Context-Aware Sequential Recommendations with Stacked Recurrent Neural Networks

Lakshmanan Rakkappan
Department of Computer Science
School of Computing, National University of Singapore
e0223207@u.nus.edu

Vaibhav Rajan
Department of Information Systems and Analytics
School of Computing, National University of Singapore
vaibhav.rajan@nus.edu.sg

ABSTRACT

Sequential history of user interactions as well as the context of interactions provide valuable information to recommender systems, for modeling user behavior. Modeling both contexts and sequential information simultaneously, in context-aware sequential recommenders, has been shown to outperform methods that model either one of the two aspects. In long sequential histories, temporal trends are also found within sequences of contexts and temporal gaps that are not modeled by previous methods. In this paper we design new context-aware sequential recommendation methods, based on Stacked Recurrent Neural Networks, that model the dynamics of contexts and temporal gaps. Experiments on two large benchmark datasets demonstrate the advantages of modeling the evolution of contexts and temporal gaps – our models significantly outperform state-of-the-art context-aware sequential recommender systems.

CCS CONCEPTS

• **Computing methodologies → Neural networks**; • **Information systems →** *Content ranking*; Collaborative filtering.

KEYWORDS

Sequential Recommendation; Context-Aware Recommendation; Stacked Recurrent Neural Network

ACM Reference Format:
Lakshmanan Rakkappan and Vaibhav Rajan. 2019. Context-Aware Sequential Recommendations with Stacked Recurrent Neural Networks. In *Proceedings of the 2019 World Wide Web Conference (WWW'19), May 13–17, 2019, San Francisco, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3308558.3313567

1 INTRODUCTION

In a world of overwhelming number of choices, recommender systems have become indispensable as personalized information filters, especially for e-commerce. Developing accurate recommendation algorithms poses several challenges [1] and continues to be of great interest to both industry and academia. Among the large number of algorithms developed, arguably, the most well known are Collaborative Filtering methods that factorize the historical user-item preference matrix to obtain latent user representations, that, in turn, are used to predict future behaviour.

Sequential history of users provide valuable clues to predict future behaviour. Recommender systems based on Markov chains [2] and combinations of matrix factorization and Markov chains [15] leverage such historical sequential behavior. But, methods based on Markov chains only capture local sequential patterns, and Recurrent Neural Networks (RNN) are found to be more effective in capturing global sequential history [24] thereby yielding better recommendations. However, these methods do not explicitly model the dependencies on the temporal gaps (also called *temporal context* in previous literature) between events.

Additional *input context* information, such as location, time of day, and local holiday information, include another dimension of information, that is often available (e.g., through smartphone apps). Context-aware recommender systems, effectively use such information and have been extensively studied [16, 18]. Although most of the previous literature has studied them separately, contexts and sequential information are not independent. For example, gap statistics between purchases can differ with location and other contextual variables. Liu *et al.* demonstrate the advantages of simultaneous modeling of context and sequential history over modeling either of them separately through their method Context-Aware RNN [10], that also explicitly models the temporal context. However, their model requires the number of possible input and temporal contexts (through discretizing time intervals into specific bins) to be pre-specified. A separate parameter matrix is learnt for each user-specified context, during training. Thus, the number of model parameters grow with the number of input and temporal contexts, making the task of training increasingly more difficult.

With long sequential histories of users, we observe that temporal trends are found even within changing input and temporal contexts, that, to our knowledge, has not been modeled by previous recommendation methods. We design new context-aware sequential recommender methods using stacked RNNs that model the dynamics of input and temporal contexts. Thus, instead of modeling separate contexts with different parameter matrices, we model how the context evolves with a single recurrent parameter matrix. This also effectively addresses the limitation of previous models: (1) different possible temporal contexts need not be specified in advance during training and (2) number of temporal context parameters do not increase with increasing number of contexts. To summarize, our main contributions in this paper are:

- We design new context-aware sequential recommendation methods:
  - that model the dynamics of input and temporal contexts, that has not been modeled in previous methods.
  - where model parameters do not depend on the number of temporal contexts; in the best previous models, the

