# Modeling Contemporaneous Basket Sequences with Twin Networks for Next-Item Recommendation

**Duc-Trong Le, Hady W. Lauw** and **Yuan Fang**
School of Information Systems, Singapore Management University, Singapore
{ductrong.le.2014, hadywlauw, yfang}@smu.edu.sg

## Abstract

Our interactions with an application frequently leave a heterogeneous and contemporaneous trail of actions and adoptions (e.g., clicks, bookmarks, purchases). Given a sequence of a particular type (e.g., purchases)– referred to as the target sequence, we seek to predict the next item expected to appear beyond this sequence. This task is known as next-item recommendation. We hypothesize two means for improvement. First, within each time step, a user may interact with multiple items (a basket), with potential latent associations among them. Second, predicting the next item in the target sequence may be helped by also learning from another supporting sequence (e.g., clicks). We develop three twin network structures modeling the generation of both target and support basket sequences. One based on "Siamese networks" facilitates full sharing of parameters between the two sequence types. The other two based on "fraternal networks" facilitate partial sharing of parameters. Experiments on real-world datasets show significant improvements upon baselines relying on one sequence type.

## 1 Introduction

In this era of digitization, most of our needs and wants are but a screen away. We shop at marketplaces such as Amazon or Alibaba; order meals from Uber Eats, stream music over Spotify or Pandora; get our screen time fix from Netflix or YouTube; etc. Consequently, some of us knowingly, while others unwittingly, are leaving our digital footprints, tracing the pages or activities where we have been. Some services, such as Foursquare, might in some cases even be able to approximate literal (walking) footprints based on check-ins.

Importantly, these traces from the past may well contain prescient signals of where we are headed in the future, in terms of our adoptions or consumptions. Hence, an important problem of wide interest and implication in both industry and academia is that of *next-item recommendation*. Based on historical data of consumers' activities, we would like to predict a new item that a consumer will likely adopt next.

Broadly, there are several main directions in the literature. One direction is collaborative filtering, whereby recommendations are driven by users' personalized preferences [Koren *et al.*, 2009]. Another direction is content-based recommendation, whereby recommendations are driven by similarity in content among items [Pazzani and Billsus, 2007]. Yet another direction is sequential preference, whereby recommendations are driven by latent dependencies between the next item and other items that a user has adopted at previous occasions.

**Problem.** We focus on *sequential preference*. There are scenarios where future actions are influenced by past actions. For one example, music streaming services are interested in generating coherent playlists, which requires paying attention to sequential transitions between songs [Chen *et al.*, 2012]. For another, the topics that Tweeters post tend to exhibit a sequential nature [Li *et al.*, 2016]. So is recommending courses, where there are progression over time and precedence relationships [Parameswaran *et al.*, 2010]. As each user is commonly associated with a sequence, the essence of this paradigm is learning sequentiality among items across users' sequences, rather than personalization per se.

Recent works on sequential recommendation are based on Recurrent Neural Networks (RNN) [Lipton *et al.*, 2015] (see Section 2). However, direct application of RNN to sequential recommendation suffers from two major limitations in modeling choices. First, it models a sequence of one type of actions (e.g., only purchases). Second, it assumes that at each time step, there is only one action (e.g., one item purchased). However, these assumptions may not bear out in some scenarios. For one, there are multiple types of actions resulting from user interaction with a system. In an online marketplace, a user may click on various items under consideration, abandon most, add some to a shopping cart, and puts others on a wish list, before an eventual purchase takes place. In a video streaming service, a user may watch some trailers, follow through to watch some shows fully, and later on may rate or review some movies, of which a few might be rated highly. In each case, we are dealing with multiple sequence types (e.g., sequence of clicks and sequence of purchases). Importantly, these sequence types are *contemporaneous*, occurring within a common period of time, and may well be capturing some related underlying behaviors. For instance, to predict what one would purchase, it may be instructive to pay attention to not only what a user has purchased previously, but also what she has clicked in the past. Therefore, we postulate the need for modeling these contemporaneous sequences jointly.

For another, we are not always dealing with a strict ordering of individual items. More frequently, we deal with groups or sessions, whereby there may be sequentiality from one session to another, but the ordering within a session may not be informative. For example, when planning travel, one day we may be searching for airfare, while on another day we may be booking accommodations. When grocery shopping, we may buy for different meal plans on different days. Though not necessarily sequentially ordered, items within a session are probably correlated to some degree, e.g., items of the same meal plan. We refer to such a group or session as "basket".

**Approach.** To address those limitations, we propose to model *contemporaneous basket sequences*. In this work, we focus on a pair[1] of sequence types: target and support. The *target* sequence refers to high-quality, high-value, and possibly sparser interactions (e.g., purchases) for which we wish to predict the next interaction (e.g., next purchase). The *support* sequence refers to more frequent and informative interactions (e.g., clicks) that would be relevant for predicting the next target item. For example, if purchasing is the target, and clicking is the support, then we are predicting the next purchase by modeling sequence of purchases and sequence of clicks.

