Interest Sustainability-Aware Recommender System

Dongmin Hyun[1], Junsu Cho[1], Chanyoung Park[2], Hwanjo Yu[1]
Dept. of Computer Science and Engineering, POSTECH, Pohang, Republic of Korea[1]
Dept. of Industrial and Systems Engineering, KAIST, Daejeon, Republic of Korea[2]
{dm.hyun, junsu7463, hwanjoyu}@postech.ac.kr, cy.park424@gmail.com

***Abstract*—The key to successful recommendations is to provide users with items likely to be consumed in the future. From real-world data, we observe that users' consumption patterns for items change over time. For example, users may no longer like some items they liked in the past. However, existing recommender systems model user's preference to items without considering how much users' interests in each item will sustain in the future. Thus, they often recommend less interesting items in the deployment time (i.e., test time). In this work, we propose a novel recommender system, called CRIS, that considers the change of users' interest in each item over time. More precisely, we first predict the interest sustainability of each item, that is, how likely each item will be consumed in the future. Then, our goal is to make users closer to the items with high interest sustainability scores in the representation space than those with low interest sustainability scores. We perform experiments on 11 real-world datasets to show the effectiveness of CRIS. We also show that considering the interest sustainability is indeed crucial for boosting the accuracy of recommendations.**

***Index Terms*—Information Retrieval, Recommender System, Representation Learning, Interest Sustainability, Concept Drift**

I. Introduction

According to a recent technical report from Amazon.com, an estimated 30% of their page views were from recommendations [1]. Consequently, a plethora of research has been devoted to building successful recommender systems. Recommender systems mainly depend on users' interactions (e.g., purchase) with items to learn the users' preference to items, and produce a list of appealing items for each user to promote consumption. An important aspect for building successful recommender systems is to consider the concept drift [2], [3] of users. More precisely, a user's interest changes over time, and the preference even towards the same type of items can change. For example, most users who liked wired earphones (e.g., EarPods) may change their interests over time and prefer wireless earphones (e.g. AirPods).

Existing methods capture the concept drift of users mainly based on each user's consumption history, but they do not take into account how users' interest in each item will sustain in the future. As a prominent approach, sequential recommender systems have been introduced [3]–[7]. They take a user's N recently-consumed items as input to predict the next item that the user would consume. The underlying intuition is that the order of items in a user's consumption history can represent the concept drift of the users. Despite their success, previous sequential recommender systems are limited in that they overlook how much users' interests in each item will sustain in the future.

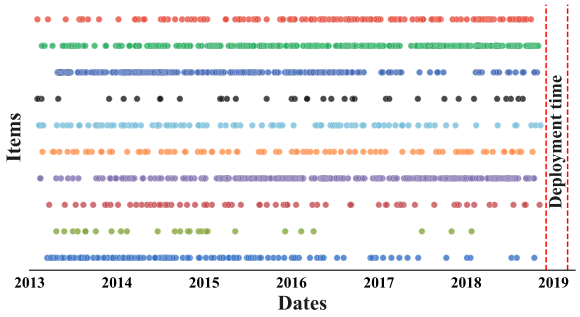


Fig. 1: Consumption timestamps of items sampled from Yelp dataset. Dots in each line indicate the timestamps at which an item was consumed. Some items (e.g., the item at the bottom) might have a lower chance to be consumed in the deployment time (i.e., test time) than other items (e.g., the top item).

To model the concept drift of users, recommender systems should focus on items that are likely to sustain users' interest until the deployment time, i.e., the actual time at which items are recommended. In Fig. 1, we illustrate the existence of such concept drift of users in Yelp dataset. Suppose there are restaurants opened in 2013, where some restaurants (e.g. the top item in Fig. 1) have attracted user's interest until recently, while other restaurants (e.g., the item at the bottom in Fig. 1) have gradually lost users' interest. In this example, since the restaurants that belong to the former case are more likely to attract users in the deployment time than those that belong to the latter case, it would be better to recommend more of the former restaurants. Therefore, we should consider how likely each item is to sustain users' interest in the deployment time.

Beyond modeling the sequential information as done by previous recommender systems, we propose a novel recommender system, Collaborative Representation Learning with Interest Sustainability (CRIS), that takes a totally different approach to model the concept drift of users. The crux of our method is to recommend items based on the interest sustainability score (ISS), which is a score of how much users' interest in each item will sustain in the future. More precisely, prior to training the recommendation model, we first compute the ISS of each item by training a neural classifier in a supervised manner. Based on the predicted ISS of each item, we then propose a metric learning framework to make users closer to the items

2374-8486/20/$31.00 ©2020 IEEE
DOI 10.1109/ICDM50108.2020.00028

with high ISSs in the representation space than those with low ISSs, thereby recommending items that would be attractive to users in the deployment time. However there can be a potential conflict between modeling the ISSs and the original objective of the metric learning, that is, making users closer to items they consumed than items they did not consume. For example, an item consumed by a user should be close to the user according to the original objective, but if the ISS of the item is low, the item is forced to be distant from the user, which prevents the recommendation system from fully learning the user's preference for items. In the light of this issue, we further improve the method with prototypes [8] to relieve the conflicts between the objectives.

We performed extensive experiments on 11 real-world datasets to evaluate the effectiveness of CRIS[1]. Experimental results show that CRIS achieves the state-of-the-art performance compared to various recommender systems including the sequential and metric learning-based methods. The improvement of CRIS against the best-performing baseline is 9.4% on average in terms of ranking metrics. We also provide analyses such as the representation space learned by CRIS, and demonstrate that the ISS is indeed crucial to boost the accuracy of recommendations.

II. Related Work

A. General Recommender Systems

The primary goal of recommender systems is to predict users' preference (e.g., the next items to be consumed or ratings of items) based on users' interaction (e.g., purchase or click) with items in online services like Amazon. Matrix factorization (MF) [9] has been a popular technique in recommendation. MF predicts a user's preference to an item as inner product between the latent factors of the user and item. Another popular technique is Bayesian personalized ranking (BPR) [10], which is a learning mechanism imposing recommender systems to more focus on items consumed by users than items not consumed by the users. The most widely-used variant of BPR is BPR-MF, which uses MF as a model with the learning mechanism of BPR. However, inner product does not satisfy the triangular inequality, and thus the relationship between users and items cannot be fully captured [11]. Collaborative Metric Learning (CML) [11] resolves the problem of inner product by learning a metric that satisfies the triangular inequality such as the euclidean distance. There have been several efforts to enhance CML [12]–[14]. Recently, Symmetric metric learning (SML) [14] further enhances CML by incorporating an item-centric metric and trainable margins for each user and item.

