicating that the model benefits from the exploitation of the context sequence. The inferior results for STAR can be attributed to the one-way relation modeling of $S_c \rightarrow S_i$. Another reason is that the simple combination strategy of representation multiplication might not address the semantic imbalance problem between item and context.

Third, the methods that do not consider context as a sequence, or without considering context, have their own strong points, depending on the model architecture and optimization target. For example, SASRec and BERT4Rec are strong baselines because they directly learn from historical behaviors based on the current state with the self-attention mechanism. ATRank has a noticeable *AUC* score because its binary classification loss function biases the global ranking performance.

5.3. Ablation study

We conducted two types of ablation studies. The first type aims to validate the effectiveness of the proposed tripled seq2-seq translation architecture. To this end, we implement the following variants of the three architectures:

(a) $S_i \overset{vae}{\rightarrow} S_c \rightarrow S_i \overset{vae}{\rightarrow} S_c \rightarrow S_i$ (**translation architecture**) is a variant of our CAST without PFP, where the stacking order can be S_aS_t or S_tS_a.
(b) $S_i||S_a||S_t$ (**parallel architecture**) extends the twin network in CBS [12] to a triple network to include two types of contexts.
(c) $(S_a||S_t) \rightarrow S_i$ (**stacking architecture**) adopts an architecture that is similar to that of STAR [23], which first models context sequences independently, integrates them in the final item sequence layer, and replaces RNN in STAR with long short-term memory (LSTM) for a fair comparison with our method.

Note that all of these architectures are conducted under the same setting, where the users' historical data are partitioned with a sliding window. The results of the architecture validation are shown in Table 3. It is clear that our translation architecture yields the best *HNF@1* scores (@10 scores are also the best in most cases) among the three architectures. This proves that modeling both the $S_i \rightarrow S_a$ and $S_a \rightarrow S_i$ relations helps to make precise recommendations. Although our architecture cannot always produce the best *AUC* scores in several cases, it is competitive with parallel and stacking architectures. These results demonstrate the effectiveness of our translation architecture. Indeed, *@K* metrics are more important in sequential recommendation, whereas *AUC* is more suitable for tasks such as CTR prediction. Another interesting phenomenon is that the parallel architecture that considers two types of contexts sometimes performs worse than the one-context model CBS. This is because of the different experimental settings. CBS considers users' whole sequences, leading to capturing the long-term preferences. However, in this ablation experiment, we divided the whole sequences into shorter training samples, where long-term information was missing.

The second type of ablation study aims to examine the impacts of different components, and we introduce four simplified models:

(a) **-TSTa:** We remove the TSTa and retain TSTt, so that only the temporal context is utilized.
(b) **-TSTt:** We remove the TSTt and retain TSTa, so that only the category context is utilized.
(c) **-VAE:** We remove all VAEs to examine their impacts.

Table 2
Performance comparison on four datasets. Note that HNF@1 denotes Hit@1, NDCG@1, and F1@1.