We explore dual-RNN structure to represent the two sequence types. Having been generated contemporaneously from the same ecosystem of interactions, the sequence types likely model related phenomena. Instead of two completely different RNN's, we base our *Contemporaneous Basket Sequences* or CBS framework on the concept of *twin networks*. Analogously to biological twins, they share some commonalities, but to different degrees in different cases. We develop three CBS architectures along the spectrum of commonalities. In all, the two sequence types share a basket encoder to capture in-basket associations among items. They vary in how much sharing occurs at the recurrent units. For CBS-SN (Siamese Networks), the sequence types share a recurrent encoder. For CBS-CFN (Concordant Fraternal Networks), they each have a different recurrent encoder with the same recurrent units. For CBS-DFN (Discordant Fraternal Networks), one sequence type has a recurrent encoder and the other does not, to model different scopes of sequential effects.

**Contributions.** As our *first* contribution, we hypothesize that modeling contemporaneous basket sequences could be beneficial for next-item recommendation due to synergies between the target and support sequences. As our *second* contribution, we develop three neural network architectures: CBS-SN, CBS-CFN and CBS-DFN, describe their design in Section 3, and note some learning details in Section 4.1. Our *third* contribution is to investigate research questions on the effectiveness of modeling contemporaneous basket sequences jointly on public datasets (see Section 4).

## 2 Related Work

Our key contribution is modeling a *pair* of contemporaneous sequences, while factoring in the *basket*-correlation among items. Most of the previous works in modeling sequential preferences are preoccupied with only *one* sequence type, and that sequence consists of *individual* items (not baskets). The implication is that such works essentially model the sequence of items *within* a session [Zhang *et al.*, 2014; Hidasi *et al.*, 2016a], whereas we model sequences *across* sessions (each session is a basket). An orthogonal direction to ours is to incorporate features, such as text or images [Hidasi *et al.*, 2016b; Tuan and Phuong, 2017], or context such as time, location, weather [Liu *et al.*, 2016]. [Wu *et al.*, 2017] inferred the recurrent recommender model via fitting concurrently items' and users' rating sequences. Recently, [Xu *et al.*, 2018; Tang and Wang, 2018] tackled the personalized sequential recommendation task using memory networks and convolutional sequence embeddings respectively. They are not comparable to our work, as they model neither baskets, nor contemporaneous sequences.

Correlative association among items is a subject of interest in some previous works. These include association among items within a basket [Li *et al.*, 2009] as well as across baskets [Le *et al.*, 2017]. [Liang *et al.*, 2016] relied on item co-occurrences as a form of regularization for matrix factorization. [Xiang *et al.*, 2010] used a graph-based method to capture the effects of recent items. [Zhu *et al.*, 2014] sought to recommend not the next item, but the next bundle of items. Our distinction from these works is our focus on modeling not individual baskets or groups, but rather *sequences* of baskets.

In that respect, there have been some efforts in modeling sequences and basket-level associations concurrently. However, these works only focus on one sequence type, while our orientation is in investigating the effects of both support sequence and target sequence. [Rendle *et al.*, 2010] was based on factorizing transition probabilities or first-order Markov chains. [Wang *et al.*, 2015] proposed an aggregative strategy to learn latent representation of baskets. [Yu *et al.*, 2016] addressed the task using a long-term sequential model with the presence of users. In Section 4, we will compare to some of these baselines having one sequence type to investigate the effects of modeling a pair of sequence types.

There are instances where "twin" networks are applied for purposes other than next-item recommendation. These applications include question answering [Das *et al.*, 2016], text similarity [Neculoiu *et al.*, 2016], and image matching [Koch *et al.*, 2015]. In such cases, the two distinct inputs are assumed to have largely similar meanings. In our case, the relationship may be asymmetric, with one being the target sequence (to predict its next item), and the other the support sequence (to assist in predicting the target sequence). For another instance, in neural machine translation [Bahdanau *et al.*, 2015; Sutskever *et al.*, 2014], the objective is to "transform" a sequence in one language to another language. In our case, the objective is not transformation, but rather predicting the next item in the target sequence.

## 3 Contemporaneous Basket Sequences (CBS) with Twin Networks

Here, we describe the framework for contemporaneous basket sequences. We begin with the notations and problem formulation, before elaborating the proposed architectures.

[1] While the fundamentals of the proposed modeling would allow further extensions beyond two sequence types, in this paper we discourse on only two sequence types for clarity of exposition.

