applied sciences

MDPI

Article

An Attention-Based Recommender System to Predict Contextual Intent Based on Choice Histories across and within Sessions

Ruo Huang [1], Shelby McIntyre [2], Meina Song [1,*], Haihong E [1] and Zhonghong Ou [1]

[1] School of Computer Science, Beijing University of Posts & Telecommunications, Beijing 100876, China; charleshuangruo@bupt.edu.cn (R.H.); ehaihong@bupt.edu.cn (H.E.); zhonghong.ou@bupt.edu.cn (Z.O.)
[2] Leavey School of Business, Santa Clara University, Santa Clara, CA 95053, USA; smcintyre@scu.edu
* Correspondence: mnsong@bupt.edu.cn; Tel.: +86-139-1102-3433

Received: 31 October 2018; Accepted: 26 November 2018; Published: 29 November 2018

Abstract: Recent years have witnessed the growth of recommender systems, with the help of deep learning techniques. Recurrent Neural Networks (RNNs) play an increasingly vital role in various session-based recommender systems, since they use the user's sequential history to build a comprehensive user profile, which helps improve the recommendation. However, a problem arises regarding how to be aware of the variation in the user's contextual preference, especially the short-term intent in the near future, and make the best use of it to produce a precise recommendation at the start of a session. We propose a novel approach named Attention-based Short-term and Long-term Model (ASLM), to improve the next-item recommendation, by using an attention-based RNNs integrating both the user's short-term intent and the long-term preference at the same time with a two-layer network. The experimental study on three real-world datasets and two sub-datasets demonstrates that, compared with other state-of-the-art methods, the proposed approach can significantly improve the next-item recommendation, especially at the start of sessions. As a result, our proposed approach is capable of coping with the cold-start problem at the beginning of each session.

Keywords: session-based recommender system; attention mechanism; contextual user intent; recurrent neural network

1. Introduction

In recent years, we have witnessed the rapid growth of neural networks for a variety of applications in computer vision, natural language processing, speech recognition, etc. Neural network provides four key benefits: (1) It works with unlabeled data; (2) It learns low-level features from minimally processed raw data; (3) It detects complex interactions among features; and (4) It works with large class memberships [1,2]. Recent achievements have shown that it is also of great use to combine different neural networks into recommender systems. There are many works which employed neural networks to improve the recommendation in the domain of recommender systems [3–7] and in the context of web-based decisions [8–12]. The success of these recommendation approaches based on neural networks demonstrates its capability.

Session-based recommender systems are good examples. Although not a novel research topic, session-based recommender systems are largely under-investigated [3,4,13]. Compared with traditional recommendation methods, a session-based recommender system is more appropriate for capturing dynamic and sequential user behavior. The Recurrent Neural Network (RNN) plays an important role in session modeling, and it seems to have promise for further improvement. An RNN, compared to many other recommendation models, takes advantage of the ordered sequence in a very natural way [3,5,7].

Automated and personalized recommendations of the type "You may also be interested in" are common on e-commerce sites nowadays, and there exists sufficient evidence that such item related recommendations can measurably impact businesses [14–19]. Undoubtedly, one of the representative business scenarios is in e-commerce. Modern online shops are not static catalogs of items anymore but tend to fulfil the user's interests by providing personalized product recommendations. In the best case, these recommendations should match both the users' long-term preference as well as their current shopping goals. On Amazon.com, for example, each product page contains multiple personalized recommendation lists with different purposes. Though the Amazon recommender system employs collaborative filtering approach [20], it also uses solo-user recommendations as well. It provides alternatives to the product currently viewed, remind the user of items recently viewed, or render recommendations that should appeal to the general taste of the user, as shown in Figure 1.

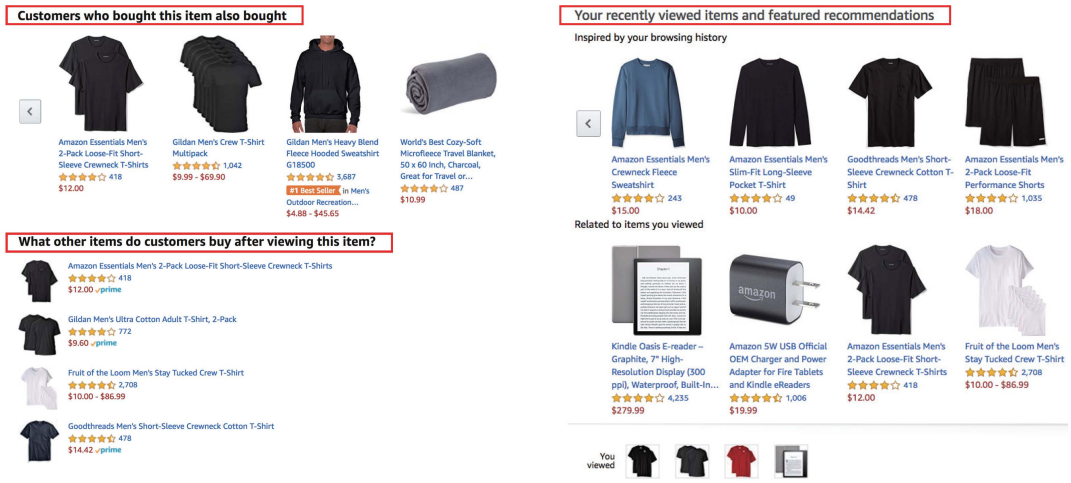

Figure 1. An example of Amazon recommender system.

As a result, the content displayed on the page not only depends on the features of the currently viewed article (i.e., the product category), but can also be influenced by an integration of the user's historical shopping behavior (long-term preference) and his most recent interactions of viewing different products (short-term shopping goals). In this case, we assume that user's preference is composed of two components: (1) the long-term preference which reflects the considerably stable interests of the users based on their online activities, and (2) the short-term intent which represents the users' current interests. There are many existing techniques that investigate the combination of the user's long-term preference and the short-term intent [13,21–27]. In addition, the user's preference for items constantly evolves over time [28], while these works implicitly assume the user preference to be stationary, which seems to be an unrealistic assumption in many scenarios, particularly in the news or shopping related scenarios [5–7,13,29]. Therefore, such methods perform poorly when preferences, in fact, are context-sensitive and transient. There are session-based approaches [7,29–32] that focus on the interest drifts, to solve the problem of the static user preference and intent, while these methods assume that each item in one session has the same influence, which is usually an invalid assumption, especially when considering a user's interactions with items of diverse characteristics. One of the main attractions of search-engine advertising is that search engines take into account the "current interest" of the user and thereby can deliver the right message, ad, or product just at the right time. The enormous growth in spending going to search advertising (about $100b in the US in 2018 [33]) is a testament to the power of contextual "interest-based" advertising. To deal with the problem of the static weights being assigned to each item, researchers proposed several approaches [34–37]. Our proposed model goes a step further, not only taking the user's interest drift into consideration, but also regarding each item in one session as "the one and only" item, which means each one has a unique influence on the user and therefore the user may pay different attention to each one.

In this work, we propose a novel approach named Attention-based Short-term and Long-term Model (ASLM), to solve the next-item recommendation problem. It has been shown that the attention mechanism is able to assign different weights according to the importance of the items, and there have been works that employed the attention mechanism to form the characteristics of the user [38–40]. In order to capture the variation in user preference and generate the precise contextual user profile, we build two different RNN-based layers: one attentive layer to model the user's short-term intent, and the other one to model his long-term preference. The main contributions of this paper include:

- We introduce the attention mechanism to investigate the variation in a user's short-term intent for session-based recommendations.
- Taking the user's previous sessions into account, we model the user's long-term preference and concatenate that with short-term intent to account for more comprehensive contextual user outcomes.
- We conduct empirical evaluations and validate the effectiveness of the model on three real-world datasets and two sub-datasets.
- We demonstrate the effectiveness of the approach in resolving the cold-start problem.

2. Related Work

Collaborative Filtering (CF) is the most commonly used and studied technology [20,41]. CF allows users to give ratings about a set of elements (e.g., videos, songs, films, etc. in a CF-based website) in such a way that when enough information is stored on the system, we can make recommendations to each user based on information provided by other users we consider to have the most in common with them [42]. The most commonly used algorithm for CF is the k Nearest Neighbors (kNN) [41,43,44], which offers several mechanisms to identify similar users and pool recommendations. The other main approach for CF is matrix factorization (MF) [42,45], which regards recommendation as a dimensionality reduction problem. However, these methods are far from researchers' satisfaction with several major challenges:

- Resulting from the high level of sparsity [46,47] in recommender system databases, user similarity measures often encounter processing and cold-start problems from inadequate mutual ratings for a comparison of users and items [48–50].
- The CF-based recommendation lacks diversity, because it only recommends popular items from similar users but cannot recommend unique tastes of a user.
- When numbers of existing users and items grow tremendously, traditional CF algorithms will suffer serious scalability problems, with computational resources going beyond practical or acceptable levels [45].
- Most importantly, they are not qualified to capture the temporal context and sequential patterns of user behavior (e.g., the evolving user long-term preference and short-term intent) [51]. In other words, when given the data which does not have ratings involved but pure user-item interaction (or choice) logs, CF-based methods may not work.