B. Sequential Recommender Systems

Concept drift has been studied in recommendation to model users' evolving preference to items over time. Sequential recommender systems have addressed the concept drift of users by considering a users' consumption history as a sequence of items. First-order Markov chain [3] is an early work of sequential recommender systems, and it takes a user's N recently-consumed items to predict the next items that the user would consume by modeling the first-order relationship among the items (i.e., an item is only affected by its previously consumed item). To model a higher-order interaction among the N items, several works depend on neural networks. Caser [7] builds a model based on convolutional neural networks (CNNs) while controlling the length of Markov chain in order to predict the next item. HGN [4] enriches the interaction among items with a hierarchical gating network instead of CNN used in Caser. In contrast to Caser and HGN, SASRec [6] adopts the self-attention mechanism in Transformer [15] to capture pair-wise interactions between items in a user's consumption history. TiSASRec [5] refines SASRec by involving time intervals in the self-attention mechanism between a user's interactions to better model users' consumption history.

Nevertheless, these methods do not consider how much users' interest in each item will sustain in the future. This motivates us to build a recommender system that captures the sustainability of users' interest in each item so as to more likely recommend items that would be consumed in the deployment time.

C. Survival Analysis in Recommendation

Survival analysis is another line of research that is related to our idea of considering the sustainability of users' interest in items. This technique estimates the probability that objects (e.g., patients in clinic or devices in engineering) will survive beyond any specific time. A pioneering work [16] performed survival analysis to explore users' browsing patterns based on users' dwell time in online services. Neural survival recommender [17] is a sequential recommender system that combines survival analysis with a neural network by considering time intervals between users' consecutive interactions to predict when users will return to services and produce recommendations. In contrast, ISS introduced in this work represents how much users' interest in each item will sustain in the future, which has not been studied in recommendation to the best our knowledge.

III. Proposed Method

In this section, we first demonstrate how to obtain the interest sustainability of items, that is, interest sustainability score (ISS) that quantifies how much users' interest in items will sustain in the future. Then, based on the obtained ISS, we propose a metric learning framework for capturing the concept drift of users.

A. Interest Sustainability Prediction

Prior to training the recommender system, we train a neural classifier, which predicts whether each item will be consumed in the future, to obtain the ISS for each item.

Consider that we have user-item interaction data D such that:

$$D = \{(u, i, t) \,|\, \text{user } u \text{ consumed item } i \text{ at time } t\}$$

[1]Source codes are available at: https://github.com/dmhyun/CRIS/

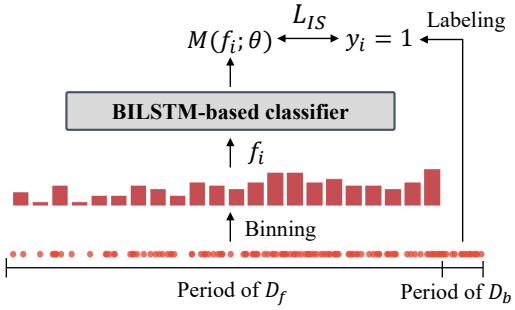


Fig. 2: Training process of a propose classifier on the interest sustainability prediction.

where data D is a general source to train recommender systems[2].

We first divide data D chronologically such that $D = D_f \parallel D_b$ where D_f and D_b denote the front and back part after dividing data D, respectively, all interactions in D_f are precedent to any interaction in D_b, and $\parallel$ is the concatenation operation. The divided data D_f and D_b are used for building input and label, respectively, as follows:

$$\textbf{Input} : i, \quad \text{item } i \text{ that appears in } D_f.$$

$$\textbf{Label} : y_i = \begin{cases} 1, & \text{if } i \text{ appears in } D_b. \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Thus, the goal is to predict whether item i, which appears in D_f, will be consumed in the future (i.e., in the time span of D_b).

We train a parameterized model M under a supervised-learning framework with binary cross entropy loss:

$$L_{IS} = \sum_{i}^{|\mathcal{I}|} y_i \, log(M(f_i;\theta)) + (1 - y_i) \, log(1 - M(f_i;\theta))$$

where θ is the model parameters, and f_i is a feature representing item i. We will demonstrate the details of the feature f_i and model M in the following section.

ISS is defined by the output of the trained model:

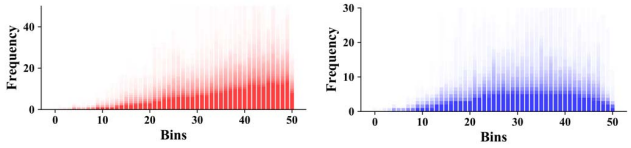
$$p_i = M(f_i;\theta).$$

where $p_i \in \mathbb{R}$ is the ISS of item i in the form of probability. After training the model, we obtain ISSs for all items that appear in data D by building the feature f_i from data D (not from D_f). It is worth noting that we can always perform this task as data D is a common source to train recommender systems.

Predictive Model and Feature Given the classification problem, we introduce the feature f_i and predictive model M as shown in Fig. 2. Intuitively, the consumption pattern of an item over time will be an important clue in determining whether the item will be consumed in the future. To model the consumption patterns of items over time, we represent the timestamps at which an item was consumed as frequency bins such that :

$$\text{item} : [t_1, t_2, \cdots, t_N] \xrightarrow{Binning} [b_1, b_2, \cdots, b_B]$$

where t_j is j-th timestamp at which an item was consumed, and N is the number of consumptions of the item in D_f. In addition, b_k is k-th frequency bin representing the number of times an item was consumed in the period of this bin, and B is the number of bins where $N \gg B$. We set the period of bins as a tunable parameter.



(a) $y_i = 1$ (b) $y_i = 0$

Fig. 3: Distribution of frequency bins corresponding to 10,000 randomly-sampled items that belong to $y_i = 1$ (a) or $y_i = 0$ (b) on Yelp dataset.

To examine the benefit of the frequency bins, in Fig. 3, we show the distribution of the frequency bins that belong to $y_i = 1$ or $y_i = 0$. We can observe that the values in the frequency bins that belong to $y_i = 1$ tend to gradually increase over time (Fig. 3a). In contrast, those that belong to $y_i = 0$ tend to decrease in recent periods (Fig. 3b). Therefore, we should consider the features that capture the consumption patterns changing over time (i.e., the sequence of frequency bins) to predict whether items will be consumed in the future. An alternative feature is to use only the number of consumption in the most recent period (i.e., b_B), but it cannot capture the temporal dynamics of the consumption patterns. We provide the comparison with the straightforward feature (i.e., using only b_B) in the experiment.