WWW '19, May 13–17, 2019, San Francisco, CA, USA

ACM ISBN 978-1-4503-6674-8/19/05.
https://doi.org/10.1145/3308558.3313567

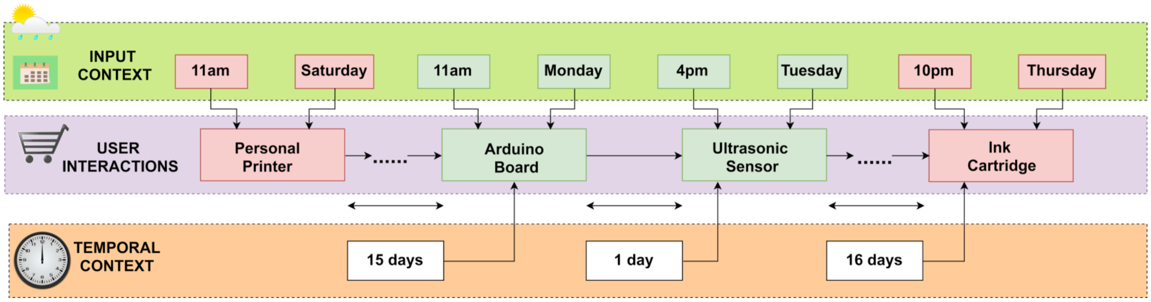

Figure 1: Schematic of user interactions with input and temporal contexts

number of parameters increase with increasing number of temporal and input contexts, thereby making training increasingly harder.

- Our experiments on two large benchmark datasets show that our models significantly outperform state-of-the-art context-aware sequential recommendation methods.

2 PROBLEM DEFINITION

Let u denote a user. Let $X = \{x_1^u, x_2^u, \ldots\}$ denote the sequence of items the user u has interacted with: x_t^u denotes the item that the user u has interacted with at timestep t. Interaction could be purchase, click, view or any other activity relevant to the application. With each interaction x_t^u of a user, we associate an **input context**, c_t^u, and a **temporal context**, z_t^u. Input context can represent physical or social context. For s types of input contexts, the input context c_t^u is represented by an se-dimensional embedding. We have Σ such embeddings for each possible value of an input context. Temporal context represents the interval between consecutive interactions, i.e., z_t^u refers to the time interval between interactions x_t^u and $x_{(t-1)}^u$. It is a scalar, in a suitable unit (like days) that can be application dependent. Figure 1 shows a schematic of user interactions and both contexts. The sequential recommendation task is to predict $x_{(t+1)}^u$ for a user u, given historical data consisting of past interactions $\{x_1^u, \ldots, x_t^u\}$ together with their input contexts $\{c_1^u, \ldots, c_t^u\}$ and temporal contexts $\{z_1^u, \ldots, z_t^u\}$.

3 RELATED WORK

Sequential Recommendation. Sequential (or next basket) recommendation algorithms based on Markov chains (e.g., [2]) use features derived from past transactions to predict a user's next interaction. These methods typically estimate a transition matrix from past data of all users, which in turn is used to predict the probability of the next interaction. Factorizing Personalized Markov Chains (FPMC) [15] combine the strengths of Markov chains and matrix factorization. In FPMC, a transition cube is estimated, the added third dimension for personalizing user-specific transitions. This transition tensor is factorized to learn latent factors and enables information propagation among similar users, items and transitions. Since the operations used are linear, FPMC cannot capture interactions among multiple factors because each component influences a user's next interaction independently [22]. The Hierarchical Representation Model (HRM) addresses this problem by summarizing multiple interacting factors through a nonlinear max pooling operation [22]. HRM uses continuous-valued representations of users and items and builds a hybrid representation over users and items from the previous interaction. Both FPMC and HRM model only local interactions between consecutive transactions.

There may be correlations between items that are not adjacent in a user's interaction history, e.g., many users may purchase an ink cartridge after buying a printer, but the time interval between the purchases may differ across users, depending on their usage differences. Further, the purchases may not be in consecutive baskets. The Dynamic REcurrent bAsket Model (DREAM) uses a pooling operation, similar to that of HRM, but uses Recurrent neural networks (RNN) to model global sequential interactions[24]. The hidden layer within the RNN is used as the dynamic user representation which is used to compute a per-item score at a time-step, with a higher score corresponding to higher likelihood of interaction. The Bayesian Pairwise Ranking (BPR) objective function [14] is optimized using Back Propagation Through Time (BPTT) [17]. DREAM significantly outperforms HRM and FPMC demonstrating the effectiveness of RNNs in mining global sequential interactions.