To solve the problem of dealing with the sequential patterns of user behavior and temporal aspects of recommendations, there are a variety of works that focus on session-based recommendation which features sequence modeling. In academia, sequential recommendation problems are typically regarded as the task of predicting the next user action. Experimental evaluations are usually based on larger, time-ordered logs of user actions (e.g., the users' item viewing and purchasing activities on an e-commerce shop), which contains both the implicit and explicit feedback from users and provides the recommender system the capability of capturing the sequential patterns of users. Zimdars et al. [25] framed CF as a sequence prediction problem and used a simple Markov model for web-page recommendation. Mobasher et al. [26] proposed a similar approach, using sequential pattern mining. Both showed the superiority of methods based on sequence over nearest-neighbor approaches.

Park et al. [27] presented a modified User-based CF method, called Session-Based CF (SSCF), that uses information from similar sessions to capture sequence and repetitiveness in the listening process. Most recently, researchers have successfully adopted deep learning techniques in recommender systems. In particular, RNN, which is capable of learning models from sequentially ordered data, is a "natural choice" for sequence modeling [4]. Hidasi et al. [13] first introduced the idea of using RNN for session-based recommendation to deal with sparsity problems and proposed the GRU4REC architecture. Tan et al. [21] improved the model performance proposed in [13] by proposing the data augmentation and the method to account for shifts in the input data distribution. Jannach et al. [22] also improved the model proposed in [13] by proposing a model combining the kNN approach with GRU4REC. Dong et al. [23] improved the RNNs [52] and the MF model [53] by combining them in a multi-task learning framework, where they performed joint optimization with shared model parameters enforcing the two parts to regularize each other. Liang et al. [24] employed RNN to extract the item and user profiles from user-defined items' tags and their tagging behaviors. However, these session-based methods regard user preference for items as a stationary property and therefore cannot capture the changing contextual interactions between user and items.

To solve the problem of the static user preference and intent, Jing et al. [30] focused on the variation in user intent at different times and proposed an LSTM-based "Just-In-Time" approach to recommend the right item at the right time. Wu et al. [31] proposed an LSTM-based approach that implicitly captures various known temporal patterns in movie ratings data without explicit inclusion in the model, to learn dynamic embeddings of users and movies. The next-basket recommendation model proposed in [32] learned the varying representation of a user and captured global sequential features among baskets to reflect the user's dynamic intent at different times and interactions of all baskets of the user over time. Ruocco et al. [7] improved the next-item recommendation and cold-start problem by implementing two RNNs to extract user preference from recent sessions and the current session, respectively. Hu et al. [29] designed an efficient personalized session-based recommender system with shallow wide-in-wide-out networks over relaxed ordered user-session contexts. These methods assume that each item in one session has the same influence. However, this assumption is usually invalid, especially when considering a user's interactions with items of diverse characteristics. In the meantime, not only his preference for items changes among different sessions, but also his intent on items within one session.

To deal with the problem of the static weights being assigned to each item, Cheng et al. [34] developed an attention-based model to capture the changing attention that a user pays to different text reviews and ratings. Wang et al. [35] designed an attention-based transaction embedding model to weight each item in a transaction without assuming order. To solve these two issues at the same time in a unified way, we follow the idea in [7] but make the following contributions: (1) We propose an Attention-based Short-term Intent Layer to assign different weights for items in the current session to capture the contextual property of user intent. (2) We propose a Long-term Preference Layer to take the user's previous sessions into account, to provide information to the attention-based short-term intent layer and enable it to combine both the user's short-term intent and long-term preference in the attention process.

3. The ASLM Architecture

In this section, we first formulize the description of the problem as explained above and the main idea of Attention-based Short-term and Long-term Model (ASLM) proposed in this paper. Afterwards, we elaborate by diagram the architecture of ASLM.

3.1. Problem Formulation

Let $\mathcal{U}$ denote a set of users and $\mathcal{V}$ denote a set of items, where $|\mathcal{U}|$ and $|\mathcal{V}|$ are the total numbers of users and items, respectively. In this work, we concentrate on extracting the user contextual intent and preference from sequential user-item interaction events (e.g., commenting to a thread or listening

to music). For each user $u \in \mathcal{U}$, the sequential sessions are denoted as $S^u = \left\{S_1^u, S_2^u, \ldots, S_T^u\right\}$, where T is the total number of time steps and $S_t^u \subseteq S^u (t \in [1, T])$ represents the item set related to the interaction of user u at time step t. In each session, there is a set of events $\{e_{t,i}^u \in \mathbb{R}^m | i = 1, 2, \ldots, |S_t^u|\}$, where $e_{t,i}^u$ represents the event i in the session S_t^u. For each event, the user u interacts with an item $v_i \in \mathcal{V}$. Both S_t^u and $e_{t,i}^u \in S_t^u$ are ordered by time sequence. For a time step t, the session S_t^u implies the u's short-term intent at time t, while the sessions before time step t, defined as $L_{t-1}^u = S_1^u \cup S_2^u \cup \ldots \cup S_{t-1}^u$, implies user u's long-term preference. As in [7], our task is to predict each successive item in a session S_t^u. That is, for a sub-session $\{e_{t,1}^u, e_{t,2}^u, \ldots, e_{t,i}^u, \ldots, e_{t,n}^u\}$ of S_t^u, the system is to predict $e_{t,i+1}^u$. This is repeated for $i = 1, 2, \ldots, |S_t^u| - 1$. A recommendation R_i is a rank ordered list of k recommended items. The goal of the system is to make the next item, $e_{t,i+1}^u$, be located as close to the top of the ranked list as possible.

3.2. Main Idea

Consider the following example of a user shopping scenario: On May 15th, John purchased a printer and a smartphone. On July 10th, when the ink cartridge ran out, he wanted another to replace it. At that point in time, he would have been happy to see an ad for any ink cartridge and would have bought another brand if the system had recommended it. Also, since John was in the purchasing process, he might also have been attracted to a cross-sell recommendation of a phone-protector case. However, he would have been much less interested, in July, about recommendations for another brand-new printer or smartphone.

In the ASLM model, there are two layers: the Long-term Preference Layer and the Attention-based Short-term Intent Layer. Our model is illustrated in Figure 2, where two mutually exclusive methods are shown in one figure to conserve space: (a) ASLM-AP, and (b) ASLM-LHS. In other words, only one method is used separately at a time, and has no connection between each other. The two ASLM models only differ by how they feed information to the Attention-based Short-term Intent Layer, either using average pooling (ASLM-AP) or the last hidden state (ASLM-LHS). Based on the experiment results discussed in Section 5, researchers can feel free to use either of these two models since their performances are at the same level.

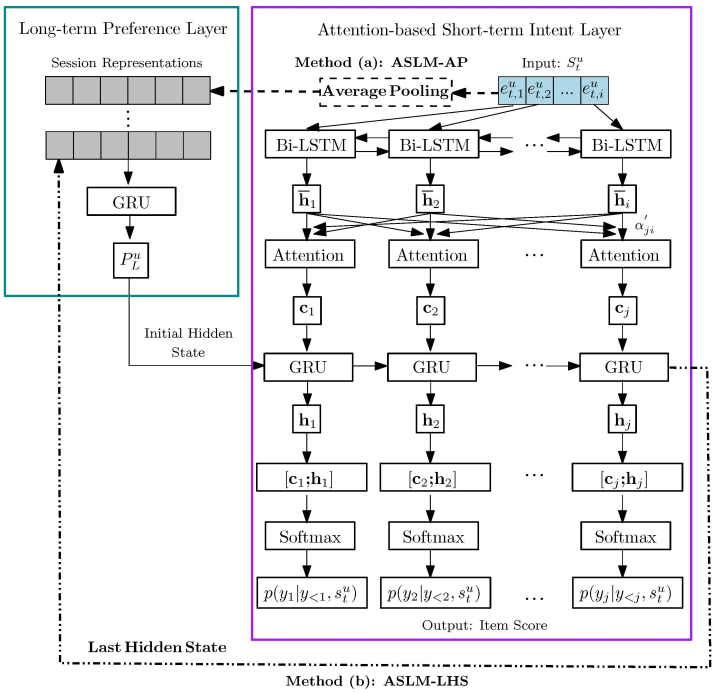

Figure 2. ASLM architecture. Two mutually exclusive methods are shown: (**a**) ASLM-AP, and (**b**) ASLM-LHS. Only one method is used at a time.

The task of the Long-term Preference Layer is to supply knowledge of the user's long-term preference to the Attention-based Short-term Intent Layer before the short-term intent layer investigates the current session at the beginning. The task of the Attention-based Short-term Intent Layer is to produce recommendations by taking the user's long-term preference into consideration when the short-term layer is processing the sequence of items in the current session to extract his short-term intent.