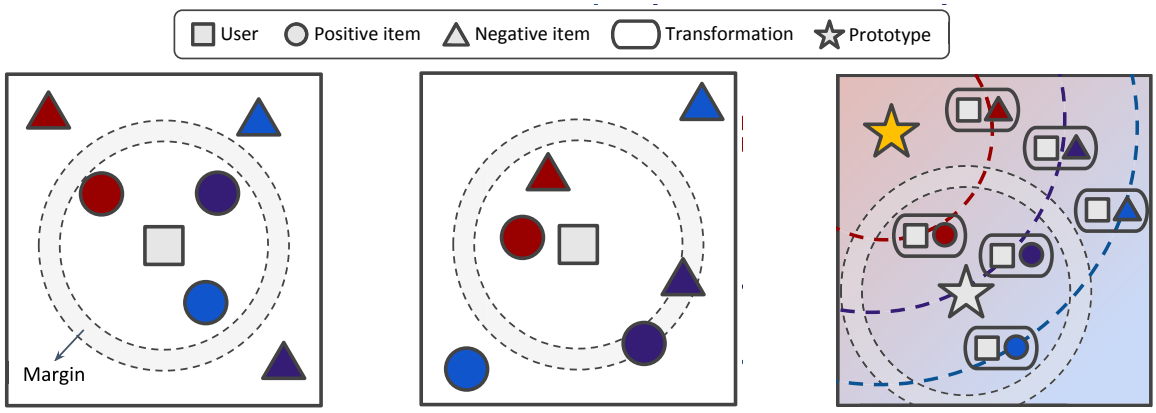
Based on the frequency bins, we design a recurrent neural network (RNN) as a sequence encoder. In this work, we adopt the bidirectional Long Short-Term Memory (BILSTM) (i.e., $\overrightarrow{LSTM}$: original direction, $\overleftarrow{LSTM}$: reverse direction), which has been effective to model sequential data [18]. We design the predictive model with BILSTM as follows:

$$M(f_i;\theta) = \sigma(\mathbf{w}^\top(\overrightarrow{LSTM}(f_i) \parallel \overleftarrow{LSTM}(f_i)) + c) \quad (2)$$

where $f_i = [b_1, b_2, \cdots, b_B] \in \mathbb{R}^B$ is a sequence of frequency bins of item i, σ is the sigmoid function, and $\mathbf{w} \in \mathbb{R}^{2l}$ and $c \in \mathbb{R}$ are a trainable weight and bias. Each LSTM encodes the feature f_i into l-dimensional vector, which is obtained from their last hidden state.

Discussion An issue of this classification problem is that assigning the label of item i that does not appear in D_b as $y_i = 0$ can be a too strong assumption as users are still interested in the item but they may not consume the item in

[2]We note that data D are training data for recommender systems, thus consequently data D do not contain any test datum.



(a) CML (b) CRIS^{reg} (c) CRIS

Fig. 4: Illustration of the representation space learned by CML, CRIS^{reg}, and CRIS. Colors characterize interest sustainability scores of items (red is high and blue is low). Yellow and gray stars represent prototypes corresponding for an interest sustainability score-based objective and a consumption-based objective, respectively.

the period of D_b. We alleviate this limitation by setting the period of D_b to be large enough (e.g., four months). Later in our experiments, we demonstrate the impact of the period of D_b. We can also consider more complex tasks to obtain richer ISS. First, instead of globally assigning labels to items as in (1), we could also consider users and assign labels to each user-item pair, e.g., $y_{u,i} = 1$ if user u consumes item i in the period of D_b. Second, we can formulate a regression problem by setting the label y_i as the count of how many times item i appears in D_b to obtain the expected number of times item i will be consumed in the future. However, in this work, we focus on the original task that relies on binary labels for items as a first step in modeling the ISS, and leave the other tasks for future work.

B. Metric Learning with Interest Sustainability Score

We here return to our original task, i.e., recommendation, with the ISS p_i to model how users' interest in each item will sustain in the future. The basis of the proposed recommender system is a metric learning framework, which makes users closer to items consumed by them (i.e., positive items) than items not consumed by them (i.e., negative items) as shown in Fig. 4a. A consumption-based objective L_C can be defined as follows:

$$L_C(u, i^+, i^-) = [m + d(\mathbf{u}, \mathbf{i}^+) - d(\mathbf{u}, \mathbf{i}^-)]_+ \quad (3)$$

where $[x]_+ = \max(x, 0)$, and $\mathbf{u}, \mathbf{i}^+, \mathbf{i}^- \in \mathbb{R}^K$ are the embedding vectors of user u, positive item i^+, and negative item i^-, respectively. We used euclidean distance as a distance metric d by following [11]. The margin $m \in \mathbb{R}_{>0}$ imposes user u to be closer to the positive item i^+ than the negative item i^- by m in the representation space. Following [11], we impose the space to be a unit sphere by normalizing the embedding vectors (e.g., $\mathbf{u} \leftarrow \mathbf{u}/\max(1, \|\mathbf{u}\|^2)$) for each epoch.

We incorporate the ISS in the above metric learning framework to consider how users' interest in each item will sustain in the future. The underlying idea is to pull items with high ISS to users and to push items with low ISS from users. To this end, we design a ISS-based objective L_S with continuous labels (i.e., p_i):

$$L_S(u, i^+, i^-) = \{(d(\mathbf{u}, \mathbf{i}^+) - d(\mathbf{u}, \mathbf{i}^-)) - (p_{i^-} - p_{i^+})\}^2.$$

The goal of the ISS-based objective L_S is to arrange the items i^+ and i^- by according to the difference of their ISSs (i.e., $p_{i^-} - p_{i^+}$). For example, if the ISS of positive item i^+ is higher than the ISS of negative item i^- (i.e., $p_{i^-} - p_{i^+} < 0$), the objective makes the positive item will be closer to the user than the negative item by $|p_{i^-} - p_{i^+}|$.

A similar approach dealing with the continuous labels is log-ratio loss [19], but it is unstable in our case due to the zero-division of the labels (i.e., p_{i^-}/p_{i^+}). A simple solution that adds a very small number to the ISSs might be still problematic as the ratio of labels can be very large. In contrast, the difference between the labels in our loss is bounded within $-1 \leq p_{i^-} - p_{i^+} \leq 1$ while the representation space is constrained as a unit sphere. Therefore, we can train the recommender system in a more stable manner with the difference loss. We also show the performance comparison between the difference and log-ratio losses in the experiment.

The final loss is a linear combination of both objectives:

$$L = \sum_{(u,i^+) \in P} \sum_{(u,i^-) \notin P} L_C(u, i^+, i^-) + \lambda L_S(u, i^+, i^-). \quad (4)$$

where P is a set of user-item interactions, λ is a balancing coefficient, and L_S acts as a regularization on the metric learning framework. Given the combination of both objectives, the metric learning method can build a representation space with considering both whether users liked items (by L_C) and how users' interest in the items sustain in the future (by L_S). We name this method as CRIS^{reg}.

Prototype Learning A limitation in the metric learning framework with the ISS is that there can be potential conflicts

between two objectives because an anchor (i.e., a user) is shared to optimize both objectives, L_C and L_S. For example, a positive item of a user can have low ISS, thus consequently the positive item can be distant from the user (Fig. 4b). Therefore, modeling the ISS can prevent the recommendation system from fully learning the user's preference for items.