Dataset	Metrics	SASRec	BERT4Rec	ANAM	ATRank	DIEN	SliRec	CTA	STAR	CBS	CASTat	CASTta
MovieLens	HNF@1	0.1635	0.1800	0.0614	0.0977	0.0715	0.0901	0.1679	0.1603	0.2087	**0.2300**	0.2292
	Hit@5	0.4422	0.4429	0.2013	0.2624	0.2278	0.2755	0.4267	0.3803	0.4579	**0.4922**	0.4918
	Hit@10	0.5826	0.5758	0.3152	0.3854	0.3373	0.4051	0.5553	0.4891	0.5763	**0.6104**	0.6065
	NDCG@5	0.3064	0.3160	0.1318	0.1810	0.1504	0.1834	0.3026	0.2729	0.3376	**0.3664**	0.3658
	NDCG@10	0.3520	0.3591	0.1685	0.2239	0.1855	0.2252	0.3441	0.3082	0.3760	**0.4048**	0.4031
	F1@5	0.1474	0.1476	0.0671	0.0875	0.0759	0.0918	0.1422	0.1268	0.1526	**0.1641**	0.1639
	F1@10	0.1059	0.1047	0.0573	0.0719	0.0613	0.0737	0.1010	0.0889	0.1048	**0.1111**	0.1103
	AUC	0.9381	0.9247	0.8921	0.9097	0.8405	0.8714	0.9213	0.9010	0.9253	**0.9382**	0.9375
Gowalla	HNF@1	0.4270	0.4491	0.4600	0.3100	0.2223	0.2634	0.2739	0.2716	0.4118	**0.5289**	0.5193
	Hit@5	0.7035	0.6879	0.6348	0.6274	0.4498	0.5453	0.4267	0.4927	0.6675	**0.7555**	0.7461
	Hit@10	0.8036	0.7800	0.7181	0.7661	0.5746	0.6865	0.5027	0.5992	0.7659	**0.8365**	0.8317
	NDCG@5	0.5732	0.5761	0.5513	0.4762	0.3399	0.4093	0.3551	0.3868	0.5476	**0.6493**	0.6401
	NDCG@10	0.6058	0.6059	0.5782	0.5213	0.3803	0.4550	0.3793	0.4212	0.5794	**0.6756**	0.6679
	F1@5	0.2345	0.2293	0.2116	0.2091	0.1699	0.1818	0.1426	0.1642	0.2225	**0.2518**	0.2487
	F1@10	0.1461	0.1418	0.1306	0.1393	0.1045	0.1248	0.0914	0.1089	0.1393	**0.1521**	0.1512
	AUC	0.9756	0.9741	0.9604	0.9798	0.9532	0.9718	0.8733	0.9394	0.9734	**0.9816**	0.9812
JD	HNF@1	0.4842	0.5074	0.5229	0.3106	0.3889	0.4118	0.4419	0.3639	0.5008	0.5294	**0.5298**
	Hit@5	0.6794	0.6692	0.6042	0.6015	0.5689	0.6486	0.5727	0.5181	0.6623	0.7132	**0.7142**
	Hit@10	0.7554	0.7311	0.6477	0.7122	0.6288	0.7208	0.6277	0.5874	0.7254	0.7787	**0.7790**
	NDCG@5	0.5882	0.5938	0.5648	0.4649	0.4848	0.5389	0.5111	0.4458	0.5873	0.6282	**0.6286**
	NDCG@10	0.6128	0.6138	0.5789	0.5008	0.5042	0.5624	0.5289	0.4681	0.6078	0.6495	**0.6497**
	F1@5	0.2265	0.2231	0.2014	0.2005	0.1896	0.2162	0.1909	0.1727	0.2208	0.2377	**0.2381**
	F1@10	0.1373	0.1329	0.1178	0.1295	0.1143	0.1311	0.1141	0.1068	0.1319	**0.1416**	**0.1416**
	AUC	0.9602	0.9512	0.9086	0.9598	0.9127	0.9465	0.8969	0.9074	0.9471	**0.9605**	0.9604
Taobao	HNF@1	0.2206	0.2593	0.2431	0.1978	0.2214	0.2518	**0.2911**	0.1776	0.2593	0.2792	0.2812
	Hit@5	0.4152	0.4421	0.3572	0.4275	0.4129	0.4669	0.4472	0.3054	0.4408	0.4778	**0.4821**
	Hit@10	0.5141	0.5275	0.4277	0.5393	0.5045	0.5613	0.5163	0.3701	0.5239	0.5656	**0.5717**
	NDCG@5	0.3218	0.3552	0.3020	0.3173	0.3216	0.3648	0.3735	0.2446	0.3548	0.3838	**0.3871**
	NDCG@10	0.3538	0.3828	0.3247	0.3535	0.3512	0.3954	0.3958	0.2655	0.3817	0.4122	**0.4162**
	F1@5	0.1384	0.1474	0.1191	0.1425	0.1376	0.1556	0.1491	0.2766	0.1469	0.1593	**0.1607**
	F1@10	0.0935	0.0959	0.0778	0.0980	0.0917	0.1020	0.0939	0.1018	0.0953	0.1028	**0.1040**
	AUC	0.9071	0.9021	0.8637	**0.9183**	0.8927	0.9149	0.8662	0.8131	0.8969	0.9102	0.9110

(d) **-PFP:** We remove the PFP module to examine whether personal information is beneficial to CAST. Note that the variant without PFP is the same as the translation architecture described in the previous subsection.

The results of the component validation are listed in Table 4. The *Original* row presents better scores between CASTat and CASTta. As expected, our complete CAST outperformed the other simplified variants in almost all cases. From the results in -TSTa and -TSTt, we find that removing one type of context will harm the performance, indicating the necessity of exploring multifaceted contexts. In addition, the removal of VAE and PFP showed a significant drop in performance. This further proves that the VAE plays a critical role in our model for handling the semantic imbalance problem, and our PFP module is effective for fusing the user's static and dynamic preferences.