We discuss the work flow of these two layers in detail in Section 3.3. Particularly, the Attention-based Short-term Intent Layer's procedure of capturing the characteristics of each step in the user behavior is thoroughly discussed at the part of decoder in Section 3.3.2. Our model is proposed according to the following characteristics of user preference. (1) User preference changes at different time steps. (2) Within one time step, the user may have another intent on a different category of items, which has a different impact on the future interaction of items. (3) Since there are different preferences across users, the importance of the same item may differ, which results in different influences on the prediction of the next item for different individuals. The main idea of our approach is to:

(1) use the GRU (Gated Recurrent Unit) [13] in the Long-term Preference Layer to process a user's recent sessions, i.e., a user's long-term preference, which is similar to the Inter-session RNN in [7].
(2) enhance the recommendations by employing an augmented attention-based encoder-decoder in a structured framework named Attention-based Short-term Intent Layer. We use the encoder to process the events within a session, i.e., a user's short-term intent, and make the decoder receive the user's long-term preference representation P_L^u from the user's history session representations as its initial hidden state. As a result, the model enables the Attention-based Short-term Intent Layer to combine both the user's long-term preference and short-term intent in the attention process. Eventually, for ASLM-LHS, the decoder stores its output state in session representations for the Long-term Preference Layer to update the states for future recommendation. For ASLM-AP, we put the average of current session's embedded vector representations of the items into session representations.

3.3. Model Description

In this section, we discuss the ASLM model's entire recommendation process and the main system components with the adopted technologies we introduced.

As illustrated in Figure 2, ASLM integrates the modeling of the user's Long-term and augmented Attention-based Short-term Intent Layers into one single architecture. The Long-term Preference Layer is evolved from the Inter-session RNN used in [7], and the model proposed in it is regarded as a baseline to compare with ASLM model.

3.3.1. Long-Term Preference Layer

For the short-term intent layer, it learns about the user's intent throughout the session, but discards all the information at the end of the session. Similar to the model proposed in [7], we employ a GRU to take the user's k most recent history sessions into account. At the beginning of each session, the GRU models the user's long-term preference representation P_L^u from the user u's history session representations and provides it for the GRU-based decoder in the attention-based short-term preference layer as the initial hidden state.

The graph inside the dark green box in Figure 2 illustrates the work flow of the Long-term Preference Layer. For each session S_t^u in u's interaction history S^u, let s_t^u be an embedded vector representation of that session. The Long-term Preference Layer takes a series of vector representation of u's k most recent history sessions $\{s_{t-k}^u, s_{t-k+1}^u, \ldots, s_{t-1}^u\}$ as input, where s_{t-1}^u is the most recent history session. Afterwards, the layer produces the initial hidden state, H_0, of the Attention-based Short-term Intent Layer before the short-term intent layer starts making predictions. As a result, it enables the Attention-based Short-term Intent Layer to combine both the user's short-term intent and long-term preference at the attention process.

3.3.2. Attention-Based Short-Term Intent Layer

For predicting next items, a user's short-term preference is important. A user's short-term preference consists of different kinds of short-term intent on different categories of items. Previously, Rendle et al. [54] has done some research on combining long- and short-term preference for sequential recommendation. However, in the earlier works, researchers regard the user's short-term preference as a stationary property, and therefore items are given the same weights and the variation in short-term intent is not evaluated properly. To resolve this problem, we employ an attention-based encoder-decoder architecture, to assign weights to both long- and short-term session representations, to form the characteristics of user u comprehensively. In addition, we employ bi-directional LSTM and GRU as the encoder and decoder, respectively. One advantage of bi-directional LSTM over uni-directional LSTM is that it uses each event $e^u_{t,i} \in S^u_t$ to predict recommendations based on each event's past and future contexts, rather than just its past ones, which can be sensitive to the variation in the user's short-term intent from both sides. Attention mechanism has been widely and successfully applied in many tasks, e.g., information retrieval [38], computer vision [39] and recommendation [40]. The key idea of attention is to learn to assign attentive weights (normalized by sum to 1) for a set of features: higher weights indicate that the corresponding features are informative for the task. It computes the importance of each item both in the short-term and long-term item sets of a given user and integrates these items' embeddings to build the representations of the user's intent and preference.

The graph inside the purple box in Figure 2 illustrates the work flow of the Attention-based Short-term Intent Layer. In the following paragraphs, we state the functionality and the working procedure of each component in the Attention-based Short-term Intent Layer.

Encoder. A bi-directional LSTM is employed as the encoder. We use item embeddings to represent each item. At the beginning of each session, each event $e^u_{t,i}$ which consists of the embedded item representation is sent through a bi-directional LSTM to be processed together with its past and future contexts, to investigate the sequence patterns of the current session. Afterwards, the bi-directional LSTM produces its output vector, namely the source hidden states vector $\left[\overline{\mathbf{h}}_1\, \overline{\mathbf{h}}_2\, \ldots\, \overline{\mathbf{h}}_i\right]$ where $\overline{\mathbf{h}}_i$ is a source hidden state regarding the item v_i, and includes the latent information of v_i. Finally, they are sent to the architecture of attention mechanism.

Decoder. A GRU is employed as the decoder. The procedure of capturing the characteristics of each step in the user behavior is stated below. At the beginning of each session, the GRU receives the user's long-term preference representation P^u_L from the Long-term Preference Layer as the initial hidden state. In the meantime, the GRU takes the context vector $\mathbf{c}_j$ produced by the Attention procedure into consideration. In other words, $\mathbf{c}_j$ provides the distribution of every event $e^u_{t,i}$'s influence upon the current target item prediction in the current session, from which it enables the GRU to find out which events are more important to the current choice. Apparently, the more important events are highly related to the user u's current intent. Therefore, when the time step t increases and the previously important events become less important, the u's intent probably has changed. The GRU is ready for capturing the variation of the user u's current intent. In other words, our proposed model is aware of the variation of the user u's short-term intent. Moreover, it makes the recommendation with the help of not only the short-term intent, but also the long-term preference. Finally, the GRU produces the output vector, namely the target hidden states vector $\left[\mathbf{h}_1\, \mathbf{h}_2\, \ldots\, \mathbf{h}_j\right]$ where $\mathbf{h}_j$ is a target hidden state regarding the current target item prediction y_j. The target hidden states vector, together with the context vector $\mathbf{c}_j$ stated below, is used for the calculation of the target item prediction.

Architecture of Attention Mechanism. The main part of the attention mechanism is illustrated in Figure 3. The context vector $\mathbf{c}_j$ is used to capture relevant source-side information to help provide the current target item prediction y_j. Given the target hidden state $\mathbf{h}_j$ and the context vector $\mathbf{c}_j$ calculated in Equation (2), we use a concatenation to integrate both vectors to produce an attentional hidden state:

$$\mathbf{a}_j = f\left(\mathbf{c}_j, \mathbf{h}_j\right) = tanh\left(\mathbf{W}_c\left[\mathbf{c}_j; \mathbf{h}_j\right]\right), \tag{1}$$

where

$$\mathbf{c}_j = \sum_i \boldsymbol{\alpha}_{ji} \overline{\mathbf{h}}_i, \tag{2}$$

where $\boldsymbol{W}_c$ is the model parameter, and $\mathbf{c}_j$ is computed by the sum of the product of each attentive weight $\boldsymbol{\alpha}_{ji}$ and source hidden state $\overline{\mathbf{h}}_i$. A dense layer is used to fully connect the neurons in the encoder and the decoder. $\boldsymbol{\alpha}_{ji}$ is the amount of attention that the output prediction of $\mathbf{h}_j$ should pay to $\overline{\mathbf{h}}_i$, which can be calculated by the SoftMax function:

$$\boldsymbol{\alpha}_{ji} = \frac{exp(\boldsymbol{\alpha}'_{ji})}{\sum_i exp(\boldsymbol{\alpha}'_{ji})}, \tag{3}$$

where $\boldsymbol{\alpha}'_{ji}$ is used to compare the target hidden state $\mathbf{h}_j$ with each of the source hidden state $\overline{\mathbf{h}}_i$:

$$\boldsymbol{\alpha}'_{ji} = \mathbf{h}_j^\top \boldsymbol{W} \overline{\mathbf{h}}_i \tag{4}$$

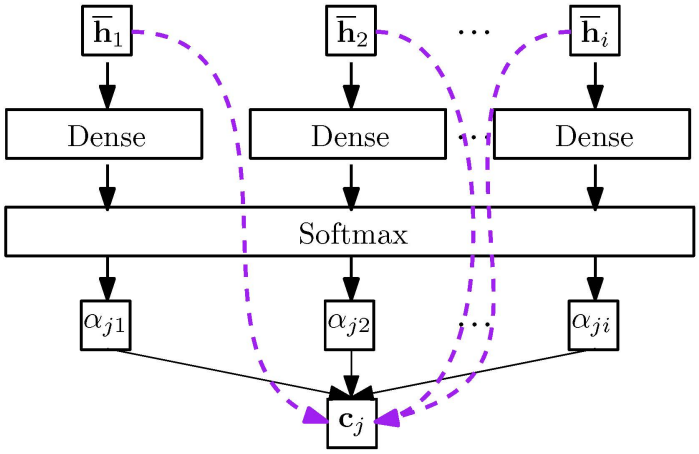

Figure 3. The main part of attention mechanism.