To alleviate such conflicts, we further extend the metric learning framework with prototypes [8], which are trainable points in the representation space. In this work, we design each prototype to be responsible for optimizing one objective. The intuition is to disentangle two objectives by using two types of anchors (i.e., prototypes) instead of a single type of anchors (i.e., users).

We first define two prototypes in the representation space:

$$\mathcal{C}, \mathcal{S} \in \mathbb{R}^K \tag{5}$$

where $\mathcal{C}$ is a prototype for optimizing the consumption objective L_C and $\mathcal{S}$ is another prototype for optimizing the interest sustainability objective L_S. We then project a user-item pair into a single point such that:

$$\mathcal{T}_{u,i} = \mathbf{u} + \mathbf{i} \tag{6}$$

where $\mathcal{T}$ is a transformation function and we use sum operation. Different transformations such as a neural network are also possible and we investigate the effect of the transformations in the experiment.

Given two prototypes, we reformulate the objectives of CRIS^{reg} as follows:

$$L_C^P(u, i^+, i^-) = [m + d(\mathcal{C}, \mathcal{T}_{u,i^+}) - d(\mathcal{C}, \mathcal{T}_{u,i^-})]_+$$

$$L_S^P(u, i^+, i^-) = \{(d(\mathcal{S}, \mathcal{T}_{u,i^+}) - d(\mathcal{S}, \mathcal{T}_{u,i^-})) - (p_{i^-} - p_{i^+})\}^2$$

Based on the prototypes, the consumption loss L_C^P makes the pair of a user and the user's positive item (i.e., $\mathcal{T}_{u,i^+}$) closer to prototype $\mathcal{C}$ than the pair of the user and the user's negative item (i.e., $\mathcal{T}_{u,i^-}$). Similarly, the ISS-based objective L_S^P makes the pair of a user and an item with high ISS closer to prototype $\mathcal{S}$ than items with low ISS. Each objective is optimized with a corresponding prototype for the objective compared to the shared anchors (i.e., users) in (4). Therefore, as shown in Fig. 4c, the recommender system can optimize both objectives with less conflicts between them than the approach of CRIS^{reg}.

We combine the prototype-based objectives with a balancing coefficient λ:

$$L^P(\theta) = \sum_{(u,i^+)\in P} \sum_{(u,i^-)\notin P} L_C^P(u, i^+, i^-) + \lambda L_S^P(u, i^+, i^-) \tag{7}$$

We can train the recommender system by minimizing the loss by using stochastic gradient descent (SGD) w.r.t. the parameters θ (i.e., $\min_\theta L^P(\theta)$).

Under the prototype-based learning, a recommendation score of user u on item i is as follow:

$$Score(u, i) = -\{d(\mathcal{C}, \mathcal{T}_{u,i}) + \gamma d(\mathcal{S}, \mathcal{T}_{u,i})\} \tag{8}$$

where γ is a tunable parameter to control the importance of the ISS in the recommendation score.

TABLE I: Data Statistics. Int. denotes user-item interactions.

Data	# Users	# Items	# Int.(M)	Avg. Int. per user	Period
Tools	16,472	10,177	0.133	7.7	Nov 1999 - Jul 2014
Toys	19,153	11,865	0.165	8.3	Jul 2000 - Jul 2014
Cell Phones	27,372	10,279	0.190	6.5	Feb 2001 - Jul 2014
Clothing	38,651	22,974	0.274	6.6	Mar 2003 - Jul 2014
Sports	34,974	18,294	0.291	7.9	Mar 2002 - Jul 2014
Health	37,842	18,358	0.339	8.4	Dec 2000 - Jul 2014
Kindle	67,193	58,110	0.935	12.7	Mar 2000 - Jul 2014
CDs	74,926	64,342	1.093	14.4	Nov 1997 - Jul 2014
Movies	122,923	49,976	1.688	13.3	Nov 1997 - Jul 2014
Yelp	47,906	78,734	2.304	47.2	Oct 2004 - Nov 2018
GoodReads	58,003	45,330	2.791	47.5	Feb 2001 - Nov 2017

IV. Experiments

A. Dataset

We compared CRIS with state-of-the-art recommender systems to examine the effectiveness of the ISS. The concept drift of users might vary over domains so that we select a variety of datasets from Amazon[3], Yelp[4], and GoodReads[5] (Table I). Amazon datasets have been a benchmark to evaluate the performance of recommender systems [4]–[6]. We select the following domains with different sizes: tools, toys, cell phones, clothing, sports, health, kindle, CDs, and movies. Yelp dataset contains users' interaction with businesses such as restaurants, hotels, and gyms. GoodReads dataset consists of users' interactions with books, and we select the comic category. We filtered out noisy data from Yelp and GoodReads datasets by maintaining only users who made at least 10 interactions and items that were involved to at least 5 interactions as done in [4]. We used the Amazon datasets per se since the preprocessing has already been performed on the datasets so that users and items have at least 5 interactions.

B. Evaluation Protocol

We used user-item interactions in the latest one month (i.e., 30 days) as test data, and the others as training data by following [20]. We then set the user-item interactions in the latest one month in the training data as validation data. Following [7], we also removed users and items, which do not appear in the training data, from the validation and test data as the cold-start issue is out of scope of our current work, and has generally been covered as a special issue [21]–[23]. We evaluated the rank of a positive item with 100 randomly-selected negative items to avoid heavy computations as done in [5], [6]. As metrics, we adopt hit ratio (H@k) and normalized discounted cumulative gain (N@k) to evaluate the ranking performance of recommender systems. H@k measures whether a positive item is ranked in the top-k recommended items. N@k assigns higher scores to positive items for higher positions in the top-k recommended items. Due to space

[3]http://jmcauley.ucsd.edu/data/amazon

[4]https://www.yelp.com/dataset

[5]https://sites.google.com/eng.ucsd.edu/ucsdbookgraph

TABLE II: Performance comparison. Δ_H and Δ_S are the relative improvements (%) of CRIS over HGN and SML, respectively, with the statistical significance $p < 0.001$ computed using the paired t-test.