Context-Aware Recommendation. Contextual information such as weather, location, holidays etc. can provide significant performance improvement to personalized recommendations [12]. For instance, users may buy some items before a specific social occasion that they may not purchase often. Context-aware recommendations, broadly, use three types of strategies. In pre-filtering strategies, data may be filtered before applying collaborative filtering. In post-filtering strategies, collaborative filtering is applied first, ignoring contextual information, and then the recommendations are adjusted taking the context into account [1]. Thirdly, the context can directly be taken into account in the recommendation model using generalizations of matrix factorization, such as collective matrix factorization [19] and tensor factorizations [8], or through the use of factorization machines FM [16]. Treating the context in a way similar to users or items, cannot model the semantic operations of contexts. Both CARS2 [18] and COT [11] address this problem through the use of context-specific latent representations. Recent related work include the use of attention-based neural models in predicting next item within a transactional context [23]. However, these methods do not model sequential dependencies in the user history.

Sequential Context-Aware Recommendation. To model both contextual information and sequential dependencies, the Context-Aware

Recurrent Neural Network (CARNN) model was proposed [10]. They use an RNN based dynamic representation of users (similar to that of DREAM). Modeling both contextual and sequential information yields significant performance improvements over both sequential recommendation methods (like DREAM, FPMC and HRM) and context-aware recommendation methods (like CARS2 and FM).

Other Related Work. Collaborative nowcasting for 'intent monitoring' also models contexts and sequential evolution of user representations [20, 21]. However they do not model sequential evolution of contexts. Moreover, their models, based on Kalman filters, can only capture linear dynamics of user representations that are not as flexible as RNNs that can capture non-linear dynamics. A Bayesian Hidden Markov Model for modeling user contexts was proposed in [7] where sequential dependencies in contexts were modeled and effectively utilized for personalized context recognition. They do not evaluate their model for sequential recommendations. Further, their model is semi-supervised and models only linear dynamics. The method CDUE integrates vector autoregressive and dynamic topic modeling of contents information and matrix factorization [4]. They learn users' interests and items' topics and their evolutions collaboratively and simultaneously. But their problem setting is different: they assume access to the user's browsing history (that can be considered a specific type of contextual information), through which topic evolution is learnt, to predict a future item.

4 CONTEXT-AWARE SEQUENTIAL RECOMMENDATION

4.1 Background

Recurrent Neural Networks (RNN) effectively model sequential user interactions by learning a recurrent hidden user representation h_t, from the user's current interaction x_t^u, at each time step t:

$$h_t = g(x_t^u M + h_{t-1} W) \tag{1}$$

where g is a non-linear activation function (e.g. sigmoid). Transition matrix M models user behavior with respect to the current interaction and weight matrix W models the recurrent connection that propagates signal between consecutive hidden states. Both M, W are learnt from historical data. In the following we call this RNN, the user RNN or *URNN*.

CARNN [10] models contextual effects through input and temporal context matrices within URNN. Specifically, the hidden representation of a user, h_t, at timestep t is given by:

$$h_t = g(x_t^u M_{c_I} + h_{t-1} W_{z_T})$$

where M_{c_I} and W_{z_T} are context-specific weight matrices for input contexts (c_I) and temporal contexts (z_T) respectively. In CARNN, the number of input and temporal context matrices (M_{c_I} and W_{z_T}) have to be pre-specified during training. For instance, the day of the week and hour of the day can be used as contexts yielding a total of $7 \times 24 = 168$ different contexts (requiring up to 168 parameter matrices). Also, since it is impossible to learn transition context weight matrix (W_{z_T}) for every possible time interval, the intervals are discretized into chosen bins, and a matrix is learnt for each such discretized interval. For long sequences, the number of temporal contexts can be very large. They could be reduced by choosing larger bins but that may not be able to capture the effects of different temporal contexts. Liu, Wu and Wang [11] also recognize the problem of growing number of context-specific parameters with increase in number of contexts and address the problem in their context-aware (but not sequential) recommendation model.

4.2 Our Approach

We model the dynamics of temporal and input contexts and their effects on hidden representations of users. The evolution of temporal contexts (z_t^u) is modeled through an RNN (that we call *TCRNN*):

$$p_t = g(z_t^u E + p_{t-1} Q),$$

and the evolution of input contexts (c_t^u) is modeled through another RNN (that we call *ICRNN*):

$$v_t = g(c_t^u A + v_{t-1} K),$$

in addition to URNN (equation 1) that models the evolution of user representation. In URNN, since matrix W propagates information between consecutive time steps, we replace W with a signal from TCRNN to incorporate the effects of evolving temporal contexts. Similarly we replace M with a signal from ICRNN to model the effects of evolving input contexts. The details of how these signals are effected are given below.