Denote s_t^u as the embedded vector representation of the session S_t^u. After the calculation of the attentional vector $\boldsymbol{a}_j$, it is fed through another SoftMax layer to produce the predictive distribution. Given the predictions $y_{<j} = \{y_1, y_2, \ldots, y_{j-1}\}$, s_t^u and the weight $\mathbf{W_i}$ of $\mathbf{a}_j$, the predictive distribution of items is formulated as:

$$p\left(y_j \middle| y_{<j}, s_t^u\right) = softmax\left(\mathbf{W_i}\mathbf{a}_j\right) \tag{5}$$

We apply two methods to generate the session representations s_t^u similar to that in [7]. One is to generate s_t^u with the average of the embedded vector representations of the items in the session, as illustrated in Part *Method (a): ASLM-AP* of Figure 2. The other is to use the last hidden state of attention-based short-term preference layer as the session representation, illustrated in Part *Method (b): ASLM-LHS* of Figure 2. On one hand, although there is more useful information for the Attention-based Short-term Intent Layer to learn from the final hidden state, it is more a representation of the end of the session, rather than the entire session. On the other hand, since the average of the embeddings is irrelevant to the sequence order of the items, it can represent the entire session.

4. Experimental Setup

We conduct experiments divided into two groups with different datasets to investigate the effectiveness of our model: (1) Group A with abundant user interactions for each user: Reddit and

Last.fm datasets. (2) Group B with scarcely enough user interactions for each user: 1/10 Reddit, 1/80 Last.fm and Tmall dataset.

4.1. Datasets

We experimented with three different datasets and two sub-datasets: Reddit (Subreddit dataset: https://www.kaggle.com/colemaclean/subreddit-interactions), Last.fm (Last.fm dataset: https://blog.csdn.net/hopygreat/article/details/96444827), Tmall (Tmall dataset: https://tianchi.aliyun.com/datalab/dataSet.html?spm=5176.100073.0.0.40236fc1kIb4f8&dataId=42), 1/10 Reddit and 1/80 Last.fm. The descriptive statistics for the datasets are shown in Table 1. The distribution histograms of the number of sessions by the frequency of users for Reddit, Last.fm and Tmall datasets are illustrated in Figures 4 and 5. Moreover, another set of distribution histograms, namely the one of the number of events by the frequency of sessions for Reddit, Last.fm and Tmall datasets, is given in Figures 6 and 7. Tables 2 and 3 show the standard deviation and the skewness (We used the skew function provided by SciPy to calculate the skewness. See also: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.skew.html) of them. To conserve space, only the standard deviation and the skewness but not the histograms of the two distributions for 1/10 Reddit and 1/80 Last.fm datasets are given in Tables 2 and 3.

Table 1. Descriptive statistics for the datasets.

	Reddit	Last.fm	Tmall	1/10 Reddit	1/80 Last.fm
Number of users	18,271	977	50,197	7405	612
Number of sessions	1,135,488	630,774	200,286	48,472	9099
Number of items	27,452	94,284	52,207	7170	7137
Average number of events in a session	3.00	8.10	2.99	2.24	2.09
Sessions per user	62.15	645.62	3.99	6.55	14.87

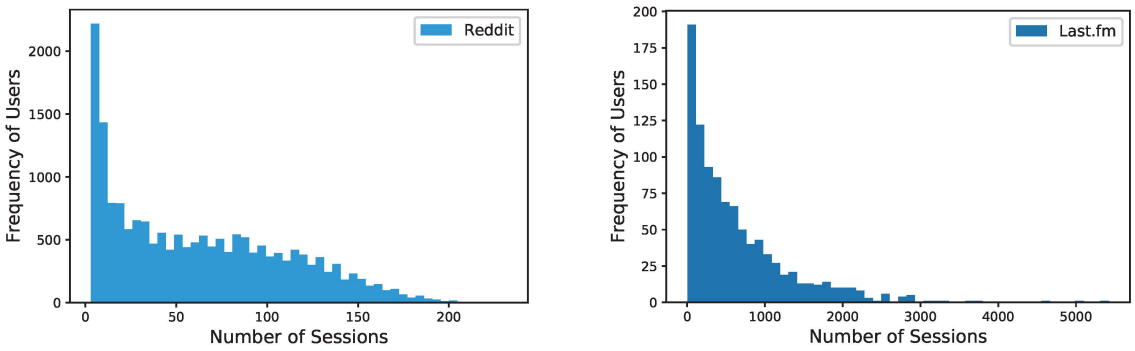

Figure 4. Distribution histograms of the number of sessions by the frequency of users for Reddit and Last.fm datasets.

Table 2. Standard deviation and skewness of the number of sessions by the frequency of users for the datasets.

	Reddit	Last.fm	Tmall	1/10 Reddit	1/80 Last.fm
Standard deviation	48.33	678.59	1.15	3.79	18.85
Skewness	0.57	2.06	12.16	1.54	4.49

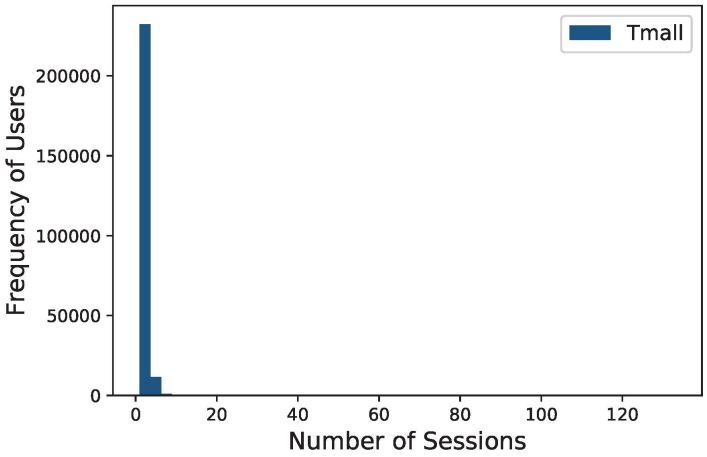

Figure 5. Distribution histogram of the number of sessions by the frequency of users for Tmall dataset.

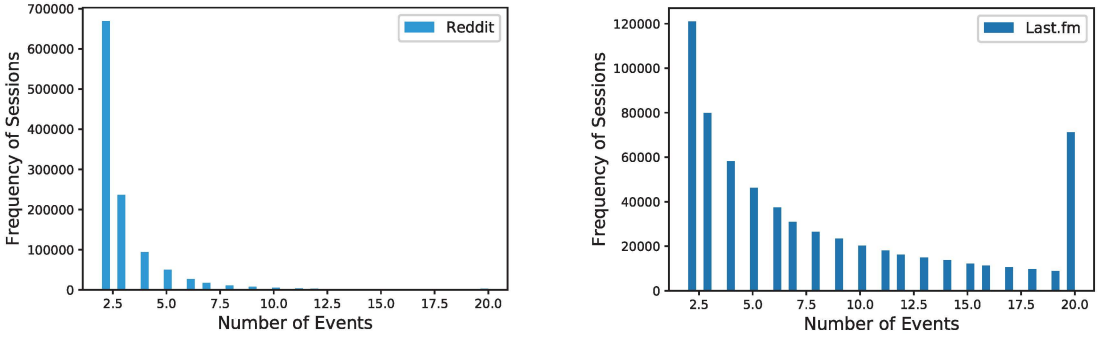

Figure 6. Distribution histograms of the number of events by the frequency of sessions for Reddit and Last.fm datasets.

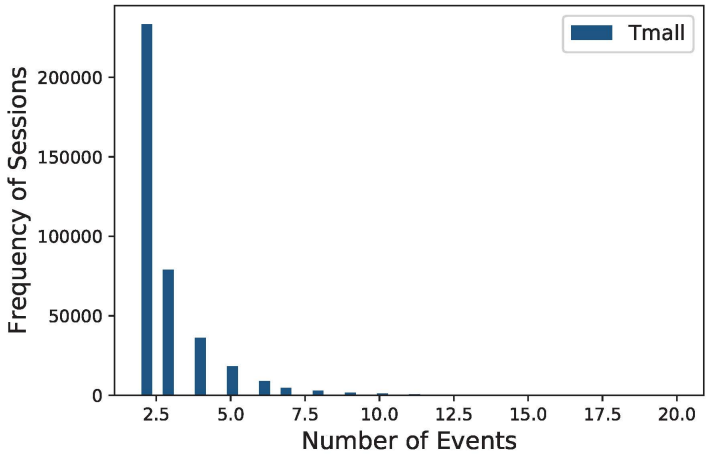

Figure 7. Distribution histogram of the number of events by the frequency of sessions for Tmall dataset.