5.4. Parameter study

In this section, we examine the impacts of two hyperparameters: the length of the sliding window *Len*, and the balance factor λ. We report the *AUC* and *HNF@1* results in Fig. 3, 4 by varying *Len* in $\{3, 4, 5, 6, 7\}$ and λ in $\{1, 10, 20, 30, 40\}$.

We can see that setting *Len* too large or too small will deteriorate the performance. A larger *Len* may introduce noise, and thus lead to worse performance. Meanwhile, a smaller *Len* can ensure that it contains a single intention, but may break the completeness of sequential patterns. Overall, $Len = 5$ is the optimal setting.

In terms of λ, we can see that CAST always achieves the best performance with $\lambda = 1$ on MovieLens, because it has the smallest number of categories. Consequently, the *KL* regularization is not necessary for model training on this dataset. However, this phenomenon is different for Gowalla, JD, and Taobao. First, these datasets had a large number of categories showing a severe semantic imbalance problem. Thus, the *KL* regularization becomes an essential part, and a relatively larger λ is required for better performance. Second, when comparing the best λ settings on *AUC* and *HNF@1*, we can observe that a rel-

Table 3
Results for architecture validation.

Dataset	Metrics	parallel	stacking	translation	
				S_aS_t	S_tS_a
MovieLens	HNF@1	0.2028	0.2104	0.2195	0.2240
	Hit@10	0.5704	0.5695	0.5840	0.5832
	NDCG@10	0.3698	0.3749	0.3862	0.3879
	F1@10	0.1037	0.1036	0.1062	0.1060
	AUC	0.9221	0.9221	0.9233	0.9230
Gowalla	HNF@1	0.4171	0.4150	0.4191	0.4144
	Hit@10	0.7693	0.7644	0.7751	0.7658
	NDCG@10	0.5838	0.5803	0.5873	0.5802
	F1@10	0.1399	0.1390	0.1410	0.1393
	AUC	0.9725	0.9722	0.9725	0.9697
JD	HNF@1	0.5077	0.4994	0.5022	0.5095
	Hit@10	0.7231	0.7175	0.7174	0.7225
	NDCG@10	0.6112	0.6044	0.6054	0.6116
	F1@10	0.1315	0.1304	0.1305	0.1314
	AUC	0.9436	0.9448	0.9413	0.9440
Taobao	HNF@1	0.2567	0.2476	0.2606	0.2626
	Hit@10	0.4944	0.4729	0.4826	0.4906
	NDCG@10	0.3674	0.3522	0.3638	0.3685
	F1@1	0.0899	0.0860	0.0877	0.0892
	AUC	0.8692	0.8635	0.8656	0.8689

Table 4
Results for components validation.

Dataset	Metrics	Original	-TSTa	-TSTt	-VAE	-PFP
MovieLens	HNF@1	0.2300	0.2268↓	0.2296↓	0.2258↓	0.2195↓
	Hit@10	0.6104	0.6072↓	0.6089↓	0.6015↓	0.5840↓
	NDCG@10	0.4048	0.4011↓	0.4039↓	0.3978↓	0.3862↓
	F1@10	0.1111	0.1104↓	0.1107↓	0.1094↓	0.1062↓
	AUC	0.9382	0.9371↓	0.9370↓	0.9358↓	0.9233↓
Gowalla	HNF@1	0.5289	0.5258↓	0.5009↓	0.4992↓	0.4191↓
	Hit@10	0.8365	0.8351↓	0.8143↓	0.7850↓	0.7751↓
	NDCG@10	0.6756	0.6733↓	0.6488↓	0.6338↓	0.5873↓
	F1@10	0.1521	0.1518↓	0.1480↓	0.1427↓	0.1410↓
	AUC	0.9816	0.9810↓	0.9787↓	0.9719↓	0.9725↓
JD	HNF@1	0.5298	0.5293↓	0.5231↓	0.5167↓	0.5095↓
	Hit@10	0.7790	0.7783↓	0.7712↓	0.7374↓	0.7225↓
	NDCG@10	0.6497	0.6495↓	0.6422↓	0.6223↓	0.6116↓
	F1@10	0.1416	0.1415↓	0.1402↓	0.1341↓	0.1314↓
	AUC	0.9605	0.9600↓	0.9582↓	0.9475↓	0.9440↓
Taobao	HNF@1	0.2812	0.2836↑	0.2739↓	0.2825↑	0.2626↓
	Hit@10	0.5717	0.5682↓	0.5622↓	0.5391↓	0.4906↓
	NDCG@10	0.4162	0.4159↓	0.4076↓	0.4017↓	0.3685↓
	F1@10	0.1040	0.1033↓	0.1022↓	0.0980↓	0.0892↓
	AUC	0.9110	0.9095↓	0.9069↓	0.9002↓	0.8689↓