Dataset	Metric	BPR	CML	SML	NTF	Caser	SASRec	TiSASRec	HGN	CRIS^{reg}	CRIS$^{wt.}$	CRIS	Δ_H	Δ_S
Tools	H@10	0.3314	0.3649	0.3740	0.3449	0.3301	0.3044	0.3264	0.3605	0.3804	0.3953	**0.4047**	12.3	8.2
	N@10	0.1818	0.2009	0.2016	0.1951	0.1858	0.1660	0.1795	0.2061	0.2118	0.2190	**0.2276**	10.4	12.9
Toys	H@10	0.3586	0.3881	0.3906	0.3496	0.3426	0.3352	0.3352	0.3848	0.4275	0.4586	**0.4602**	19.6	17.8
	N@10	0.2155	0.2306	0.2343	0.197	0.1924	0.1870	0.1831	0.2269	0.2561	0.2656	**0.2726**	20.1	16.3
Cell Phones	H@10	0.4278	0.4547	0.4709	0.5315	0.4711	0.4659	0.4793	0.4763	0.5300	0.4620	**0.5642**	18.5	19.8
	N@10	0.2675	0.2825	0.2901	0.3190	0.2899	0.2790	0.2930	0.3037	0.3203	0.2863	**0.3416**	12.5	17.8
Clothing	H@10	0.3657	0.4073	0.4121	0.3809	0.3443	0.3421	0.3340	0.3912	0.4254	0.4016	**0.4473**	14.3	8.5
	N@10	0.2149	0.2437	0.2443	0.2117	0.1990	0.1959	0.1878	0.2339	0.2511	0.2394	**0.2652**	13.4	8.6
Sports	H@10	0.4458	0.4909	0.4914	0.4256	0.4366	0.4250	0.4216	0.4659	0.4877	0.4857	**0.5171**	11.0	5.2
	N@10	0.2637	0.2891	0.2887	0.2433	0.2566	0.2469	0.2430	0.2823	0.2853	0.2878	**0.3056**	8.3	5.9
Health	H@10	0.4239	0.4713	0.4746	0.4431	0.4336	0.4272	0.4396	0.4586	0.4804	0.4728	**0.4985**	8.7	5.0
	N@10	0.2501	0.2843	0.2835	0.2717	0.2639	0.2487	0.2632	0.2972	0.2972	0.2825	**0.3056**	2.8	7.8
Kindle	H@10	0.7136	0.7235	0.7235	0.5945	0.6403	0.6082	0.6497	0.7083	0.7603	0.7214	**0.787**1	11.1	8.8
	N@10	0.4672	0.4829	0.4834	0.3541	0.4019	0.3748	0.4141	0.4759	0.5171	0.4805	**0.5462**	14.8	13.0
CDs	H@10	0.6959	0.7104	0.7046	0.6426	0.5815	0.5826	0.6107	0.6591	0.7189	0.6727	**0.7389**	12.1	4.9
	N@10	0.4470	0.4610	0.4585	0.4003	0.3513	0.3563	0.3764	0.4289	0.4782	0.4404	**0.4931**	15.0	7.5
Movies	H@10	0.6938	0.7024	0.7020	0.6785	0.6421	0.6597	0.6553	0.6771	0.7056	0.6951	**0.7250**	7.1	3.3
	N@10	0.4504	0.4543	0.4544	0.4428	0.4111	0.4234	0.4244	0.4549	0.4582	0.4570	**0.4686**	3.0	3.1
Yelp	H@10	0.8715	0.8853	0.8857	0.8348	0.8052	0.8383	0.8701	0.8658	0.8928	0.8861	**0.9070**	4.8	2.4
	N@10	0.6031	0.6305	0.6294	0.5578	0.5146	0.5503	0.5829	0.5969	0.6138	0.6300	**0.6630**	11.1	5.3
GoodReads	H@10	0.7442	0.7541	0.7518	0.7243	0.6997	0.6437	0.7219	0.7381	0.7559	0.7576	**0.7920**	7.3	5.3
	N@10	0.5005	0.5115	0.5105	0.4906	0.4892	0.4293	0.5067	0.5308	0.5032	0.5144	**0.5377**	1.3	5.3

limitation, we report results with $k = 10$, which has been a major configuration in the previous work [4]–[7]. We ran recommender systems 5 times and report the averaged results.

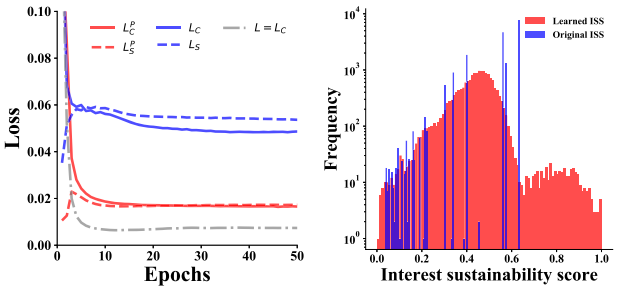
C. Methods Compared

- **BPR** [10] is a conventional and popular recommender system in top-k recommendation. We used MF as a model with the learning objective of BPR.
- **CML** [11] is a metric learning method, which models users' preference to items with a metric (e.g., euclidean distance) instead of inner product.
- **SML** [14] is a state-of-the-art metric learning method enhancing CML by including a item-centric metric and trainable margins for each user and item.
- **NTF** [20] utilizes timestamps of user-item interactions to capture user's periodical behaviors by extending tensor factorization [24] with a neural network.
- **Caser** [7] is a sequential recommender system based on CNNs to extract local features from the sequence of users' consumption.
- **SASRec** [6] considers pair-wise interactions between items in the sequence of users' consumption using a self-attention mechanism [15].
- **TiSASRec** [5] extends SASRec by exploiting the time intervals between two consecutive items in the sequence of user's consumption.
- **HGN** [4] is a state-of-the-art sequential recommender system based on a hierarchical gating network to better capture both long- and short-term user's interactions.
- **CRIS**^{reg} is a method that models the ISSs on the metric learning framework as a regularization approach based on (4).
- **CRIS**$^{wt.}$ is a straightforward method to utilize the ISS with the metric learning method. It uses the ISS only when computing the recommendation score such as $Score(u, i) = (p_i)^{\lambda} \cdot (-d(u, i))$ while training the model with only L_C (i.e., CML).
- **CRIS** is a method that models the ISS on the metric learning framework, while incorporating two prototypes to relieve the conflicts between the objectives (i.e., L_C and L_S) based on (7).

We also note that the popularity-based recommender system, which simply recommends the most frequent items in training data, was consistently worse than the baselines methods in our experiment. In this respect, we omit the comparison of the popularity-based method to save space.

D. Implementation Details

We used Adam [25] as an optimizer for all methods. The learning rate and mini-batch size were tuned in $\{0.0001, 0.001, 0.01\}$ and $\{128, 256, 1024, 2048, 4096\}$, respectively. The dimension of user, item, and prototype embeddings K was tuned in $\{10, 20, 30, 40, 50\}$. For the baseline methods, we tuned their architecture-specific tunable parameters (e.g., the number of filters in CNN or the number of fully-connected layers) as reported in their paper. In case of CRIS, we tuned the balancing coefficients such as $\lambda \in \{0.01, 0.1, 0.2, 0.3, 0.4\}$, and $\gamma \in \{0.4, 0.8, 1.2, 1.6, 2.0\}$. We



(a) Convergence comparison of CRIS, CRIS^{reg}, and CML. (b) Histogram of original and learned ISSs.