Table 3. Standard deviation and skewness of the number of events by the frequency of sessions for the datasets.

	Reddit	**Last.fm**	**Tmall**	**1/10 Reddit**	**1/80 Last.fm**
Standard deviation	2.08	6.14	1.45	0.70	0.42
Skewness	4.08	0.83	3.05	6.61	13.63

Reddit dataset. Reddit contains a log of user interactions on different subreddits (sub-forums) at different timestamps. Since the dataset does not split the interactions into sessions, we did this by setting a time limit for inactivity and integrate the actions that occurred within one hour into a session as in [7]. Also, as in [7], we replicated their model against ours on a training set of 80% and holdout set of 20% using the same Recall and MRR (Mean Reciprocal Rank) metrics as they did [7]. We discuss the two evaluation metrics in Section 4.3. Our results replicate theirs (not shown here to minimize space). Additionally, for a deeper assessment as advocated in [55], we split the sessions into a training set (60%), a validation set (20%) and a test set (20%) to establish the performance of the alternative models on an identical basis.

Last.fm dataset. For Last.fm dataset, we split logs of every user into sessions the same way as that for Reddit. Instead, we chose a 30-minute limit as in [7]. The same procedure for assessment was used on Last.fm dataset as stated above. Afterwards, we split the sessions similar to that for Reddit dataset.

1/10 Reddit dataset. For the 1/10 Reddit dataset, we select 10% random samples from the original Reddit dataset. The same procedure for assessment was used on 1/10 Reddit dataset as stated above.

1/80 Last.fm dataset. For the 1/80 Last.fm dataset, we select 1/80 random samples from the original Last.fm dataset. The same procedure for assessment was used on 1/80 Last.fm dataset as stated above.

Tmall dataset. The Tmall dataset contains anonymized users' shopping logs in the past 6 months before and on the "Double 11" day (11 November). We only kept the purchase data. We chose a one-day limit as in [29]. Only the training set (80%) and the test set (20%) are used because of the lack of enough user interactions for each user in Tmall dataset.

4.2. Baselines

We compare our model with the following baseline algorithms.

Most Popular. This baseline always recommends the absolute most popular items in the training set. All items are sorted by their number of occurrences in the training set, and the top K items are recommended at each time step. In other words, the same items, which are the most popular ones among all the users, are recommended to each user. Although it is a simple baseline, it is usually a strong baseline especially in recommendation domains.

II-RNN-AP [7]. This baseline uses two RNNs to extract information from the user's current session and history sessions, respectively. II-RNN-AP is the II-RNN Model with average pooling to create session representations from items. The model's configurations are the same with that in [7].

II-RNN-LHS [7]. II-RNN-LHS is the II-RNN Model where the last hidden state of the Intra-session RNN is stored as the session representation. We use the same configurations in [7].

SWIWO [29]. This full model is a three-layer network that predicts the relevant item based on the user-session context. We use the same configurations in [29].

SWIWO-I [29]. This baseline is the simplified SWIWO model which only models item-session contexts without considering users. We use the same configurations in [29].

4.3. Evaluation Metrics and Hyperparameters

To evaluate the performance of each method, we employ two commonly used metrics Recall@K and MRR@K with K = 5, 10, 20 to evaluate all models. The first metric evaluates the fraction of ground truth items that have been rightly ranked over top-K items in all testing sessions, while the second metric, MRR, is a statistic measure for evaluating any process that produces a list of possible responses to a sample of queries, ordered by probability of correctness. The reciprocal rank of a query response is the multiplicative inverse of the rank of the first correct answer. For hyperparameters, we experimented with different ones to find the best configurations of our method for each dataset. The configurations of our model are summarized in Table 4.

Table 4. Configurations for ASLM.

	Reddit	Last.fm	Tmall	1/10 Reddit	1/80 Last.fm
Embedding size	50	100	100	50	100
Learning rate	0.0001	0.0001	0.0001	0.0001	0.0001
Dropout rate	0	0.2	0	0	0.2
Max. recent session representations	20	20	20	20	20
Mini-batch size	100	100	100	100	100

5. Results

In this section, the model performances of the proposed models are shown, according to the experimental settings we stated in the previous section.

Model Performance

Tables 5–7 show two groups' test set performances of all methods under the metric of Recall@K and MRR@K: (1) For Reddit and Last.fm, we set K = 5, 10 and 20. (2) For 1/10 Reddit and 1/80 Last.fm, we set the same K as the full versions of Reddit and Last.fm. For Tmall, to keep consistent with the evaluation in [29], we use Recall@10, Recall@20 and MRR@20 to evaluate methods, and cited the same evaluation results of SWIWO and SWIWO-I in [29]. Relative scores are given compared to the strongest baselines. For all datasets, ASLM-LHS outperforms all baselines with moderate margins. Table 8 shows the validation set performances of three different methods under the metric of Recall@K and MRR@K in Reddit and Last.fm datasets. Please note that, for the Reddit dataset, the validation set performance of ASLM-LHS is better than that of ASLM-AP, and, for Reddit and Last.fm datasets, the validation set performances of ASLM-AP and ASLM-LHS are all better than the test set performances with moderate margins, respectively. Table 9 shows the validation set performances of three different methods in 1/10 Reddit and 1/80 Last.fm datasets. The validation set performances of the two ASLM models in the 1/10 Reddit dataset are all better than the test set performances, while, in the 1/80 Last.fm dataset, the majority of the validation set performances of the two ASLM models are better than the test set performances.

Table 5. Group A: Recall and MRR scores for the ASLM models and the baselines for the test sets in Reddit and Last.fm datasets.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.1241	0.1799	0.2499	0.0817	0.0890	0.0940
II-RNN-AP	0.2943	0.3717	0.4527	0.1920	0.2024	0.2080
II-RNN-LHS	0.3112	0.3904	0.4709	0.2025	0.2131	0.2186
ASLM-AP	0.3162 (+1.62%)	0.3879 (−0.63%)	0.4643 (−1.40%)	0.2277 (+12.44%)	0.2373 (+11.36%)	0.2426 (+10.95%)
ASLM-LHS	**0.3396 (+9.12%)**	**0.4100 (+5.04%)**	**0.4843 (+2.85%)**	**0.2473 (+22.10%)**	**0.2567 (+20.48%)**	**0.2619 (+19.79%)**

(a) Reddit Dataset (Test).

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.0569	0.0689	0.0866	0.0470	0.0487	0.0499
II-RNN-AP	0.1009	0.1341	0.1784	0.0682	0.0725	0.0755
II-RNN-LHS	0.0855	0.1142	0.1555	0.0588	0.0625	0.0654
ASLM-AP	**0.1053 (+4.36%)**	0.1387 (+3.46%)	0.1837 (+3.01%)	**0.0732 (+7.33%)**	**0.0776 (+6.99%)**	**0.0806 (+6.75%)**
ASLM-LHS	0.1042 (+3.34%)	**0.1415 (+5.52%)**	**0.1914 (+7.31%)**	0.0693 (+1.66%)	0.0742 (+2.34%)	0.0776 (+2.78%)

(b) Last.fm Dataset (Test).

Table 6. Group B: Recall and MRR scores for the ASLM models and the baselines for the test sets in 1/10 Reddit and 1/80 Last.fm datasets.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.1451	0.2155	0.2882	0.0966	0.1060	0.1111
II-RNN-AP	0.1484	0.2161	0.2928	0.0969	0.1058	0.1112
II-RNN-LHS	0.1614	0.2292	0.3062	0.1061	0.1151	0.1204
ASLM-AP	**0.1904 (+17.96%)**	**0.2507 (+9.38%)**	0.3245 (+5.98%)	**0.1256 (+18.37%)**	**0.1336 (+16.07%)**	**0.1387 (+15.17%)**
ASLM-LHS	0.1886 (+16.83%)	0.2488 (+8.55%)	**0.3251 (+6.19%)**	0.1234 (+16.27%)	0.1314 (+14.13%)	0.1366 (+13.48%)

(a) 1/10 Reddit Dataset (Test).

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.0371	0.0493	0.0596	0.0290	0.0308	0.0314
II-RNN-AP	0.0349	0.0448	0.0604	0.0282	0.0294	0.0305
II-RNN-LHS	0.0361	0.0459	0.0608	0.0285	0.0298	0.0308
ASLM-AP	**0.0388 (+4.38%)**	0.0496 (+0.57%)	0.0656 (+7.95%)	0.0308 (+6.18%)	0.0323 (+4.93%)	0.0334 (+6.13%)
ASLM-LHS	**0.0388 (+4.38%)**	**0.0503 (+1.92%)**	**0.0674 (+10.97%)**	**0.0309 (+6.52%)**	**0.0324 (+5.37%)**	**0.0336 (+6.98%)**

(b) 1/80 Last.fm Dataset (Test).

Table 7. Group B: Recall and MRR scores for the ASLM models and the baselines in the Tmall dataset.