atively larger λ results in a better overall ranking performance owing to the *KL* regularization function, which prevents the model from overfitting. Thus, we choose $\lambda = 20$ on Gowalla and Taobao, and $\lambda = 40$ on JD as the best settings.

5.5. Computational complexity analysis

We first study the training efficiency of our CAST model and draw the learning curves in Fig. 5). Owing to space limitations, we only present the results for Gowalla. Obviously, the losses for prediction decrease rapidly, while the losses for *KL* increase at an early training stage. After adding the *KL* terms after the 4^{th} epoch, all losses show a substantial change, and then converge quickly. These results demonstrate that the proposed model is easy to train.

We then compare the computational cost of CAST with that of four representative methods: SASRec and ATRank for those without considering context or without considering context as a sequence, and CBS and STAR for context sequence-based methods. The results are listed in Table 5.

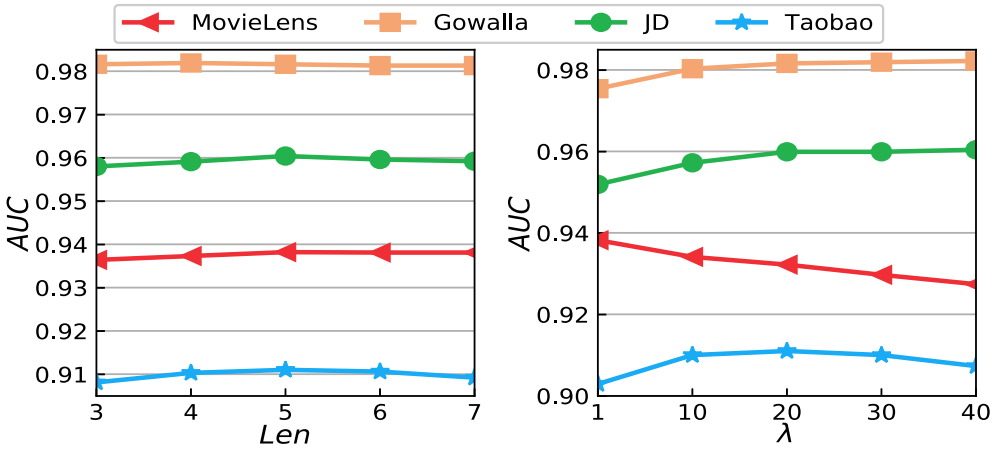

Fig. 3. Impacts of two hyperparameters on AUC.

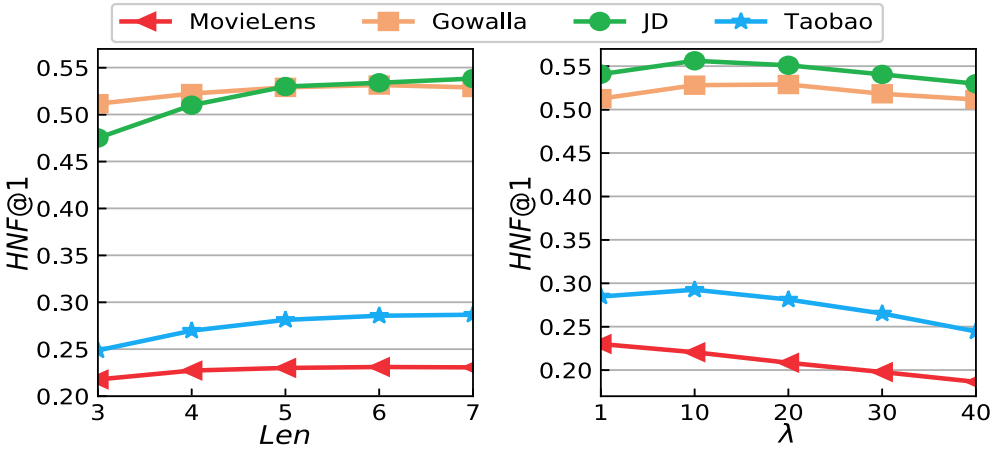

Fig. 4. Impacts of two hyperparameters on HNF@1.