Fig. 5: Analysis on variants of CRIS on Health dataset.

implemented all the methods in PyTorch [26] framework for the sake of fair comparison. We used the average of H@10 and N@10 of each method on the validation data as a criterion to tune the parameters. It is worth noting that we initialized the embedding parameters in all methods (e.g., users and items embeddings) with Xavier initialization [27]. We observed that initializing the embeddings vectors to small weights (e.g., using Xavier initialization) is important to obtain better and stable performance in most methods.

E. Recommendation Performance Analysis

1) Performance Comparison with Baseline Methods: Table II tabulates the ranking performances of the recommender systems including the proposed method, CRIS. We have the following observations: 1) CRIS consistently shows the best performance compared to the other baseline methods on 11 real-world datasets. Its relative improvements on H@10 and N@10 are 11.5% and 10.1% against HGN, and 8.5% and 10.3% against SML on average. This result indicates the importance of the ISS to boost the accuracy of recommendations. 2) Notably, the metric learning-based methods (i.e., SML and CML) show the best performance among the the baseline methods despite using only user-item interaction information. These results suggest future work in modeling the additional information (e.g., a sequence of items users consumed) should consider these recommender systems as baseline methods. To the best of our knowledge, this work is first to extensively compare the metric learning methods to the sequential recommender systems. 3) Among sequential recommender systems, HGN is the best performing method, and achieves the best performance on some datasets in terms of N@10. 4) NTF, which utilizes timestamps of user-item interactions, shows the best performance in a dataset compared to other baseline methods. 5) BPR shows competitive performance compared to other baseline methods, in contrast to the previous work [4], [6], [7]. This observation is related to a recent concern that the conventional methods are not fully tuned in the literature [28]. In our experiments, the important factors for the performance of BPR are the initialization of user and item embeddings and tuning $L2$ regularization coefficient.

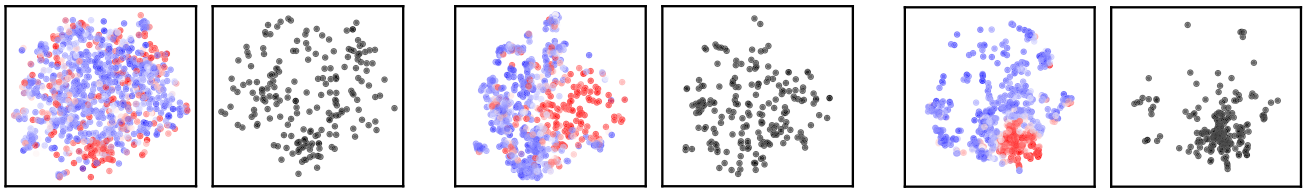
TABLE III: Ablation study. Rand., LogR., and NN denote random ISS, log-ratio loss, and neural network, respectively.

Dataset	Metric	Only $\mathcal{C}$	Only $\mathcal{S}$	Rand.	Oracle	LogR.	NN	CRIS
Toys	H@10	0.385	0.352	0.361	0.652	0.441	0.401	0.460
	N@10	0.233	0.175	0.199	0.377	0.249	0.219	0.273
Clothing	H@10	0.401	0.320	0.371	0.543	0.456	0.291	0.447
	N@10	0.240	0.157	0.207	0.309	0.266	0.158	0.265
Health	H@10	0.460	0.237	0.411	0.538	0.480	0.414	0.499
	N@10	0.279	0.132	0.249	0.325	0.291	0.235	0.306
Movies	H@10	0.705	0.554	0.678	0.793	0.724	0.690	0.725
	N@10	0.458	0.299	0.446	0.518	0.481	0.433	0.469
Yelp	H@10	0.887	0.429	0.890	0.961	0.904	0.879	0.906
	N@10	0.632	0.193	0.629	0.738	0.643	0.617	0.657

2) Comparison with Variants of CRIS: The variants of CRIS (i.e., CRIS^{reg} and CRIS$^{wt.}$) are better than the baseline methods on some datasets. However, CRIS consistently shows better performance than the variants. This observation signifies the adequate way of modeling the ISS is important to boost the accuracy of recommendations. We provide an analysis for each variant to deeply understand the benefits of CRIS. 1) In Fig. 5a, we show the training convergence for each loss in CRIS and CRIS^{reg} along with a consumption-based loss (i.e., $L = L_C$) that is optimized without the ISS-based loss (i.e., CML). We can observe that the losses of CRIS converge at a lower point than those of CRIS^{reg}, which means the prototypes are indeed helpful to reduce the conflicts between two objectives (i.e., L_C and L_S). 2) In Fig. 5b, we compare the original ISSs (i.e., p_i) and the ISSs learned by CRIS (we denote it as $\hat{p}_i$). To obtain the ISSs learned by CRIS, we first compute the distance $d(\mathcal{S}, \mathcal{T}_{u,i})$ between the interest-sustainability prototype $\mathcal{S}$ and all pairs of users and items, $\mathcal{T}_{u,i}$. Then, we average the distances for each item and take min-max normalization on the averaged distances for items to ensure the normalized values $x_i \ \forall i$ are within $x_i \in [0, 1]$. Lastly, we take the complement of the values (i.e., $\hat{p}_i \leftarrow 1 - x_i$) to obtain the ISSs learned by CRIS. We observe that the ISSs learned by CRIS tends to follow the original ISSs, but the learned ISSs are smoother than the original ISSs. We conjecture the original ISSs can be inaccurate as we will see in Section IV-I, thus consequently CRIS learns to reduce the noise of the original ISSs. Therefore, CRIS is more robust to the noise of the ISSs than the CRIS$^{wt.}$, which utilizes the original ISSs per se.

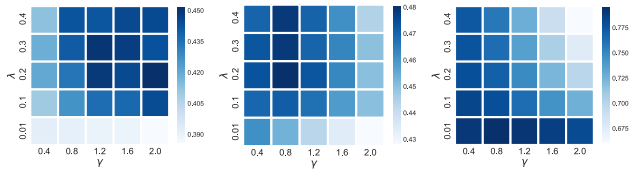
F. Ablation Study

In this study, we demonstrate the impact of each design choice of CRIS (Table III). 1) A method (Only $\mathcal{C}$) that is trained only on the consumption-based loss, L_C^P, shows similar performances of CML in Table II. This observation is natural because the goal of both losses (L_C^P and L_C) is identical as learning users' preference based on the user-item interactions. 2) Similarly, the method (Only $\mathcal{S}$) that is trained only on the ISS-based loss, L_S^P, shows the consistent degradation in the accuracy of recommendations. This result indicates we should utilize the ISSs along with the consumption-based



(a) HGN (b) SML (c) CRIS

Fig. 6: Visualization of randomly-sampled 1,000 item representations learned by SML, HGN, and CRIS from Movies dataset using t-SNE. In left figure for each method, a point represents an item that appears in the training data and color represents an interest sustainability score of each item (red is high and blue is low). In right figure for each method, a point represents an item that appears in the test time.