Thus we have two (or more) interacting RNNs where the evolving hidden representation of the user h_t is also affected by the recurrent hidden context representations. Our models can be seen as variants of stacked RNNs [3, 13]. Stacked RNNs are constructed by stacking multiple recurrent hidden layers, with the goal of capturing representations at different timescales in each recurrent level. However, in our models, each RNN models a distinct temporal process with different inputs at each time step. Thus, they can also be viewed as multiple interacting RNNs.

We describe three models – STAR, SIAR and SITAR. STAR and SIAR model evolving temporal and input contexts respectively, and SITAR models the evolution of both temporal and input contexts.

STAR: Evolving Temporal Contexts

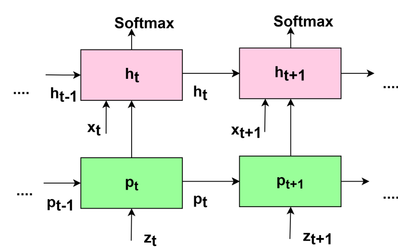

Figure 2: Stacked Temporal-context Aware RNN (STAR)

A schematic of our model Stacked Temporal-context Aware RNN (STAR) is shown in figure 2. Each block is an RNN cell at a single timestep (t). The RNN shown above in figure 2 models the user's evolving representation (h_t) and is dependent on not just the current interaction (x_t^u) and previous user representation (h_{t-1}) but also the hidden representation (p_t) of the evolving temporal context (z_t^u) modeled by the lower TCRNN.

In TCRNN, the recurrent connection weight matrix Q propagates sequential signals between consecutive hidden representations p_t and p_{t-1}, and E is a transition matrix between the temporal context and its hidden representation. We use an intermediate feedforward layer to transform p_t to J_t – to maintain a lower dimension of d in the TCRNN's hidden representation p_t and transform to a higher dimension of $d \times d$ in J_t that is used, instead of W, in URNN. The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} p_t &= g(z_t^u E + p_{t-1} Q) \\ J_t &= g(p_t L) \\ h_t &= g(x_t^u M + h_{t-1} J_t) \end{aligned}$$

where g denotes the activation function. Matrix M plays the role of a transition matrix between the user interaction (x_t^u) and user representation (h_t). The transformed temporal context representation J_t (reshaped from $1 \times d^2$ to $d \times d$) is used to propagate the signal between h_{t-1} and h_t. Weight matrices E, Q, L and M are of dimensions $1 \times d$, $d \times d$, $d \times d^2$ and $d \times d$ respectively, and are learnt during training. The key difference between URNN (equation 1) and STAR is that the weight matrix W that propagates information between consecutive time steps in URNN, is replaced by J_t from TCRNN to incorporate the effects of evolving temporal contexts.

SIAR: Evolving Input Contexts

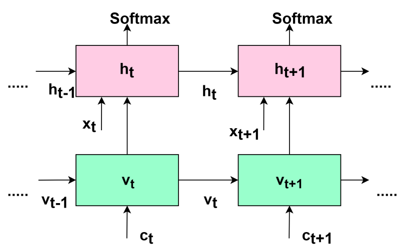

Figure 3: Stacked Input-context Aware RNN (SIAR)

A schematic of our model Stacked Input-context Aware RNN (SIAR) is shown in figure 3. Each block is an RNN cell at a single timestep (t). The RNN shown above in figure 3 models the user's evolving representation (h_t) and is dependent on not just the current interaction (x_t^u) and previous user representation (h_{t-1}) but also the hidden representation (v_t) of the evolving input context (c_t^u) modeled by the lower ICRNN. In ICRNN, the recurrent connection weight matrix K propagates sequential signals between consecutive hidden representations v_t and v_{t-1}, and A is a transition matrix between the input context and its hidden representation. Just like in the case of STAR, we use an intermediate feedforward layer to transform v_t to I_t. This is done to maintain a lower dimension of d in the ICRNN's hidden representation v_t and transform to a higher dimension of $d \times d$ in I_t that is used, instead of M, in URNN. The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} v_t &= g(c_t^u A + v_{t-1} K) \\ I_t &= g(v_t F) \\ h_t &= g(x_t^u I_t + h_{t-1} W) \end{aligned}$$

where g denotes the activation function. The matrix W is learnt to propagate the signal between h_{t-1} and h_t. The transformed input context representation I_t (reshaped from $1 \times d^2$ to $d \times d$) is used to model the transition matrix between the user interaction (x_t^u) and user representation (h_t). A, K, F and W are weight matrices of dimensions $se \times d$, $d \times d$, $d \times d^2$ and $d \times d$ respectively that are learnt during training. For s types of input contexts, the input context c_t^u is represented by an se-dimensional embedding. The main difference between URNN (equation 1) and SIAR is that the weight matrix M that models the effect of current interaction in URNN, is replaced by I_t from ICRNN to incorporate the effects of evolving input contexts.