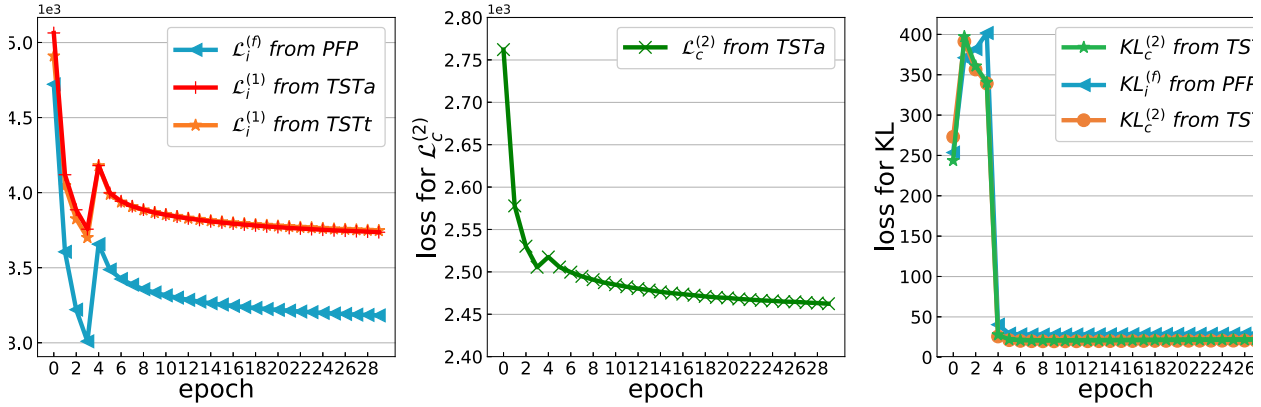

Fig. 5. Learning curves on Gowalla.

Table 5
Computational cost comparison on Gowalla.

Methods	runtime/epoch	converge epoch	total runtime
SASRec	12.35s	200	2470.0s
ATRank	3226.69s	5	16133.45s
CBS	24.0s	100	2400.0s
STAR	1276.9s	20	25538.0s
CAST	124.4s	30	3732.0s

We can see that CBS and SASRec have similar efficiencies, and ATRank and STAR are extremely time-consuming. Overall, we find that our model is computationally comparable with CBS and SASRec, and it is much more efficient than ATRank and STAR.

6. Conclusion

We proposed a novel context-aware seq2seq translation (CAST) model for sequential recommendation. The key insight is that the bidirectional inter-sequence dependency between item sequence S_i and context sequence S_c plays a critical role in sequential recommendation. We implemented this insight with a tripled seq2seq translation (TST) module to forward translate S_i into S_c, and then backward translate S_c to S_i. We further injected a VAE structure into the translation process to address the semantic imbalance problem between the item and the context. Moreover, we stacked the TST module to incorporate multiple context sequences. Finally, we combined the TST module with a personalized prediction module into a complete CAST model, such that both dynamic and static preferences can take effect. We conducted extensive experiments using four real-world datasets. The results demonstrate that the proposed CAST model outperforms the state-of-the-art baseline methods.

CRediT authorship contribution statement

Ke Sun: Investigation, Data curation, Conceptualization, Methodology, Software, Visualization, Validation, Writing - original draft, Writing - review & editing. **Tieyun Qian:** Supervision, Investigation, Conceptualization, Methodology, Writing - original draft, Writing - review & editing. **Xu Chen:** Investigation, Conceptualization. **Ming Zhong:** Investigation, Conceptualization.

Declaration of Competing Interest

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

Acknowledgments

This work was supported in part by NSFC Projects (61572376, 91646206). It is also funded by the Joint Laboratory on Credit Science and Technology of CSCI-Wuhan University. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University.