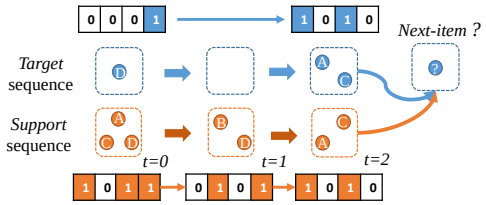


Figure 1: Example representations of target and support sequences

Let $V = \{v_0, v_1, ..., v_{N-1}\}$ denote the set of items under consideration. $B_t \subset V$ denotes a basket of items "adopted" at time step $t$. Adoption could mean purchasing, clicking, reviewing, or any other binary indication of preference. Another equivalent representation of a basket $B_t$ is a binary vector of length $N$, i.e., $X_t =< x_0, x_1, ..., x_{N-1} > \in \{0,1\}^N$, whereby $x_i = 1$ when an item $v_i \in B_t$, and 0 otherwise.

The data is a set of sequence pairs $D = \{\langle T_j, S_j \rangle\}_{j=0}^{M-1}$. For the $j^{th}$ instance, $T_j$ is its target sequence, while $S_j$ is its support sequence. Both $T_j$ and $S_j$ are represented as sequences of baskets/binary vectors $\{X_t\}$. For example, $D$ may concern $M$ users, whereby $T_j$ comprises the sequence of baskets purchased by user $j$, and $S_j$ comprises her sequence of clicks. For ease of illustration, and without loss of generality, subsequently we may use "purchase" or "target" interchangeably, as well as "click" or "support" interchangeably. Generally, the objective is to learn a model that uses information from both target and support sequences to predict the next item in the target sequence. The two sequence types are *contemporaneous*, i.e., occurring over a common time period. More precisely, the time period covered by $T_j$ overlaps with $S_j$, but the last basket of $S_j$ does not occur later than the next-item to be predicted for $T_j$, to avoid using future information to predict a past event. Figure 1 illustrates one instance of a pair of sequence types for four items $A$ to $D$. In the first time step, the user "clicks" on $A$, $C$, $D$ (represented as $[1, 0, 1, 1]$), eventually "purchasing" $D$ (represented as $[0, 0, 0, 1]$). The subsequent time steps involve baskets of different items.

Since we are dealing with sequences, we build on the foundation of RNNs, known for its capacity for generating sequential data. However, since we need to model two sequence types simultaneously, we investigate dual-RNN architectures or *twin networks*. We develop three such architectures, which differ in the degree of parameter sharing between the two sequence types. Their etymologies are inspired by biological terms describing twins [Hoekstra *et al.*, 2007]. CBS-SN is named after "Siamese twins"or identical twins with 100% gene sharing, to signify how the two sequence types will be modeled by identical RNNs. CBS-CFN and CBS-DFN are named after "fraternal twins" that on average share 50% of their genes, to signify both similarities and differences between the sequence types. The diagrammatic illustrations of three models are shown in Figure 2.

### 3.1 CBS with Siamese Networks (CBS-SN)

Our first model CBS-SN is based on the idea of Siamese networks. This structure contains twin networks that receive two distinct inputs, have their parameters tied so as to constrain the two inputs to the same feature space, and are conjoined together (concatenated) at the top layer [Bromley *et al.*, 1994]. The specific realization depends on the problem scenario.

Figure 2(a) illustrates the architecture of CBS-SN. We describe it layer by layer. The bottom layer is the *basket encoder*. In each time step, we have a basket/binary vector $X_t$. We hypothesize that there are correlated items that may co-occur within baskets. To capture the correlative information, we utilize a dense (fully connected) layer to map a basket's binary vector $X_t$ into its hidden representation $b_t$ as follows:

$$b_t = f(\Theta_b X_t + \Omega_b) \tag{1}$$

where $f$ is an activation function, $L$ is the number of latent dimensions in the dense layer; and $\Theta_b \in \mathbb{R}^{L \times N}, \Omega_b \in \mathbb{R}^L$; are parameters to be learned.

The middle layer is the *recurrent encoder* based on LSTM. It seeks to capture the sequential effect by feeding the basket representation $b_t$ into a recurrent layer. The hidden recurrent representation $h_t$ at the time step $t$ is computed as follows:

$$h_t = g(\Phi_b b_t + \Phi_h h_{t-1} + \Omega_h) \tag{2}$$

where $g$ is an activation function, $H$ is the number of hidden recurrent units; and $\Phi_b \in \mathbb{R}^{H \times L}, \Phi_h \in \mathbb{R}^{H \times H}, \Omega_h \in \mathbb{R}^H$; are parameters to be learned.