SITAR: Evolving Temporal and Input Contexts

Our model, Stacked Input-context and Temporal-context Aware RNN (SITAR), models both evolving temporal and input contexts through TCRNN and ICRNN respectively and signals from both the RNNs are used to model the recurrent user representation as shown in figure 4.

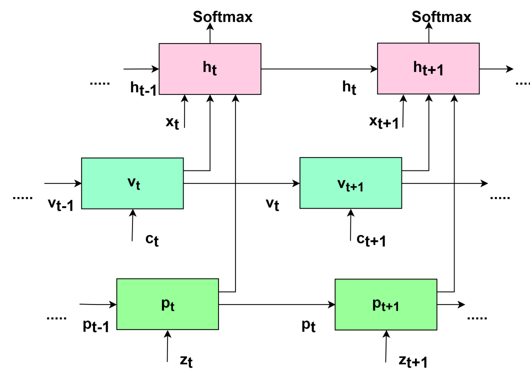

Figure 4: Stacked Input-context and Temporal-context Aware RNN (SITAR)

The user's evolving representation (h_t) is thus dependent on the current interaction (x_t^u), the previous user representation (h_{t-1}) as well as the hidden representations (p_t, v_t) of the evolving input context (c_t^u) and temporal context (z_t^u). The dynamic d-dimensional hidden user representation h_t is given by:

$$\begin{aligned} v_t &= g(c_t^u A + v_{t-1} K) \\ I_t &= g(v_t F) \\ p_t &= g(z_t^u E + p_{t-1} Q) \\ J_t &= g(p_t L) \\ h_t &= g(x_t^u I_t + h_{t-1} J_t) \end{aligned}$$

where weight matrices A, K, F, E, Q, L (as defined in models STAR and SIAR earlier) are learnt during training. Comparing this model with URNN (equation 1) and CARNN, we can view this model as using time-dependent parameters for both input and temporal contexts (through I_t and J_t), that implicitly model different possible input and temporal contexts through their temporal evolution, instead of explicitly modeling different possible contexts.

4.3 Training and Hyperparameter Settings

Backpropagation Through Time (BPTT) was used to learn the model parameters in all our models. We used the Adam optimizer [9] for training our models with a learning rate of 0.001, $\beta_1, \beta_2, \epsilon$ values set to 0.9, 0.999 and 10^{-8} respectively. We used cross-entropy loss with L2 regularization, sigmoid activation function (g), and random initialization. The source code for our models and all experimental results are available on our public repository.[1]

Number of Context-dependent Parameters. For s input context types, let Σ be the sum of possible values and Π be the number of combinations of values of the input contexts. E.g., with weekday and hour as contexts, $s = 2, \Sigma = 7 + 24 = 31, \Pi = 7 \times 24 = 168$. With p temporal contexts, d-dimensional hidden representation, and e-dimensional input embedding, the number of context-dependent parameters to be learnt in our models and CARNN are listed in table 1.

Table 1: Number of context-dependent parameters

Model:	STAR	SIAR	SITAR	CARNN
# Parameters:	0	$sed + e\Sigma$	$sed + e\Sigma$	$(\Pi + p)d^2$

Both d, e are fixed numbers that do not grow with increasing number of contexts. In our experiments, the models perform well at $e = d = 20$ and the performance does not significantly change at higher values of d (figures 5 and 6). However, the number of temporal contexts (p) can potentially be very large in long sequential user histories. The number of parameters in SIAR and SITAR are independent of the number of temporal contexts while in STAR, they are independent of both input and temporal contexts. In contrast, number of parameters in CARNN increase with both contexts.

Time Complexity. The per-timestep training time complexity is $O(d^3)$ for STAR and $O(d^3 + sed)$ for SIAR and SITAR.

4.4 Prediction

In our models, we use a softmax layer that takes the hidden representation of the user h_t at each timestep to generate a probability vector for all the items. The user representation is transformed to a vector m of size N through a linear transformation, $m = h_t B$, where N is the total number of items, to which probabilities are assigned at each time step, and B is learnt during training. The probability of the next item j is obtained through softmax: $\frac{e^{m_j}}{\sum_k e^{m_k}}$. The most probable item is used as the prediction. In our experiments, BPR [14] (instead of softmax) did not improve results.