	R@10	R@20	MRR@20
Most Popular	0.0234	0.0420	0.0123
SWIWO-I	0.3177	0.3810	0.1903
SWIWO	0.3082	0.3703	0.1885
ASLM-AP	0.5412 (+70.34%)	0.5661 (+48.57%)	0.4037 (+112.12%)
ASLM-LHS	**0.5428 (+70.86%)**	**0.5679 (+49.06%)**	**0.4082 (+114.50%)**

Table 8. Group A: Recall and MRR scores for the ASLM models and the baselines for the validation sets in Reddit and Last.fm datasets.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.1269	0.1830	0.2529	0.0835	0.0908	0.0957
II-RNN-AP	0.3024	0.3827	0.4653	0.1971	0.2079	0.2136
II-RNN-LHS	0.3191	0.4013	0.4837	0.2073	0.2182	0.2239
ASLM-AP	0.3255 (+2.02%)	0.3991 (−0.54%)	0.4768 (−1.43%)	0.2336 (+12.72%)	0.2434 (+11.55%)	0.2488 (+11.10%)
ASLM-LHS	**0.3489 (+9.34%)**	**0.4218 (+5.12%)**	**0.4982 (+3.00%)**	**0.2527 (+21.90%)**	**0.2624 (+20.26%)**	**0.2677 (+19.56%)**

(a) Reddit Dataset (Validation).

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.0563	0.0685	0.0862	0.0460	0.0477	0.0489
II-RNN-AP	0.1082	0.1460	0.1962	0.0703	0.0753	0.0787
II-RNN-LHS	0.0921	0.1249	0.1705	0.0614	0.0658	0.0689
ASLM-AP	0.1147 (+6.07%)	0.1526 (+4.50%)	0.2027 (+3.33%)	**0.0776 (+10.44%)**	**0.0826 (+9.69%)**	**0.0860 (+9.36%)**
ASLM-LHS	**0.1179 (+9.00%)**	**0.1613 (+10.48%)**	**0.2186 (+11.42%)**	0.0760 (+8.21%)	0.0818 (+8.63%)	0.0857 (+8.94%)

(b) Last.fm Dataset (Validation).

Table 9. Group B: Recall and MRR scores for the ASLM models and the baselines for the validation sets in 1/10 Reddit and 1/80 Last.fm datasets.

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.1516	0.2206	0.2913	0.1004	0.1095	0.1145
II-RNN-AP	0.1551	0.2215	0.2966	0.1026	0.1113	0.1165
II-RNN-LHS	0.1702	0.2366	0.3116	0.1108	0.1196	0.1248
ASLM-AP	**0.1982 (+16.47%)**	**0.2563 (+8.33%)**	0.3282 (+5.32%)	**0.1315 (+18.68%)**	**0.1392 (+16.42%)**	**0.1442 (+15.58%)**
ASLM-LHS	0.1935 (+13.71%)	0.2549 (+7.76%)	**0.3302 (+5.96%)**	0.1275 (+15.04%)	0.1356 (+13.38%)	0.1408 (+12.82%)

(a) 1/10 Reddit Dataset (Validation).

	R@5	R@10	R@20	MRR@5	MRR@10	MRR@20
Most Popular	0.0360	0.0473	0.0626	0.0284	0.0299	0.0310
II-RNN-AP	0.0343	0.0411	0.0563	0.0286	0.0295	0.0306
II-RNN-LHS	0.0372	0.0454	0.0641	0.0317	0.0327	0.0340
ASLM-AP	0.0380 (+1.97%)	0.0476 (+0.55%)	**0.0663 (+3.54%)**	0.0318 (+0.21%)	0.0330 (+0.92%)	**0.0343 (+0.88%)**
ASLM-LHS	**0.0398 (+6.89%)**	**0.0482 (+1.75%)**	0.0644 (+0.57%)	**0.0321 (+1.26%)**	**0.0332 (+1.43%)**	**0.0343 (+0.88%)**

(b) 1/80 Last.fm Dataset (Validation).

6. Discussion

In this section, we discuss the superiority of the proposed models through comparison of performance and demonstrate their effectiveness on the session cold-start problem.

6.1. Comparison of Performance

For Group A, in Table 5, we observe that ASLM-LHS consistently outperforms all the baselines under all measurements for testing cases in both Reddit and Last.fm datasets with moderate margins; In most cases, ASLM-AP outperforms all the baselines in both datasets, except in Recall@10 and Recall@20 scores in the Reddit test set. Specifically, ASLM-LHS improves 9.12% and 22.10% in Recall@5 and MRR@5 scores compared with the II-RNN-LHS method for test cases in the Reddit dataset, respectively. ASLM-AP improves 4.36% and 7.33% in Recall@5 and MRR@5 compared with the II-RNN-AP method for test cases in the Last.fm dataset, respectively. The reason ASLM-AP's Recall@10 and Recall@20 scores are lower than those of II-RNN-LHS in the Reddit dataset is that the average pooling method only utilizes the average of the embedding of each event $e^u_{t,i} \in S^u_t$ as the representation of the current session S^u_t, and the important context information, e.g., the sequential patterns and the user's intent captured by the attention mechanism, will be lost when this representation is stored in the user's long-term session representations. It results in the fact that the abundant context information cannot be fully utilized by ASLM-AP, and thus its performance declined. However, the performance of ASLM-AP is still better than II-RNN-AP, which also utilizes the average pooling method. The good performance of ASLM-LHS is attributed to the attention mechanism and the bidirectional-LSTM we employed in the attention-based layer and the last hidden state method. One of the main characteristics of the attention mechanism is that it calculates the importance of each given event and captures the user's short-term intent with it. The bidirectional-LSTM can extract the sequential patterns from each given event in both the forward and the backward directions. Both ASLM-LHS and II-RNN-LHS utilize the last hidden state method which retains the context information from the current session and regards it as one part of the user's long-term session representations, compared with the average pooling method. Therefore, the model's subsequent training process benefits from this method by receiving the valuable context information from history sessions. In addition, in Table 1 we observe that, in the

Reddit dataset, the number of sessions per user (namely, 62.15) and the average number of events in a session (namely, 3.00) are much smaller than those in the Last.fm dataset (namely, 645.62 and 8.10), which shows that ASLM-LHS can perform better when the user's history information is less adequate. Tables 5 and 8 show the results that the performance of ASLM for validation cases and that for testing cases in Reddit and Last.fm datasets are at the same level, which assure us the validity of our model.

For Group B, similar results can be seen in Tables 6 and 7 that ASLM-LHS and ASLM-AP outperform all the baselines under all measurements in 1/10 Reddit, 1/80 Last.fm, and Tmall datasets. For the Tmall dataset, the evaluation result of ASLM-LHS shown in Table 7 improves 70.86%, 49.06%, and 114.50% in R@10, R@20, and MRR@20 scores compared with the SWIWO-I method, respectively. Please note that, for the Tmall dataset, as shown in Table 1, there is scarcely enough sessions for each user (namely, 3.99 per user) and number of average session length (2.99, per user) compared with those in Reddit and Last.fm. For 1/10 Reddit and 1/80 Last.fm, since the user history information is reduced dramatically compared with the full Reddit and Last.fm datasets, most of the models' performances significantly decrease accordingly. However, both ASLM models still outperform the strongest baselines, as shown in Table 6. It demonstrates that ASLM can reach better performance even when it is severely short of both long-term and short-term user behavior.

When K changes from 5 to 20, for the Reddit and 1/10 Reddit datasets, there is a downward trend in the relative scores of ASLM-LHS and ASLM-AP. For the Last.fm dataset, there is an upward trend in those of ASLM-LHS. The reason for the upward trend is that there is more abundant user history information in the Last.fm dataset, and, with the increase of K, ASLM-LHS still has the potential to further capture the context information implied by the Last.fm dataset, which results in some space to improve its performance.

In terms of the two versions of ASLM (ASLM-AP using average pooling and ASLM-LHS using the last hidden state), Tables 5–7 show that, for the Reddit dataset, the test performance of ASLM-LHS has an obvious advantage compared to that of ASLM-AP, while they are at the same level for the other four datasets. As mentioned above, the model's subsequent training process benefits from the last hidden state method by receiving the valuable context information from history sessions.

6.2. *Effectiveness on Session Cold-Start Problem*

In addition to the overall recommendations we reported in Sections 5 and 6.1, we evaluate the effectiveness of ASLM-LHS on recommendations for the first n time steps, for $n = 1, ..., 5, L$, where L is the maximum session length and $L = 19$ in our experiment.

Figure 8 shows the comparison of performances of Recall@5 at the first n recommendations of each session for Reddit and Last.fm datasets between ASLM and II-RNN.