The final layer is the *aggregation layer*. The assumption of CBS-SN is that the target sequence and the support sequence are distinct manifestations of the same underlying phenomenon. Therefore, the two sequence types share the same basket encoder and LSTM recurrent encoder. Let $\hat{h}_T$ be the last hidden recurrent representation for the target sequence, and correspondingly $\hat{h}_S$ for the support sequence. In this Siamese networks-inspired structure, the aggregated representation is as follows:

$$\hat{h} = concat(\hat{h}_T, \hat{h}_S)$$
$$r_{\text{agg}} = W.\hat{h} + \Omega_c \tag{3}$$

$W \in \mathbb{R}^{N \times 2H}, \Omega_c \in \mathbb{R}^N$ are parameters to be learned. The scores of items for the next item recommendation task are computed as a function of this aggregated representation:

$$Y = \sigma(r_{\text{agg}}) \tag{4}$$

where $Y \in \mathbb{R}^{1 \times N}$, the *softmax* function $\sigma(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}}$.

The output is the vector $Y$ of length $N$, where each element is the likelihood of each item to be the next adoption.

### 3.2 CBS with Concordant Fraternal Networks (CBS-CFN)

The earlier assumption that the sequence types reflect the same underlying phenomenon may be too strong in some cases. For instance, purchases and clicks are related, in that some clicks lead to purchases. However, clicking or browsing actions are low-cost and easy to undo, as opposed to purchases that require a larger commitment of resources. Therefore, they may reflect different sequential behaviors.

As illustrated in Figure 2(b), our second model CBS-CFN leverages on two distinct recurrent encoders: *LSTM Layer 1*

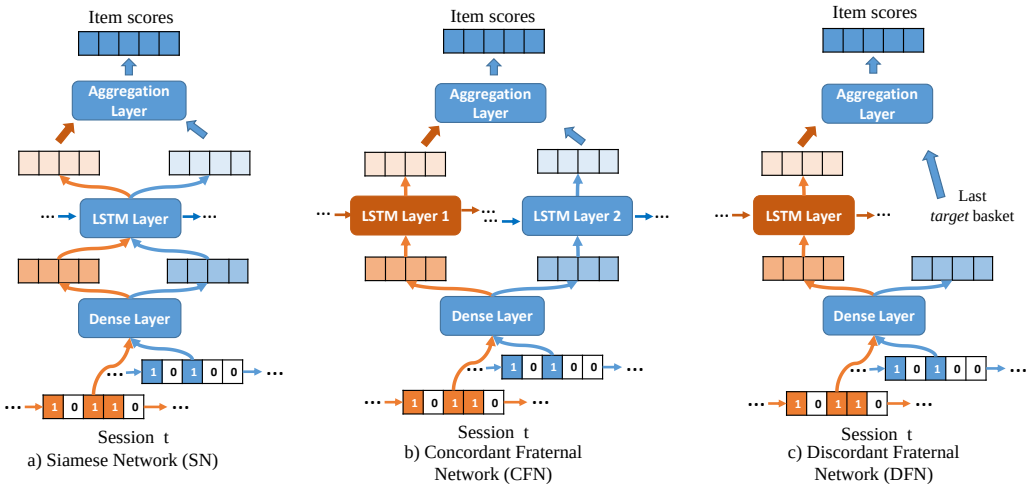


Figure 2: Modeling Contemporaneous Basket Sequences with Twin Networks

for the support sequence and *LSTM Layer 2* for the target sequence. They still share the same basket encoder, as in-basket associations are likely to still be similar in both cases. From partial sharing of parameters, different recurrent encoders but same basket encoder, arises the notion of *fraternal* networks. The term *concordant* signifies the same size of LSTMs, capturing longer-term sequentiality in both sequence types.

Because CBS-CFN assumes the target sequence and support sequence are distinct, we would like to aggregate them in a way that allows their contributions to be weighted accordingly. The last hidden recurrent representations $\hat{h}_T$ and $\hat{h}_S$ are aggregated as follows:

$$r_{\text{agg}} = W_1.\hat{h}_S + W_2.\hat{h}_T + \Omega_c \quad (5)$$

where $W_1, W_2 \in \mathbb{R}^{N \times H}, \Omega_c \in \mathbb{R}^N$ are parameters to be learned. The output $Y$ is computed as in Equation 4.