References

[1] T. Bai, J.Y. Nie, W.X. Zhao, Y. Zhu, P. Du, J.R. Wen, An attribute-aware neural attentive model for next basket recommendation, in: Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, 2018, pp. 1201–1204.
[2] B. Chang, Y. Park, D. Park, S. Kim, J. Kang, Content-aware hierarchical point-of-interest embedding model for successive poi recommendation, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3301–3307.
[3] X. Chen, H. Xu, Y. Zhang, J. Tang, Y. Cao, Z. Qin, H. Zha, Sequential recommendation with user memory networks, in: Proceedings of the 11th ACM International Conference on Web Search and Data Mining, 2018, pp. 108–116.
[4] S. Guo, Y. Wang, H. Yuan, Z. Huang, J. Chen, X. Wang, Taert: triple-attentional explainable recommendation with temporal convolutional network, Inf. Sci. 567 (2021) 185–200.
[5] J. He, X. Li, L. Liao, Category-aware next point-of-interest recommendation via listwise bayesian personalized ranking, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 1837–1843.
[6] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based recommendations with recurrent neural networks. arXiv:1511.06939 (2015)..
[7] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, A. Lerchner, Beta-vae: learning basic visual concepts with a constrained variational framework, in: Proceedings of the 5th International Conference on Learning Representations, 2017, pp. 1–22..
[8] J. Huang, Z. Ren, W.X. Zhao, G. He, J.R. Wen, D. Dong, Taxonomy-aware multi-hop reasoning networks for sequential recommendation, in: Proceedings of the 12th ACM International Conference on Web Search and Data Mining, 2019, pp. 573–581.
[9] X. Huang, S. Qian, Q. Fang, J. Sang, C. Xu, Csan: Contextual self-attention network for user sequential recommendation, in: Proceedings of the 26th ACM International Conference on Multimedia, 2018, pp. 447–455.