(a) Toys (b) Health (c) Yelp

Fig. 7: Sensitivity analysis on λ and γ.

objective, since modeling the ISS alone cannot capture user's personalized preferences. 3) Next, we investigate the accuracy of recommendations according to the quality of the ISSs. Randomly-initialized ISSs (Rand.) hurt the model performance in all the cases as the representations of users and items will be wrongly learned by the random ISSs. 4) In contrast, as an oracle, we set ISS p_i as 1 if item i appear in test data and 0 else. The oracle shows much superior performances compared to CRIS. We may not be possible to obtain the oracle, but it provides the upper bound of the recommendation performance of CRIS. 5) CRIS with the log-ratio loss (LogR.) shows slightly worse performances (-1.91% degradation on average) than the CRIS with the difference loss due to the unstable behavior when $p_i = 0$. 6) We also examine a neural transformation (i.e., a fully-connect layer) to project a pair of a user and an item into a point in the representation space instead of the sum operation. On all the datasets, the neural transformation is worse than the sum-based transformation. We suspect the additional parameters of the neural network make the method to overfit compared to the parameter-free approach (i.e., sum).

G. Comparison of Learned Representations

In Fig. 6, we visualize the item representations learned by SML, HGN, and CRIS to investigate whether they can consider the interest sustainability of items. We randomly sampled 1,000 items and extract their representations (i.e., embedding vectors) from each method after training. We colorize the item representations (i.e., points in the left figure for each method) with the ISSs, which are obtained from our approach. Among the 1,000 items, we plot the items that appear in test time in the right figure for each method. We observe that the baseline methods (i.e., HGN and SML) suffer from distinguishing the items with respect to their ISSs. Thus, the items that appear in test time spread over the representation space. This result indicates that modeling only user-item interactions is limited to capture whether each item will be consumed in the future. In contrast, CRIS successfully captures the interest sustainability of items thanks to the ISSs of items. As a result, the items that appear in test time are more clearly clustered than those of the baselines. Thus, we conclude that the ISSs of items are essential signal that enables the recommender system to consider how users' interest in each item will sustain in the future.

H. Effect of Balancing Coefficients

Fig. 7 illustrates the sensitivity of the balancing coefficients, λ and γ, in CRIS based on its performance (i.e., average of H@10 and N@10) on the validation data. we have the following observations: 1) CRIS achieves the best performance with small λ (less than 0.5) but large γ (about 1.0 on average) on the datasets, which indicates the importance of the ISSs differs between on training time and evaluation time. We conjecture that the inconsistency between training and evaluation time is caused by the noise in the ISSs. As we will see in Section IV-I, the accuracy of the neural classifier is not very high so that the ISSs obtained from the classifier cannot perfectly represent how users' interest in items will sustain. Thus, while training, CRIS depends less on the ISS-based objective L_S^P to avoid overfitting to the noisy ISSs, which reduces the noise in the ISSs as we observed in Fig. 5b. As a consequence, in the evaluation time, CRIS largely depends on the denoised ISSs when determining the recommendation scores. From this observation, we can conclude that CRIS can handle the noise in the ISSs by adjusting the balancing coefficients λ and γ. 2) The value of the best-performing λ is less than 0.5, which reaffirms that the ISSs should be modeled with the consumption-based objective (i.e., L_C^P) to learn the users' personalized preferences.

I. Performance on Interest Sustainability Prediction

Hereafter, we demonstrate the experiment results on the interest sustainability prediction. Table IV shows the class

TABLE IV: Class distribution of datasets for the interest sustainability prediction. Positive and Negative denote, among items in training data, the numbers of items that appear in test data and those of items that do not appear in test data, respectively.

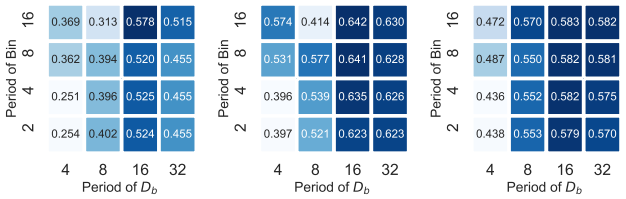
Dataset	Positive	Negative	Dataset	Positive	Negative
Tools	2,085	8,092	Kindle	14,522	43,588
Toys	1,736	10,129	CDs	4,278	60,064
Cell	2,603	7,676	Movies	10,352	39,624
Clothing	5,923	17,051	Yelp	13,551	65,183
Sports	4,784	13,510	GoodReads	6,580	38,750
Health	5,333	13,025			

TABLE V: F1-score on the interest sustainability prediction. FCL and GoodR. denote a fully-connected layer and GoodReads dataset, respectively. The method taking only the last bin as input is denoted as b_B.

Data	BILSTM	b_B	FCL	Data	BILSTM	b_B	FCL
Tools	**0.426**	0.343	0.403	Kindle	**0.590**	0.515	0.525
Toys	**0.383**	0.223	0.346	Cds	**0.333**	0.125	0.308
Cell	**0.589**	0.524	0.564	Movies	**0.569**	0.493	0.571
Clo.	**0.534**	0.475	0.484	Yelp	**0.544**	0.403	0.527
Sports	**0.497**	0.325	0.488	GoodR.	**0.501**	0.498	0.493
Health	**0.538**	0.423	0.512				

distributions of the test data. We note that the class distributions of the training data change by the period of data D_b, and thus we report only the statistics of the test data. We tuned the tunable parameters in the BILSTM-based classifier on the validation data, which are used for the recommender systems. The periods of the frequency bins and data D_b were tuned in $\{2, 4, 8, 16\}$ and $\{4, 8, 16, 32\}$ weeks, respectively. The dimension of the hidden state in LSTM was tuned in $\{64, 128, 256\}$. We used Adam with 0.01 as its learning rate and 128 as the mini-batch size, which show the stable performance over the datasets. We also tuned the class weight for positive instances in $\{0.01, 0.1, 0.5, 1.0\}$ to address the class imbalance problem (Table IV), and set the value as 0.01, which was the best-performing value over the datasets. We used F1-score to measure the classification performance instead of accuracy due to the severe class imbalance (e.g., 92.8% accuracy by random guessing on CDs dataset).

Table V reports the classification performance on the datasets. We have observations as follows. 1) The proposed BILSTM-based classifier consistently shows the best classification results on the datasets. However, even with the best method, the classification performances are not very high on the datasets. Accordingly, the ISS obtained from the method can be noisy. Despite the noises, The ISSs are helpful signal to enhance the recommendation performance as we can see in Table II. We can also expect the better classifier on the interest sustainability prediction can further enhance the recommendation performance of CRIS by reducing the noisy in the ISSs. Thus, this observation suggests designing better classifiers on this classification problem is promising as a future work. 2) The proposed classifier taking only the last frequency bin as input (denoted as b_B) shows the lowest performance on most datasets. In addition, the fully-connected layer (FCL), which ignores sequential consumption patterns, shows worse performance than BILSTM. From these results from the variants, we can conclude the sequential features (i.e. sequences of frequency bins) and the corresponding sequence encoder are effective to predict the interest sustainability of items.