### 3.3 CBS with Discordant Fraternal Networks (CBS-DFN)

The previous model seeks to capture distinct sequence types of the same sequential dependencies. In some scenarios, it may be appropriate to capture different scopes of sequential dependency. For instance, browsing and clicking may have longer-term dependency than purchases. This is reflected in our third model CBS-DFN, where the support sequence has a recurrent encoder to learn longer-term recurrence relations, but the target sequence relies on shorter-term relations and directly makes use of the output of the bottom layer (basket encoder). The term *discordant* refers to this varying treatment. The aggregated operation is defined as follows:

$$r_{\text{agg}} = W_1.\hat{h}_S + W_2\hat{b}_T + \Omega_c \quad (6)$$

where $\hat{b}_T$ is the hidden representation of the last basket, $W_1 \in \mathbb{R}^{N \times H}, W_2 \in \mathbb{R}^{N \times L}, \Omega_c \in \mathbb{R}^N$ are parameters to be learned. The output $Y$ is computed as in Equation 4.

| Dataset | | #Sequence | #Item | #Average Length | #Average Basket Size |
|---|---|---|---|---|---|
| Alibaba | Support | 23740 | 13498 | 11.2 | 5.5 |
| | Target | | | 5.3 | 1.8 |
| MovieLens | Support | 189858 | 8202 | 34.5 | 2.5 |
| | Target | | | 16.6 | 1.8 |

Table 1: Statistics for Alibaba, MovieLens

## 4 Experiments

We delve into several research questions on the utility of modeling longer sequences, as opposed to short-term dependencies; and the utility of modeling two contemporaneous sequence types, as opposed to relying on one sequence type.

### 4.1 Setup

**Datasets.** We experiment with two public real-life datasets of different domains. The statistic is summarized in the Table 1.

*Alibaba*[2]: Alibaba provided mobile shopping data for the period from 18/11/2014 to 18/12/2014. For each user, we construct her session sequence, where each session contains the items she adopted within a day. From session sequences, we generate contemporaneous basket sequences of the two adoption types: *click* as support and *purchase* as target.

*MovieLens*[3]: This is a popular rating dataset. Considering the last three years from 01/2006 to 01/2009, we build a session sequence for each user, where each session represents what movies she adopted in a specific day. Here, the two adoption types are *selecting a movie to rate* as support (akin to clicking), and *assigning a movie a high rating* as target (akin to purchasing). High rating is at least 4.5 out of 5.

**Preprocessing.** We filter out infrequent items, i.e., fewer than 50 clicks for Alibaba or 20 ratings for MovieLens.

[2]https://tianchi.aliyun.com/datalab/dataSet.htm?id=4
[3]https://grouplens.org/datasets/movielens/10m

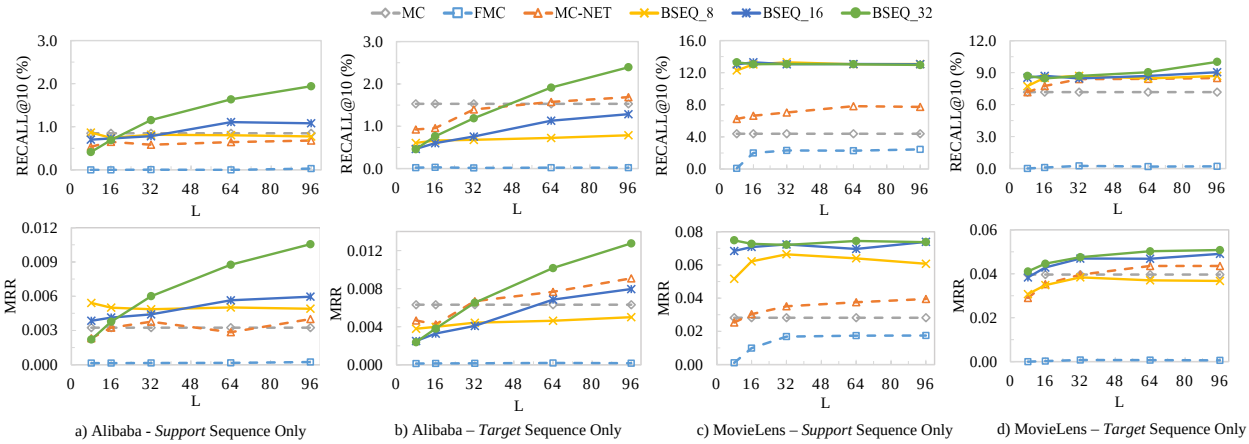


Figure 3: Performance Comparison of Next-item Recommendations using *Markov Chain* vs. *RNN* on Alibaba, MovieLens

Since the aim is to model sequences, we filter out sequences with less than 2 baskets. Sequences are separated chronologically by three non-overlapping periods, denoted as $(P_{\text{train}}, P_{\text{validate}}, P_{\text{test}})$. They are $(29, 1, 1)$ day(s) for Alibaba and $(31, 3, 3)$ months the for MovieLens. Following [Rendle *et al.*, 2010], we seek to recommend new items, and so ignore item candidates that have occurred in the most recent basket.