[10] W.C. Kang, J. McAuley, Self-attentive sequential recommendation, in: Proceedings of the 2018 IEEE 18th International Conference on Data Mining, 2018, pp. 197–206.
[11] D.P. Kingma, M. Welling, Auto-encoding variational bayes. arXiv:1312.6114 (2013)..
[12] D.T. Le, H.W. Lauw, Y. Fang, Modeling contemporaneous basket sequences with twin networks for next-item recommendation, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 3414–3420.
[13] J. Li, Y. Wang, J. McAuley, Time interval aware self-attention for sequential recommendation, in: Proceedings of the 13th ACM International Conference on Web Search and Data Mining, 2020, pp. 322–330.
[14] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, J. Ma, Neural attentive session-based recommendation, in: Proceedings of the 26th ACM International Conference on Information and Knowledge Management, 2017, pp. 1419–1428.
[15] R. Li, Y. Shen, Y. Zhu, Next point-of-interest recommendation with temporal and multi-level context attention, in: Proceedings of the 2018 IEEE 18th International Conference on Data Mining, 2018, pp. 1110–1115.
[16] D. Liang, R.G. Krishnan, M.D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: Proceedings of the 27th International Conference on World Wide Web, 2018, pp. 689–698.
[17] Q. Liu, S. Wu, D. Wang, Z. Li, L. Wang, Context-aware sequential recommendation, in: Proceedings of the 2016 IEEE 16th International Conference on Data Mining, 2016, pp. 1053–1058.
[18] Q. Liu, S. Wu, L. Wang, T. Tan, Predicting the next location: a recurrent model with spatial and temporal contexts, in: Proceedings of the 30th AAAI Conference on Artificial Intelligence, 2016, pp. 194–200.
[19] Y. Luo, Q. Liu, Z. Liu, Stan: spatio-temporal attention network for next location recommendation, in: Proceedings of the 30th International Conference on World Wide Web, 2021, pp. 2177–2185.
[20] M.T. Luong, H. Pham, C.D. Manning, Effective approaches to attention-based neural machine translation. arXiv:1508.04025 (2015)..
[21] C. Ma, P. Kang, X. Liu, Hierarchical gating networks for sequential recommendation, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data mining, 2019, pp. 825–833.
[22] C. Ma, L. Ma, Y. Zhang, J. Sun, X. Liu, M. Coates, Memory augmented graph neural networks for sequential recommendation, in: Proceedings of the 34th AAAI Conference on Artificial Intelligence, 2020, pp. 5045–5052.
[23] L. Rakkappan, V. Rajan, Context-aware sequential recommendations with stacked recurrent neural networks, in: Proceedings of the 28th International Conference on World Wide Web, 2019, pp. 3172–3178.
[24] S. Rendle, C. Freudenthaler, L. Schmidt-Thieme, Factorizing personalized markov chains for next-basket recommendation, in: Proceedings of the 19th International Conference on World Wide Web, 2010, pp. 811–820.
[25] N. Sachdeva, G. Manco, E. Ritacco, V. Pudi, Sequential variational autoencoders for collaborative filtering, in: Proceedings of the 12th ACM International Conference on Web Search and Data Mining, 2019, pp. 600–608.
[26] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: sequential recommendation with bidirectional encoder representations from transformer, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1441–1450.
[27] K. Sun, T. Qian, H. Yin, T. Chen, Y. Chen, L. Chen, What can history tell us?, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 1593–1602.
[28] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Proceedings of the Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[29] Q. Tan, J. Zhang, N. Liu, X. Huang, H. Yang, J. Zhou, X. Hu, Dynamic memory based attention network for sequential recommendation, in: Proceedings of the 35th AAAI Conference on Artificial Intelligence, 2021, pp. 4384–4392.
[30] J. Tang, K. Wang, Personalized top-n sequential recommendation via convolutional sequence embedding, in: Proceedings of the 11th ACM International Conference on Web Search and Data Mining, 2018, pp. 565–573.
[31] C. Wang, M. Zhang, W. Ma, Y. Liu, S. Ma, Make it a chorus: knowledge-and time-aware item modeling for sequential recommendation, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 109–118.
[32] H. Wang, P. Li, Y. Liu, J. Shao, Towards real-time demand-aware sequential poi recommendation, Inf. Sci. 547 (2021) 482–497.
[33] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, X. Cheng, Learning hierarchical representation model for nextbasket recommendation, in: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2015, pp. 403–412.
[34] J. Wu, R. Cai, H. Wang, Déjà vu: a contextualized temporal attention mechanism for sequential recommendation, in: Proceedings of the 29th International Conference on World Wide Web, 2020, pp. 2199–2209.
[35] S. Wu, Y. Zhang, C. Gao, K. Bian, B. Cui, Garg: anonymous recommendation of point-of-interest in mobile networks by graph convolution network, Data Sci. Eng. 5 (4) (2020) 433–447.
[36] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, X. Zhang, : Self-supervised hypergraph convolutional networks for session-based recommendation, in: Proceedings of the 35th AAAI Conference on Artificial Intelligence, 2021, pp. 4503–4511.
[37] C. Yang, L. Bai, C. Zhang, Q. Yuan, J. Han, Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 1245–1254.
[38] D. Yao, C. Zhang, J. Huang, J. Bi, Serm: a recurrent model for next location prediction in semantic trajectories, in: Proceedings of the 26th ACM International Conference on Information and Knowledge Management, 2017, pp. 2411–2414.
[39] W. Ye, S. Wang, X. Chen, X. Wang, Z. Qin, D. Yin, Time matters: sequential recommendation with complex temporal information, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1459–1468.
[40] F. Yu, Q. Liu, S. Wu, L. Wang, T. Tan, A dynamic recurrent model for next basket recommendation, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2016, pp. 729–732.
[41] Z. Yu, J. Lian, A. Mahmoody, G. Liu, X. Xie, Adaptive user modeling with long and short-term preferences for personalized recommendation, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 4213–4219.
[42] F. Yuan, X. He, H. Jiang, G. Guo, J. Xiong, Z. Xu, Y. Xiong, Future data helps training: Modeling future contexts for session-based recommendation, in: Proceedings of the 29th International Conference on World Wide Web, 2020, pp. 303–313.
[43] S. Zhang, Y. Tay, L. Yao, A. Sun, Next item recommendation with self-attention. arXiv preprint arXiv:1808.06414 (2018)..
[44] C. Zhou, J. Bai, J. Song, X. Liu, Z. Zhao, X. Chen, J. Gao, Atrank: an attention-based user behavior modeling framework for recommendation, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, 2018, pp. 4564–4571.
[45] G. Zhou, N. Mou, Y. Fan, Q. Pi, W. Bian, C. Zhou, X. Zhu, K. Gai, Deep interest evolution network for click-through rate prediction, in: Proceedings of the 33rd AAAI Conference on Artificial Intelligence, 2019, pp. 5941–5948.
[46] H. Yu, T. Qian, Y. Liang, B. Liu, Agtr: Adversarial generation of target review for rating prediction, Data Sci. Eng. 5 (4) (2020) 346–359.
[47] Q. Zeng, M. Zhong, Y. Zhu, T. Qian, J. Li, Business location planning based on a novel geo-social influence diffusion model, Inf. Sci. 559 (2021) 61–74.