5 EXPERIMENTS

5.1 Datasets

We used two large benchmark datasets:
1. **Movielens-1M** data[2]: that contains 1,000,209 movie ratings of 3,900 movies that has been collected from 6,040 users [5].
2. **Amazon Books** data[3]: that has 22,507,155 user ratings of 2,330,066 books from 8,026,324 users that was collected between May 1996 to July 2014 [6].

In each dataset, we utilised the first 80% of each user's history for training and the remaining 20% as the test set. We did not use data of users and items with very few interactions to avoid data sparsity: in the Movielens dataset, we selected users with at least 10 records and items with at least 3 records. In the Amazon dataset, we chose users and items with at least 25 records each. We extracted both input and temporal contextual information from the timestamps of ratings in both datasets. For temporal contexts, we measured time intervals in days using upward rounding. All intervals greater than 30 days were mapped to a single value. For input contexts, we only used the day of the week and the hour of the day to have a fair comparison with our main baseline CARNN (that used the same contexts), to have 7 day-based contexts and 24 hour-based contexts.

5.2 Baselines

CARNN [10] is our main baseline since it is a state-of-the-art method that models both contextual information and sequential dependencies. It was shown to outperform both context-aware recommenders (CARS2 and FM) and sequential recommenders (DREAM, FPMC and HRM) in their experiments. In addition, we use the RNN-based model **DREAM** [24] and Markov chain based model **FPMC** [15] as baselines, to compare with competitive sequential recommender models. The parameter transaction length is set to 1 week in both these methods. Methods that take neither context nor sequential information into account, **BPR** [14] (using matrix factorization) and **POP**, are used as additional baselines. Publicly available code provided by the authors were used for these baselines, except for POP, where items are ranked by frequency and the most frequent item is recommended.

5.3 Evaluation Metrics

For each user we predict the final 20% of the interactions. We denote this list of items, concatenated over all users, by Y and Y_q denotes the item in the q^{th} position in this list. At each step, our models output a list of probabilities (through softmax). Let $\hat{Y}_q$ denote this list, $\{\hat{Y}_q^1, \hat{Y}_q^2, \ldots, \hat{Y}_q^N\}$ at the q^{th} step with top-N predictions. The j^{th} element of this list, at the q^{th} step, is denoted by $\hat{Y}_q^j$. Following Liu et al. [10], we use metrics Recall@N (R@N), Precision@N (P@N), F1@N and normalized discounted cumulative gain (nDCG) averaged over the total number of items (where I denotes the indicator function) as follows:

$$P@N = \frac{1}{|Y|} P_q@N = \frac{1}{|Y|} \sum_{p=1}^{|Y|} \frac{|Y_q \cap \hat{Y}_q|}{|\hat{Y}_q|}$$

$$R@N = \frac{1}{|Y|} R_q@N = \frac{1}{|Y|} \sum_{p=1}^{|Y|} \frac{|Y_q \cap \hat{Y}_q|}{|Y_q|}$$

$$F1@N = 2 \times \frac{P@N \times R@N}{P@N + R@N}$$

$$nDCG = \frac{1}{|Y|} \sum_{q=1}^{|Y|} \sum_{j=1}^{|\hat{Y}|} \frac{2^{I(|\hat{Y}_q^j \cap Y_q|)} - 1}{log_2(j+1)}$$

5.4 Results

Table 2 show the results of all our models and the baseline methods on the Movielens-1M dataset. Among the baselines, CARNN has the best performance over all the metrics. Note that *all* the three

[1]https://bitbucket.org/cdal/stackedcontextawarernn
[2]http://grouplens.org/datasets/movielens/1m/
[3]http://jmcauley.ucsd.edu/data/amazon/

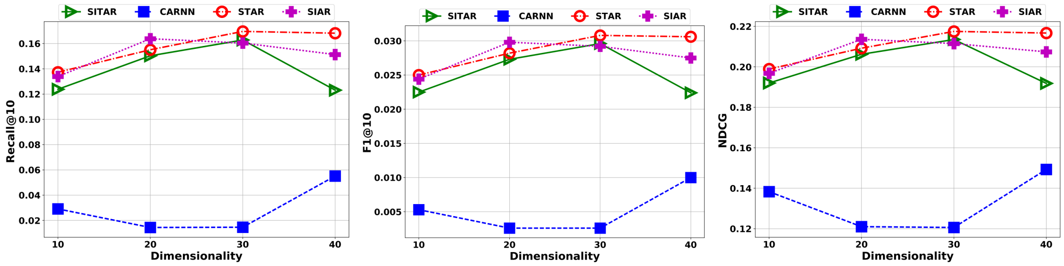

Figure 5: Performance of STAR, SIAR, SITAR and CARNN with varying dimensionality on Movielens dataset