**Evaluation Task & Metrics.** The task is evaluated via top-$K$ recommendations. For each testing sequence pair $\langle S, T \rangle$, we hide the last $target$ basket $B$ to create $|B|$ testing instances with the ground-truth. We utilize two conventional metrics for top-$K$ recommendations. The first metric is recall (*Recall@K*), defined as the percentage of testing instances with the ground truth item in top$-K$. In experiments, we mainly rely on the top-10 recommendations, i.e., *Recall@10*, but will later show other top$-K$ performances as well. To evaluate the overall ranking performance, the second metric is Mean Reciprocal Rank (*MRR*), computed as follows:

$$MRR = \frac{\sum_B \sum_{v \in B} \frac{1}{\text{rank of } v \text{ for } (S, T \setminus B)}}{\#\text{total testing instances}} \quad (7)$$

We cut the recommendation list off at 200 because the rest contribute almost zero to *MRR*. The performances are averaged across 30 runs with different random initializations. Comparisons are supported by one-tailed paired-sample Student's $t$-test at 0.05 significance level.

**Learning Details.** To learn parameters, we seek to minimize the softmax-cross-entropy loss based the output of the Eq (4). All neural networks are trained in 20 epochs of batch-size 32 by the *Adam* optimizer with the learning rate 0.001. The dense layer use the *ReLu* activation function to only keep positive weights. In the recurrent layer, $LSTM$ unit is applied with a 0.3 dropout probability. Additionally, we also measure the *Recall@10* for both training and validating datasets. The performance on validation is used to decide whether to save learned models. After each epoch, its model is kept if the validation's accuracy is better than the previous epoch. Finally, the best model is used to generate top-$K$ prediction on the testing set.

## 4.2 Research Questions

**RQ1. Is modeling sequential data useful for next-item recommendation?** To focus on the effect of sequence itself, rather than the effect of joining contemporaneous sequences, we first create a single-sequence variant of CBS, which we call *Basket Sequences* or BSEQ. We compare it to another approach that models only short-term transitions based on the *Markov chain* (MC) property. The first baseline MC generates conditional probabilities of the next-item given the previous item. The probability of the next-item given the previous basket is the average over the transition probabilities from each basket item to the next item. The second baseline FMC factorizes the MC conditional probabilities to reduce the sparsity [Rendle *et al.*, 2010]. We use the *LibFM*[4] library to learn this model. The third baseline MC-NET is a simple neural network that feeds the last basket representation from the basket encoder to predict the next item. For each sequence type, we train MC, FMC, MC-NET, BSEQ with various latent dimensions $L \in \{8, 16, 32, 64, 96\}$. BSEQ is investigated with three settings of the hidden state size $H \in \{8, 16, 32\}$, resulting in BSEQ_8, BSEQ_16, and BSEQ_32.

On Alibaba, Figure 3(a) shows the performance of the models when using only the support sequence, while Figure 3(b) shows the same for the target sequence only. The three *Markov*-based models are not influenced much by various latent dimensions $L$, except MC-NET on the *target* sequence. Because we are predicting for the target sequence, it is reasonable that the *Markov*-based models perform better when learnt on the target sequence than the support sequence. Importantly, not only are the three variants of BSEQ more sensitive to different latent dimensions, but they also show better results given sufficient $L$. BSEQ_32 is the best variant with a consistent improvement trend, which verifies the presence of longer-term dependencies in Alibaba sequences.

Figures 3(c) and (d) demonstrate the performances on MovieLens. We can draw similar conclusions as before on the strength of the BSEQ variants over the *Markov* baselines.

[4] http://www.libfm.org

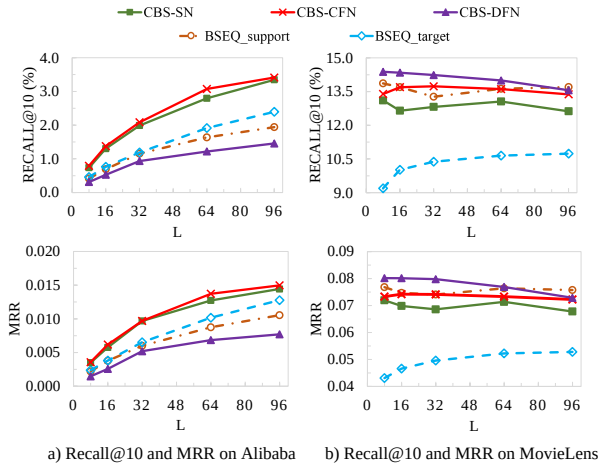


Figure 4: Performance Comparison of the BSEQ and CBS models on Alibaba, MovieLens.