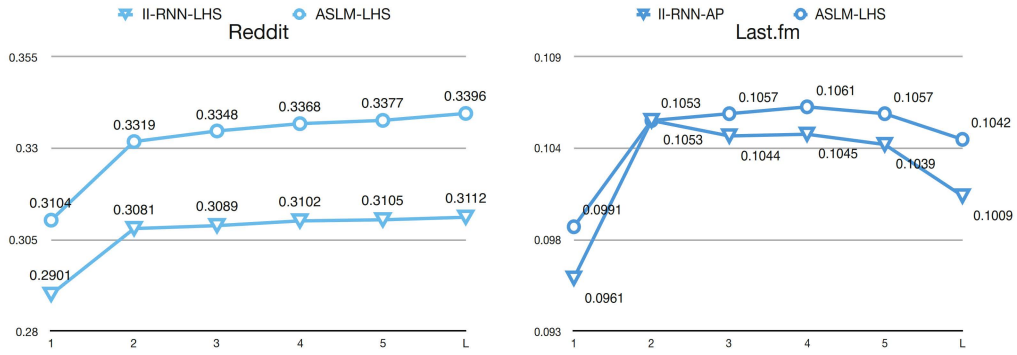

Figure 8. Comparison of cold-start effects at the start of a new session for Reddit and Last.fm datasets between ASLM and II-RNN.

Notice that, for Reddit and Last.fm, where abundant user history behavior is given, our attention-based ASLM-LHS model achieves $R@5 > 0.3$ and $R@5 > 0.099$ when $n = 1$, respectively, and,

under most circumstances, each *R*@5 score of the ASLM-LHS model increases throughout the session. ASLM-LHS has an advantage over II-RNN-LHS and II-RNN-AP at the start of a new session in Reddit and Last.fm datasets, with the improvements of 7.00% and 3.12%, respectively. When more events have been investigated in the current session, for the Reddit dataset, the advantage becomes more obvious, while, for the Last.fm dataset, since the performances of ASLM-LHS and II-RNN-AP are identical when $n = 2$, the advantage becomes greater when n increases from 3.

Similar results are shown in Figure 9 for the Tmall dataset. In terms of the Tmall dataset, which is severely short of both long-term and short-term user behavior, ASLM-LHS still provides 0.5063 in *R*@5 with just a slight difference compared with the overall *R*@5 (i.e., 0.5074). Due to the lack of history information from each user in the Tmall dataset, its performance is not stable at the start of a session when n is between 1 and 3, however it gets better when n increases from 3. As a result, the attention-based short-term intent layer can thoroughly reveal the variation in the user's short-term intent between events from the current session S_t^u. To sum up, ASLM significantly alleviates the cold-start problem.

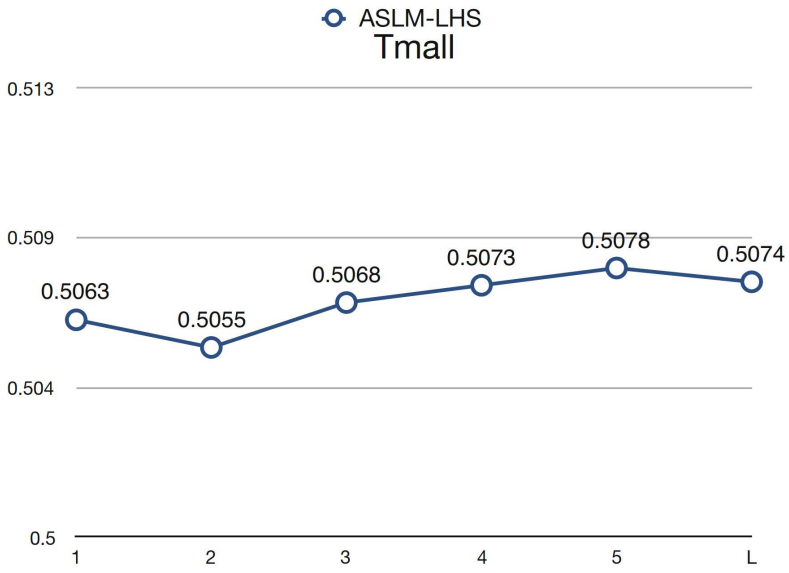

Figure 9. Cold-start effect at the start of a new session for the Tmall dataset. Please note that no data are provided for the cold-start problem by the SWIWO method.

6.3. Implications of the Work

This system is designed to address situations where only past decisions of the users are available (e.g., there are no ratings by the users of the items). This is a very common situation, and yet recommendations are still needed and can be produced by estimating the interest drift of user preferences from past choices across items, and the pattern and sequence of past choices across sessions and particularly in the current session.

The ASLM can be applied by recommender systems in the scenarios of news, music, and e-commerce (e.g., Reddit, Last.fm and Tmall, respectively) where the user's interest drift is the main topic since it takes place frequently. The model depends on the data availability because it is a data-driven method, and continuously requires the user's session data to process when running. During the experiment, it took about 28 min for the ASLM to run every epoch on Reddit dataset which contains 1,135,488 user sessions with a NVIDIA GTX 1080 Ti graphic card. In other words, it took about 1.5 ms for each session, and that is appropriate for real-time recommendation. Therefore, the system can be employed online and executed in real-time, although we conducted the experiment offline.

7. Conclusions

In this work, we proposed an attention-based RNNs named ASLM for next-item recommendation problems. By integrating both the user's short-term intent and long-term preference at the same time with a two-layer network, our model makes the best use of the awareness of the variation in the user's contextual preference, especially the short-term intent, to cope with the cold-start problem within sessions. We have conducted experiments on three real-world datasets and two sub-datasets and demonstrated considerable improvements over strong baselines. Moreover, our proposed model can provide accurate recommendation at the start of sessions, which remarkably mitigates the cold-start problem.

Author Contributions: Conceptualization, R.H., S.M., M.S. and H.E.; Methodology, R.H.; Software, R.H.; Validation, R.H., S.M., M.S., H.E. and Z.O.; Formal Analysis, R.H.; Investigation, R.H.; Resources, R.H., S.M., M.S. and H.E.; Data Curation, R.H.; Writing—Original Draft Preparation, R.H.; Writing—Review & Editing, R.H., S.M., M.S., H.E. and Z.O.; Visualization, R.H.; Supervision, S.M., M.S., H.E. and Z.O.; Project Administration, Z.O.; Funding Acquisition, Z.O.

Funding: This research was funded by National Natural Science Foundation of China (Grant No. 61702046).

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

Abbreviations

The following abbreviations are used in this manuscript (in alphabetical order):

AP	Average Pooling
ASLM	Attention-based Short-term and Long-term Model
GRU	Gated Recurrent Unit
II-RNN	Inter- and Intra-session Recurrent Neural Network
LHS	Last Hidden State
LSTM	Long Short-Term Memory
RNN	Recurrent Neural Network
SWIWO	Session-based Wide-In-Wide-Out

References

1. Deep Learning: A Brief Guide for Practical Problem Solvers. Available online: https://www.infoworld.com/article/3003315/big-data/deep-learning-a-brief-guide-for-practical-problem-solvers.html (accessed on 28 November 2018).
2. Karthik, S.; Paul, A.; Karthikeyan, N. *Deep Learning Innovations and Their Convergence with Big Data*; IGI Global: Hershey, PA, USA, 2017.
3. Zhang, S.; Yao, L.; Sun, A. Deep learning based recommender system: A survey and new perspectives. *arXiv*, 2017; arXiv:1707.07435.
4. Ludewig, M.; Jannach, D. Evaluation of Session-based Recommendation Algorithms. *arXiv*, 2018; arXiv:1803.09587.
5. Quadrana, M.; Cremonesi, P.; Jannach, D. Sequence-aware recommender systems. *arXiv*, 2018; arXiv:1802.08452.
6. Quadrana, M.; Karatzoglou, A.; Hidasi, B.; Cremonesi, P. Personalizing session-based recommendations with hierarchical recurrent neural networks. In Proceedings of the 11th ACM Conference on Recommender Systems, Como, Italy, 27–31 August 2017; pp. 130–137.
7. Ruocco, M.; Skrede, O.S.L.; Langseth, H. Inter-Session Modeling for Session-Based Recommendation. In Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems, Como, Italy, 27–31 August 2017; pp. 24–31.
8. Kraus, M.; Feuerriegel, S. Decision support from financial disclosures with deep neural networks and transfer learning. *Decis. Support Syst.* **2017**, *104*, 38–48. [CrossRef]