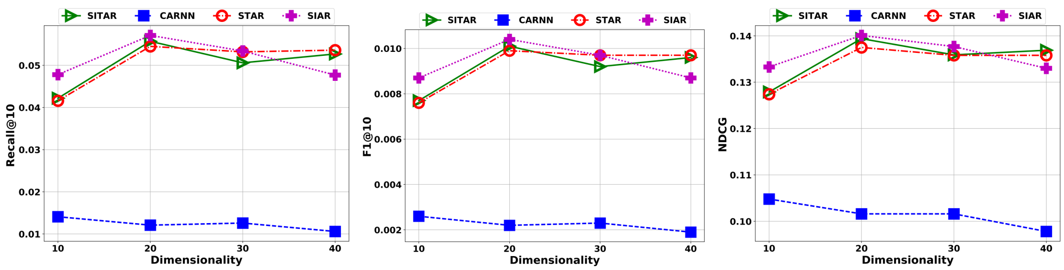

Figure 6: Performance of STAR, SIAR, SITAR and CARNN with varying dimensionality on Amazon Books dataset

models, STAR, SIAR, and SITAR, outperform CARNN in all the metrics. STAR is overall the best performing model.

Table 2: Performance of our models and baselines on the Movielens-1M dataset. Column-wise best results in bold, best baseline results in italics.

Model	R@1	R@5	R@10	F1@1	F1@5	F1@10	nDCG
POP	0.0016	0.0052	0.0118	0.0016	0.0017	0.0021	0.0434
BPR	0.0043	0.0237	0.0436	0.0043	0.0079	0.0033	0.1188
FPMC	0.0054	0.0255	0.0440	0.0054	0.0085	0.0080	0.1291
DREAM	0.0063	0.0318	0.0484	0.0063	0.0106	0.0088	0.1364
CARNN	*0.0075*	*0.0394*	*0.0551*	*0.0075*	*0.0131*	*0.0100*	*0.1493*
STAR	**0.0334**	**0.1070**	**0.1682**	**0.0334**	**0.0357**	**0.0306**	**0.2167**
SIAR	0.0303	0.1003	0.1604	0.0303	0.0334	0.0292	0.2115
SITAR	0.0300	0.1011	0.1628	0.0300	0.0337	0.0296	0.2135

Table 3 shows the results of all our models and the baseline methods on the Amazon Books dataset. Among the baselines, FPMC has the best performance over all the metrics. Note that all the three models, STAR, SIAR, and SITAR, outperform FPMC in all the metrics. Model SIAR has the best performance for metrics R@10, F1@10 and nDCG, whereas model STAR has the best performance for metrics R@1, R@5, F1@1 and R@5.

These results indicate that modeling the temporal evolution of input and temporal contexts yields significant improvement in recommendation performance. In the Movielens-1M dataset temporal context dynamics appears to have a more prominent role, whereas in the Amazon Books dataset, dynamics of both the contexts seem to be important. Although SITAR has significantly better performance than CARNN, it does not outperform STAR or SIAR. This could be due to the larger number of parameters that has to be learnt in SITAR.

5.4.1 Impact of Dimensionality. Figures 5 and 6 compare the performance of our models – STAR, SIAR and SITAR with CARNN, using evaluation metrics R@10, F1@10 and NDCG, at dimensions

Table 3: Performance of our models and baselines on the Amazon Books dataset. Column-wise best results in bold, best baseline results in italics.