| Dataset | Model | L | $MRR$ | $Recall@K$ (%) 10 | 20 | 50 |
|---|---|---|---|---|---|---|
| Alibaba | POP | - | 0.004 | 0.51 | 0.68 | 1.19 |
| | $DRM_{support}$ | 64 | 0.011 | 2.14 | 2.91 | 4.02 |
| | $DRM_{target}$ | 32 | 0.004 | 0.72 | 1.19 | 1.98 |
| | $BSEQ_{support}$ | 96 | 0.011 | 1.94 | 2.54 | 3.92 |
| | $BSEQ_{target}$ | 96 | 0.013 | 2.39 | 3.14 | 4.56 |
| | CBS-SN | 96 | 0.014 | 3.34 | 4.13 | 5.43 |
| | CBS-CFN | 96 | **0.015**$^{\ddagger\S}$ | **3.41**$^{\ddagger\S}$ | **4.36**$^{\ddagger\S}$ | **5.60**$^{\ddagger\S}$ |
| | CBS-DFN | 96 | 0.008 | 1.45 | 1.90 | 3.02 |
| MovieLens | POP | - | 0.006 | 1.79 | 2.89 | 6.58 |
| | $DRM_{support}$ | 96 | 0.002 | 0.37 | 0.71 | 1.53 |
| | $DRM_{target}$ | 16 | 0.001 | 0.20 | 0.36 | 0.77 |
| | $BSEQ_{support}$ | 8 | 0.075 | 13.87 | 18.65 | 28.50 |
| | $BSEQ_{target}$ | 64 | 0.050 | 10.65 | 15.55 | 25.95 |
| | CBS-SN | 8 | 0.070 | 13.11 | 17.66 | 27.67 |
| | CBS-CFN | 32 | 0.072 | 13.73 | 18.80 | **29.65**$^{\ddagger\S}$ |
| | CBS-DFN | 8 | **0.078**$^{\ddagger\S}$ | **14.38**$^{\ddagger\S}$ | **19.39**$^{\ddagger\S}$ | 29.33 |

Table 2: Best Performance Comparison on Alibaba, MovieLens. The symbols $\ddagger, \S$ denote the statistically significant improvements of our best model over the BSEQ and DRM models respectively

Interestingly, the gap between the two families is larger on the *support* than on the *target*. This indicates a stronger longer-term dependency on the *support* sequences, while the robustness of the *Markov* baselines on the *target* sequences implies greater effect of short-term transitions on MovieLens's target sequence. Overall, BSEQ_32 still shows the best performance, and will be used subsequently for future comparisons.

**RQ2. Is modeling contemporaneous sequences useful?** We consider two model families: the single-sequence BSEQ and the dual-sequence CBS. Both run under the same setting of the hidden state size ($H = 32$) and latent dimensions $L \in \{8, 16, 32, 64, 96\}$. In Figure 4(a), CBS-SN and CBS-CFN improve significantly upon the BSEQ models on Alibaba. Modeling contemporaneous basket sequences gives more information and supportive evidence by taking advantage of the long-term dependencies from both sequences types. The CBS-DFN model does not work as well, which could be explained by the loss of information from confining the target sequence only to the most recent basket. This short-term dependency is different to the actual sequential dependency reflecting on the target sequence. Therefore, it triggers in the disagreement in the aggregation layer.

Figure 4(b) illustrates the performance of the two model families on MovieLens dataset. The observations of the two BSEQ models are consistent with what we found in the previous experiments. The longer-term dependency is stronger on the *support* sequence and weaker in the *target* sequence. This is the appropriate scenario for CBS-DFN, which shows the biggest improvements over the BSEQ_support model.

Generally, the fraternal networks (CBS-CFN and CBS-DFN) tend to perform better than the Siamese networks (CBS-SN) because the former could more flexibly fit the two sequence types. Between CBS-CFN and CBS-DFN, CBS-CFN has an advantage when there is more long-term dependency in the target sequence whilst CBS-DFN has an advantage when there is more short-term dependency in the target sequence.

**RQ3. How does the proposed CBS models perform against other baselines?** We summarize the best performance of our proposed models as compared to baselines in Table 2. POP recommends items based on popularity. DRM is the recently proposed dynamic recurrent model [Yu *et al.*, 2016], which is a state-of-the-art baseline capable of modeling basket sequences of a single type. By design, it is limited to fixing the same number of latent dimensions and hidden state size ($H = L$). We consider the same setting $H = 32$ for BSEQ and CBS. For BSEQ, CBS, and DRM, we tune $L \in \{8, 16, 32, 64, 96\}$ for their respective best performance and indicate the chosen $L$ in Table 2. For Alibaba, CBS-CFN significantly outperforms the baselines as well as CBS-SN, implying while contemporaneous sequences are useful, the model benefits from giving the sequence types some flexibility in the recurrent layers. For MovieLens, CBS-DFN is the best-performing model. The performance of DRM is low, possibly due to local optima. The outperformances by CBS-CFN on Alibaba and by CBS-DFN on MovieLens against the BSEQ and DRM are statistically significant.