9. Bountris, P.; Kotronoulas, G.; Tagaris, T.; Haritou, M.; Spathis, A.; Karakitsos, P.; Koutsouris, D. CxCaDSS: A web-based clinical decision support system for cervical cancer. In Proceedings of the 6th European Conference of the International Federation for Medical and Biological Engineering, Dubrovnik, Croatia, 7–11 September 2014; Volume 45, pp. 757–760.
10. Zeng, Y.; Wang, L.; Chen, T.; Lu, Y. A hybird intelligent decision support system for spare parts inventory control using neural network and gene algorithms approach. In Proceedings of the IASTED International Conference on Power, Energy, and Applications, PEA 2006: Science and Technology for Development in the 21st Century, Gaborone, Botswana, 11–13 September 2006; pp. 32–36.
11. Lai, K.K.; Yu, L.; Wang, S. A neural network and web-based decision support system for forex forecasting and trading. In *Data Mining and Knowledge Management*; Springer: Berlin, Germany, 2005; pp. 243–253.
12. Zavadskas, E.K.; Kaklauskas, A.; Gulbinas, A. Multiple criteria decision support web-based system for building refurbishment. *J. Civ. Eng. Manag.* **2004**, *10*, 77–85. [CrossRef]
13. Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; Tikk, D. Session-based Recommendations with Recurrent Neural Networks. In Proceedings of the International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016; pp. 1–10.
14. Cremonesi, P.; Garzotto, F.; Turrin, R. Investigating the persuasion potential of recommender systems from a quality perspective: An empirical study. *ACM Trans. Interact. Intell. Syst.* **2012**, *2*, 11. [CrossRef]
15. Garcin, F.; Faltings, B.; Donatsch, O.; Alazzawi, A.; Bruttin, C.; Huber, A. Offline and online evaluation of news recommender systems at swissinfo.ch. In Proceedings of the 8th ACM Conference on Recommender Systems, Foster City, CA, USA, 6–10 October 2014; pp. 169–176.
16. Gomez-Uribe, C.A.; Hunt, N. The netflix recommender system: Algorithms, business value, and innovation. *ACM Trans. Manag. Inf. Syst.* **2016**, *6*, 13. [CrossRef]
17. Jannach, D.; Hegelich, K. A case study on the effectiveness of recommendations in the mobile internet. In Proceedings of the Third ACM Conference on Recommender Systems, New York, NY, USA, 23–25 October 2009; pp. 205–208.
18. Kirshenbaum, E.; Forman, G.; Dugan, M. A live comparison of methods for personalized article recommendation at forbes.com. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Riva del Garda, Italy, 19–23 September 2012; Springer: Berlin, Germany, 2012; pp. 51–66.
19. Lerche, L.; Jannach, D.; Ludewig, M. On the value of reminders within e-commerce recommendations. In Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, Halifax, NS, Canada, 13–17 July 2016; pp. 27–35.
20. Herlocker, J.L.; Konstan, J.A.; Terveen, L.G.; Riedl, J.T. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.* **2004**, *22*, 5–53. [CrossRef]
21. Tan, Y.K.; Xu, X.; Liu, Y. Improved recurrent neural networks for session-based recommendations. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 17–22.
22. Jannach, D.; Ludewig, M. When recurrent neural networks meet the neighborhood for session-based recommendation. In Proceedings of the 11th ACM Conference on Recommender Systems, Como, Italy, 27–31 August 2017; pp. 306–310.
23. Dong, D.; Zheng, X.; Zhang, R.; Wang, Y. Recurrent Collaborative Filtering for Unifying General and Sequential Recommender. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 3350–3356.
24. Liang, N.; Zheng, H.T.; Chen, J.Y.; Sangaiah, A.K.; Zhao, C.Z. TRSDL: Tag-Aware Recommender System Based on Deep Learning–Intelligent Computing Systems. *Appl. Sci.* **2018**, *8*, 799. [CrossRef]
25. Zimdars, A.; Chickering, D.M.; Meek, C. Using temporal data for making recommendations. In Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence, Seattle, WA, USA, 2–5 August 2001; pp. 580–588.
26. Mobasher, B.; Dai, H.; Luo, T.; Nakagawa, M. Using sequential and non-sequential patterns in predictive web usage mining tasks. In Proceedings of the IEEE International Conference on Data Mining, Maebashi City, Japan, 9–12 December 2002; pp. 669–672.

27. Park, S.E.; Lee, S.; Lee, S.G. Session-based collaborative filtering for predicting the next song. In Proceedings of the IEEE 2011 First ACIS/JNU International Conference on Computers, Networks, Systems and Industrial Engineering (CNSI), Jeju Island, Korea, 23–25 May 2011; pp. 353–358.
28. Moore, J.L.; Chen, S.; Turnbull, D.; Joachims, T. Taste Over Time: The Temporal Dynamics of User Preferences. In Proceedings of the 14th International Society for Music Information Retrieval Conference, Curitiba, Brazil, 4–8 November 2013; pp. 401–406.
29. Hu, L.; Cao, L.; Wang, S.; Xu, G.; Cao, J.; Gu, Z. Diversifying personalized recommendation with user-session context. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 1858–1864.
30. Jing, H.; Smola, A.J. Neural survival recommender. In Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge, UK, 6–10 February 2017; pp. 515–524.
31. Wu, C.Y.; Ahmed, A.; Beutel, A.; Smola, A.J.; Jing, H. Recurrent recommender networks. In Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge, UK, 6–10 February 2017; pp. 495–503.
32. Yu, F.; Liu, Q.; Wu, S.; Wang, L.; Tan, T. A Dynamic Recurrent Model for Next Basket Recommendation. In Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 729–732.
33. Digital Search Advertising Revenue Growth in the United States from 2017 to 2022. Available online: https://www.statista.com/statistics/460955/search-advertising-revenue-growth-digital-market-outlook-usa/ (accessed on 26 November 2018).
34. Cheng, Z.; Ding, Y.; He, X.; Zhu, L.; Song, X.; Kankanhalli, M.S. A^3NCF: An Adaptive Aspect Attention Model for Rating Prediction. In Proceedings of the 27th International Joint Conference on Artificial Intelligence, Stockholm, Sweden, 13–19 July 2018; pp. 3748–3754.
35. Wang, S.; Hu, L.; Cao, L.; Huang, X.; Lian, D.; Liu, W. Attention-based transactional context embedding for next-item recommendation. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; pp. 2532–2539.
36. Tay, Y.; Tuan, L.A.; Hui, S.C. Multi-Pointer Co-Attention Networks for Recommendation. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 1–10.
37. Zhou, C.; Bai, J.; Song, J.; Liu, X.; Zhao, Z.; Chen, X.; Gao, J. ATRank: An Attention-Based User Behavior Modeling Framework for Recommendation. *arXiv*, 2017; arXiv:1711.06632.
38. Xiong, C.; Callan, J.; Liu, T.Y. Word-Entity Duet Representations for Document Ranking. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Tokyo, Japan, 7–11 August 2017; pp. 763–772.
39. Chen, L.; Zhang, H.; Xiao, J.; Nie, L.; Shao, J.; Liu, W.; Chua, T.S. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6298–6306.
40. Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; Chua, T.S. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In Proceedings of the 26th International Joint Conference on Artificial Intelligence, Melbourne, Australia, 19–25 August 2017; pp. 3119–3125.
41. Adomavicius, G.; Tuzhilin, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 734–749. [CrossRef]
42. Bobadilla, J.; Ortega, F.; Hernando, A.; Gutiérrez, A. Recommender systems survey. *Knowl. Based Syst.* **2013**, *46*, 109–132. [CrossRef]
43. Bobadilla, J.; Hernando, A.; Ortega, F.; Bernal, J. A framework for collaborative filtering recommender systems. *Expert Syst. Appl.* **2011**, *38*, 14609–14623. [CrossRef]
44. Schafer, J.B.; Frankowski, D.; Herlocker, J.; Sen, S. Collaborative filtering recommender systems. In *The Adaptive Web*; Springer: Berlin, Germany, 2007; pp. 291–324.
45. Su, X.; Khoshgoftaar, T.M. A survey of collaborative filtering techniques. *Adv. Artif. Intell.* **2009**, *2009*. [CrossRef]
46. Bobadilla, J.; Serradilla, F. The effect of sparsity on collaborative filtering metrics. In Proceedings of the Twentieth Australasian Conference on Australasian Database, Wellington, New Zealand, 1 January 2009; Volume 92, pp. 9–18.

47. Luo, X.; Xia, Y.; Zhu, Q. Incremental collaborative filtering recommender based on regularized matrix factorization. *Knowl. Based Syst.* **2012**, *27*, 271–280. [CrossRef]
48. Bobadilla, J.; Ortega, F.; Hernando, A.; Bernal, J. A collaborative filtering approach to mitigate the new user cold start problem. *Knowl. Based Syst.* **2012**, *26*, 225–238. [CrossRef]
49. Kim, H.N.; El-Saddik, A.; Jo, G.S. Collaborative error-reflected models for cold-start recommender systems. *Decis. Support Syst.* **2011**, *51*, 519–531. [CrossRef]
50. Schein, A.I.; Popescul, A.; Ungar, L.H.; Pennock, D.M. Methods and metrics for cold-start recommendations. In Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval, Tampere, Finland, 11–15 August 2002; pp. 253–260.
51. Devooght, R.; Bersini, H. Collaborative filtering with recurrent neural networks. *arXiv*, 2016; arXiv:1608.07400.
52. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef] [PubMed]
53. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
54. Rendle, S.; Freudenthaler, C.; Schmidt-Thieme, L. Factorizing personalized markov chains for next-basket recommendation. In Proceedings of the 19th International Conference on World Wide Web, Raleigh, NC, USA, 26–30 April 2010; pp. 811–820.
55. What Is the Difference between Test and Validation Datasets? Available online: https://machinelearningmastery.com/difference-test-validation-datasets/ (accessed on 26 November 2018).