Model	R@1	R@5	R@10	F1@1	F1@5	F1@10	nDCG
POP	0.0005	0.0016	0.0036	0.0005	0.0005	0.0007	0.0843
BPR	0.0008	0.0041	0.0078	0.0008	0.0014	0.0014	0.1016
FPMC	*0.0039*	*0.0163*	*0.0285*	*0.0039*	*0.0054*	*0.0052*	*0.1231*
DREAM	0.0008	0.0034	0.0063	0.0008	0.0011	0.0011	0.0915
CARNN	0.0015	0.0060	0.0106	0.0015	0.0020	0.0019	0.0978
STAR	**0.0247**	**0.0419**	0.0536	**0.0247**	**0.0140**	0.0097	0.1358
SIAR	0.0209	0.0413	**0.0571**	0.0209	0.0138	**0.0104**	**0.1401**
SITAR	0.0180	0.0397	0.0557	0.0180	0.0132	0.0101	0.1394

10, 20, 30 and 40. In all cases, our models outperform CARNN. In the Movielens-1M dataset, figure 5, performance of STAR, SIAR and SITAR are comparable at $d = 30$ whereas STAR outperforms the others at $d = 40$. In the Amazon Books dataset, figure 6, the best results of STAR, SIAR and SITAR are comparable, at $d = 20$, with SIAR marginally better than the others. In both the datasets, increasing the dimension beyond 20 does not significantly change the performance of STAR, SIAR and SITAR.

6 CONCLUSION

We design and evaluate context-aware sequential recommendation methods that, for the first time, model the dynamics of input and temporal contexts. Further, the number of parameters in STAR, one of our best performing models does not increase with the number of contexts, which was a limitation of the best previous model. Our experiments on large benchmark datasets show that modeling the evolution of contexts has considerable impact on recommendation performance – all our models significantly outperform state-of-the-art context-aware sequential recommendation models.

ACKNOWLEDGEMENTS

This work is supported by Singapore Ministry of Education Academic Research Fund Tier 1 [R-253-000-138-133].

REFERENCES

[1] Charu C Aggarwal. 2016. *Recommender Systems.* Springer.

[2] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. 2012. Playlist prediction via metric embedding. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 714–722.

[3] Salah El Hihi and Yoshua Bengio. 1996. Hierarchical recurrent neural networks for long-term dependencies. In *Proceedings of the 8th International Conference on Neural Information Processing Systems (NIPS).* 493–499.

[4] Li Gao, Jia Wu, Chuan Zhou, and Yue Hu. 2017. Collaborative Dynamic Sparse Topic Regression with User Profile Evolution for Item Recommendation.. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17).* 1316–1322.

[5] F Maxwell Harper and Joseph A Konstan. 2016. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TIIS)* 5, 4 (2016), 19.

[6] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web.* 507–517.

[7] Baoxing Huai, Enhong Chen, Hengshu Zhu, Hui Xiong, Tengfei Bao, Qi Liu, and Jilei Tian. 2014. Toward personalized context recognition for mobile users: A semisupervised Bayesian HMM approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 9, 2 (2014), 10.

[8] Alexandros Karatzoglou, Xavier Amatriain, Linas Baltrunas, and Nuria Oliver. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM Conference on Recommender Systems.* ACM, 79–86.

[9] Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations (ICLR).*

[10] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware sequential recommendation. In *Proceedings of the 16th International Conference on Data Mining (ICDM), 2016 IEEE.* IEEE, 1053–1058.

[11] Qiang Liu, Shu Wu, and Liang Wang. 2015. COT: Contextual Operating Tensor for Context-Aware Recommender Systems.. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence.* 203–209.

[12] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. 2008. Using context to improve predictive modeling of customers in personalization applications. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 20, 11 (2008), 1535–1549.

[13] Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2013. How to construct deep recurrent neural networks. *arXiv preprint arXiv:1312.6026* (2013).

[14] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence (UAI).* 452–461.

[15] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web.* ACM, 811–820.

[16] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 635–644.

[17] David E Rumelhart and James L McClelland. 1986. *Parallel distributed processing: explorations in the microstructure of cognition. volume 1. foundations.* MIT Press, Cambridge, MA.

[18] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, and Alan Hanjalic. 2014. Cars2: Learning context-aware representations for context-aware recommendations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM).* ACM, 291–300.

[19] Ajit P Singh and Geoffrey J Gordon. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 650–658.

[20] Yu Sun, Nicholas Jing Yuan, Yingzi Wang, Xing Xie, Kieran McDonald, and Rui Zhang. 2016. Contextual intent tracking for personal assistants. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM, 273–282.

[21] Yu Sun, Nicholas Jing Yuan, Xing Xie, Kieran McDonald, and Rui Zhang. 2016. Collaborative nowcasting for contextual recommendation. In *Proceedings of the 25th International Conference on World Wide Web.* 1407–1418.

[22] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. 2015. Learning hierarchical representation model for nextbasket recommendation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 403–412.

[23] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. 2018. Attention-based transactional context embedding for next-item recommendation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence.*

[24] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM, 729–732.