## 5 Conclusion

In this paper, we address the next-item recommendation by modeling contemporaneous basket sequences. We introduce three architectures based on twin networks, which vary in the degree of similarity or parameter sharing across the two sequence types. Experiments show that there is utility to modeling sequential data, with two sequence types better than one. The two sequence types may benefit from some flexibility in their parameters and size, as supported by the good performance of the fraternal variants over the Siamese variant.

## Acknowledgments

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its NRF Fellowship Programme (Award No. NRF-NRFF2016-07).

## References


[Bahdanau *et al.*, 2015] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

[Bromley *et al.*, 1994] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a" siamese" time delay neural network. In *NIPS*, pages 737–744, 1994.

[Chen *et al.*, 2012] Shuo Chen, Josh L Moore, Douglas Turnbull, and Thorsten Joachims. Playlist prediction via metric embedding. In *KDD*, pages 714–722, 2012.

[Das *et al.*, 2016] Arpita Das, Harish Yenala, Manoj Chinnakotla, and Manish Shrivastava. Together we stand: Siamese networks for similar question retrieval. In *ACL*, volume 1, pages 378–387, 2016.

[Hidasi *et al.*, 2016a] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *ICLR*, 2016.

[Hidasi *et al.*, 2016b] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Recsys*, pages 241–248, 2016.

[Hoekstra *et al.*, 2007] Chantal Hoekstra, Zhen Zhen Zhao, Cornelius B Lambalk, Gonneke Willemsen, Nicholas G Martin, Dorret I Boomsma, and Grant W Montgomery. Dizygotic twinning. *Human reproduction update*, 14(1):37–47, 2007.

[Koch *et al.*, 2015] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015.

[Koren *et al.*, 2009] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[Le *et al.*, 2017] Duc-Trong Le, Hady Wirawan LAUW, and Yuan Fang. Basket-sensitive personalized item recommendation. In *IJCAI*, pages 2060–2066, 2017.

[Li *et al.*, 2009] Ming Li, Benjamin M Dias, Ian Jarman, Wael El-Deredy, and Paulo JG Lisboa. Grocery shopping recommendations based on basket-sensitive random walk. In *SIGKDD*, pages 1215–1224, 2009.

[Li *et al.*, 2016] Yang Li, Ting Liu, Jing Jiang, and Liang Zhang. Hashtag recommendation with topical attention-based lstm. In *COLING*, pages 3019–3029, 2016.

[Liang *et al.*, 2016] Dawen Liang, Jaan Altosaar, Laurent Charlin, and David M. Blei. Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In *Recsys*, pages 59–66, 2016.

[Lipton *et al.*, 2015] Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv:1506.00019*, 2015.

[Liu *et al.*, 2016] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. Context-aware sequential recommendation. In *ICDM*, pages 1053–1058, 2016.

[Neculoiu *et al.*, 2016] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *Workshop on Representation Learning for NLP*, pages 148–157, 2016.

[Parameswaran *et al.*, 2010] Aditya G Parameswaran, Georgia Koutrika, Benjamin Bercovitz, and Hector Garcia-Molina. Recsplorer: recommendation algorithms based on precedence mining. In *SIGMOD*, pages 87–98, 2010.

[Pazzani and Billsus, 2007] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. 2007.

[Rendle *et al.*, 2010] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820, 2010.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Tang and Wang, 2018] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, 2018.

[Tuan and Phuong, 2017] Trinh Xuan Tuan and Tu Minh Phuong. 3d convolutional networks for session-based recommendation with content features. In *Recsys*, pages 138–146, 2017.

[Wang *et al.*, 2015] Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*, pages 403–412, 2015.

[Wu *et al.*, 2017] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *WSDM*, pages 495–503, 2017.

[Xiang *et al.*, 2010] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. Temporal recommendation on graphs via long-and short-term preference fusion. In *KDD*, pages 723–732, 2010.

[Xu *et al.*, 2018] Chen Xu, Xu Hongteng, Zhang Yongfeng, Tang Jiaxi, Cao Yixin, Qin Zheng, and Zha Hongyuan. Sequential recommendation with user memory networks. In *WSDM*, 2018.

[Yu *et al.*, 2016] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. A dynamic recurrent model for next basket recommendation. In *SIGIR*, pages 729–732, 2016.

[Zhang *et al.*, 2014] Yuyu Zhang, Hanjun Dai, Chang Xu, Jun Feng, Taifeng Wang, Jiang Bian, Bin Wang, and Tie-Yan Liu. Sequential click prediction for sponsored search with recurrent neural networks. In *AAAI*, pages 1369–1375, 2014.

[Zhu *et al.*, 2014] Tao Zhu, Patrick Harrington, Junjun Li, and Lei Tang. Bundle recommendation in ecommerce. In *SIGIR*, pages 657–666, 2014.