(a) Clothing (b) Kindle (c) GoodReads

Fig. 8: Sensitivity analysis on the periods of data D_b and frequency bins. The numbers in both axes denote the number of weeks.

J. *Effect of Periods*

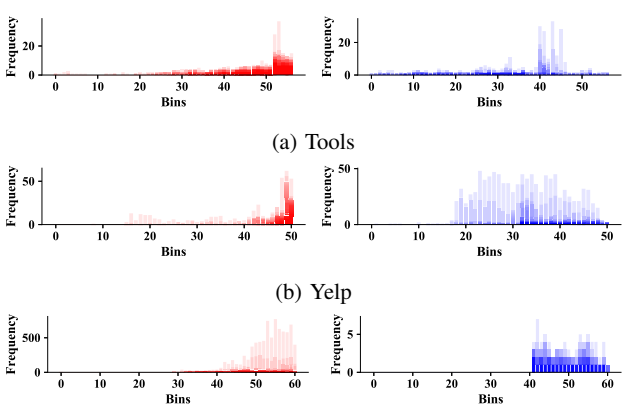
Fig. 8 illustrates the classification performance with respect to the periods of data D_b and the frequency bins. First, the performances are sensitive to the period of D_b, and long periods (e.g., 16 weeks) show the best performance. From this observation, we speculate that the period of data D_b should be long enough to reliably determine whether an item will be consumed in the future. Second, the long period of the frequency bins generally shows better classification performances. We presume that if the periods of the frequency bins are too short, the fluctuation of values in the bins becomes severe, which makes features of items noisy. Therefore, adjusting these two periods is essential to successfully predicting the interest sustainability of items.

K. *Pattern Analysis in Item Features*

In this experiment, we probe the behavior of the BILSTM-based classifer. In Fig. 9, we show the distribution of features assigned by the neural classifier as the positive or negative class. We select the features associated with the most confident 100 predictions for each class. The frequency of the features in the positive class tends to gradually increase over time, whereas the frequency of the features in the negative class tends to decrease over time. These observations reassert the necessity of the sequential feature (i.e., sequences of frequency bins) and the sequence encoder (i.e., BILSTM) to capture the temporal dynamics of users' consumption patterns.

V. Conclusion

In this work, we build a recommender system based on the interest sustainability score (ISS) of items to consider how users' interest in each will sustain in the future. We first predict the interest sustainability of items to obtain the



(a) Tools

(b) Yelp

(c) GoodReads

Fig. 9: Distribution of features associated with the most confident 100 predictions in each class. Left and right figures represent the features classified as positive and negative, respectively.

ISS for each item based on a neural classifier. Afterward, we build a recommender system based on the metric learning framework with the ISSs of items to capture the concept drift of users. Experimental results on the real-world datasets, the proposed recommender system (CRIS) achieves the state-of-the-art performance compared to the baseline methods. Furthermore, through in-depth analyses, we reveal that the ISSs are indeed crucial to boost the accuracy of recommendations.

VI. Acknowledgement

This research was supported by the MSIT(Ministry of Science, ICT), Korea, under the High-Potential Individuals Global Training Program)(NO.IITP-2020-0-01649) supervised by the IITP(Institute for Information & Communications Technology Planning & Evaluation) and it has been supported by Microsoft Research.

References

[1] B. Smith and G. Linden, "Two decades of recommender systems at amazon. com," *IEEE Internet Computing*, 2017.

[2] A. Tsymbal, "The problem of concept drift: definitions and related work," *Computer Science Department, Trinity College Dublin*, vol. 106, no. 2, p. 58, 2004.

[3] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of the 19th international conference on World wide web*, 2010, pp. 811–820.

[4] C. Ma, P. Kang, and X. Liu, "Hierarchical gating networks for sequential recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 825–833.

[5] J. Li, Y. Wang, and J. McAuley, "Time interval aware self-attention for sequential recommendation," in *Proceedings of the 13th International Conference on Web Search and Data Mining*, 2020, pp. 322–330.

[6] W.-C. Kang and J. McAuley, "Self-attentive sequential recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 197–206.

[7] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 565–573.

[8] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Advances in neural information processing systems*, 2017, pp. 4077–4087.

[9] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Advances in neural information processing systems*, 2008, pp. 1257–1264.

[10] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," *arXiv preprint arXiv:1205.2618*, 2012.

[11] C.-K. Hsieh, L. Yang, Y. Cui, T.-Y. Lin, S. Belongie, and D. Estrin, "Collaborative metric learning," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 193–201.

[12] Y. Tay, L. Anh Tuan, and S. C. Hui, "Latent relational metric learning via memory-based attention for collaborative ranking," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 729–739.

[13] C. Park, D. Kim, X. Xie, and H. Yu, "Collaborative translational metric learning," in *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 367–376.

[14] M. Li, S. Zhang, F. Zhu, W. Qian, L. Zang, J. Han, and S. Hu, "Symmetric metric learning with adaptive margin for recommendation," in *2020 AAAI Conference on Artificial Intelligence (AAAI)*.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[16] C. Liu, R. W. White, and S. Dumais, "Understanding web browsing behaviors through weibull analysis of dwell time," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, 2010, pp. 379–386.

[17] H. Jing and A. J. Smola, "Neural survival recommender," in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, 2017, pp. 515–524.

[18] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," in *International Conference on Artificial Neural Networks*. Springer, 2005, pp. 799–804.

[19] S. Kim, M. Seo, I. Laptev, M. Cho, and S. Kwak, "Deep metric learning beyond binary supervision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2288–2297.

[20] X. Wu, B. Shi, Y. Dong, C. Huang, and N. V. Chawla, "Neural tensor factorization for temporal interaction learning," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019, pp. 537–545.

[21] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 278–288.

[22] M. Volkovs, G. W. Yu, and T. Poutanen, "Content-based neighbor models for cold start in recommender systems," in *Proceedings of the Recommender Systems Challenge 2017*, 2017, pp. 1–6.

[23] R. He and J. McAuley, "Fusing similarity models with markov chains for sparse sequential recommendation," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 2016, pp. 191–200.

[24] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[25] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[27] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[28] M. F. Dacrema, P. Cremonesi, and D. Jannach, "Are we really making much progress? a worrying analysis of recent neural recommendation approaches," in *Proceedings of the 13th ACM Conference on Recommender Systems*, 2019, pp. 101–109.