Intent Contrastive Learning with Cross Subsequences for Sequential Recommendation

Xiuyuan Qin*
Soochow University, China
20215227016@stu.suda.edu.cn

Huanhuan Yuan*
Soochow University, China
hhyuan@stu.suda.edu.cn

Pengpeng Zhao†
Soochow University, China
ppzhao@suda.edu.cn

Guanfeng Liu
Macquarie University, Australia
guanfeng.liu@mq.edu.au

Fuzhen Zhuang
Institute of Artificial Intelligence & SKLSDE, School of Computer Science
Beihang University, China
zhuangfuzhen@buaa.edu.cn

Victor Sheng
Texas Tech University, United States
victor.sheng@ttu.edu

ABSTRACT

The user purchase behaviors are mainly influenced by their intentions (e.g., buying clothes for decoration, buying brushes for painting, etc.). Modeling a user's latent intention can significantly improve the performance of recommendations. Previous works model users' intentions by considering the predefined label in auxiliary information or introducing stochastic data augmentation to learn purposes in the latent space. However, the auxiliary information is sparse and not always available for recommender systems, and introducing stochastic data augmentation may introduce noise and thus change the intentions hidden in the sequence. Therefore, leveraging user intentions for sequential recommendation (SR) can be challenging because they are frequently varied and unobserved. In this paper, Intent contrastive learning with Cross Subsequences for sequential Recommendation (ICSRec) is proposed to model users' latent intentions. Specifically, ICSRec first segments a user's sequential behaviors into multiple subsequences by using a dynamic sliding operation and takes these subsequences into the encoder to generate the representations for the user's intentions. To tackle the problem of no explicit labels for purposes, ICSRec assumes different subsequences with the same target item may represent the same intention and proposes a coarse-grain intent contrastive learning to push these subsequences closer. Then, fine-grain intent contrastive learning is mentioned to capture the fine-grain intentions of subsequences in sequential behaviors. Extensive experiments conducted on four real-world datasets demonstrate the superior performance of the proposed ICSRec[1] model compared with baseline methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

*These authors contributed equally to this work.
†Corresponding author.
[1]Our code is available at https://github.com/QinHsiu/ICSRec.

WSDM '24, March 4–8, 2024, Merida, Mexico.

ACM ISBN 979-8-4007-0371-3/24/03…$15.00
https://doi.org/10.1145/XXXXXX.XXXXXX

ACM Reference Format:
Xiuyuan Qin, Huanhuan Yuan, Pengpeng Zhao, Guanfeng Liu, Fuzhen Zhuang, and Victor Sheng. 2024. Intent Contrastive Learning with Cross Subsequences for Sequential Recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM '24), March 4–8, 2024, Merida, Mexico.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/XXXXXX.XXXXXX

1 INTRODUCTION

Recommender systems are widely studied in academia and industry for their ability to efficiently alleviate data overload by accurately capturing users' preferences and providing personalized recommendation suggestions [1, 32, 49]. Sequential Recommendation (SR) [8, 15, 34, 40, 48], one of the best recommendation models for dynamically modeling user preferences, focus on predicting the next item that users are likely to interact with based on their chronological behaviors.

Users' purchase behaviors mainly depend on their intentions (e.g., buying a watch for decoration, buying books for reading, etc.) [5, 21]. Leveraging such intentions for SR can significantly improve the performance and robustness [20], and this has attracted widespread attention. The methods for modeling intentions can be divided into two categories based on whether or not to use auxiliary information (e.g., user action types, item category information, etc.). Some exciting works [2, 20, 38] model users' intentions by constructing an auxiliary task (e.g., predicting the next item's category, predicting the user's next action type, etc.). Since such information is more sparse, does not always accurately reflect the user's intentions, and may not be readily accessible [5], some other exciting works choose to capture user intentions in the latent space. DSSRec [26] proposes a seq2seq training strategy, which infers the intentions based on individual sequence representation via clustering. SINE [35] proposes a sparse interest extraction module to adaptively infer the interacted intentions of a user from a large pool of intention groups. ICLRec [5] obtains the intent prototype representation from the embedded space of all user behavior sequences via clustering and builds a contrastive learning task to leverage the learned intentions into the SR model. IOCRec [21] extracts intent representations from two stochastic augmented views for contrastive learning and further improves the performance by alleviating the noise problem.

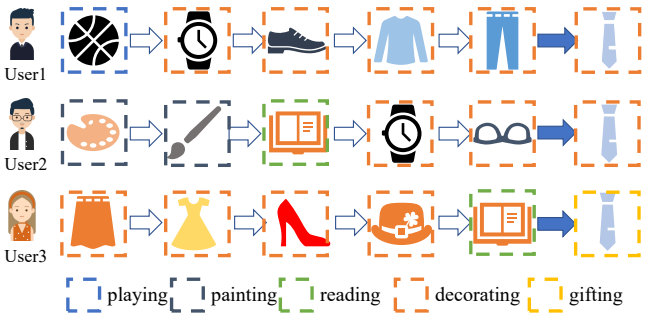

Figure 1: An example showing different interaction sequences exist with subsequences of the same target item.

While the above attempts to construct contrastive learning tasks to learn the user's intentions, they usually model each user interaction sequence individually and thus ignore the correlation between users with similar subsequence patterns [50]. Considering the example illustrated in Figure 1, User1 and User2 may purchase the item 'watch' for decorating at different steps. Meanwhile, User2 and User3 may buy the item 'book' for learning at different steps. Different users may have the same intention to purchase the same item at different moments. However, modeling sequences as a whole does not fully utilize the intent supervisory signals hidden in the history of different user interactions. Furthermore, the intention to buy the same item in different contexts may be different [25]. As shown in Figure 1, all three Users buy the item 'tie'. User3 may buy it as a gift, but User1 and User2 may buy it for decoration. Ignoring this restriction may result in the model's failure to learn the representation of the user's intention, thus decreasing the performance.

To address the above issues, we propose a novel model named ICSRec (i.e., Intent contrastive learning with Cross subsequences for Sequential Recommendation), which utilizes cross subsequences patterns to construct intent supervisory signals for intent representation learning. Specially, we first process the original training sequence into several subsequences via a split operation, and we put subsequences with the same target item into the same set to construct coarse-grain intent supervisory signals. ICSRec assumes that two different subsequences with the same target item have the same intention. Therefore, a coarse-grain intent contrastive learning method is introduced to put the two subsequences with the same intention closer. To alleviate the problem that the same item may represent different intentions in different contexts, we further propose a fine-grain intent contrastive leaning to make the subsequence closer to its intention prototype obtained by clustering. Finally, ICSRec achieves a significant improvement (average increase of 24%) compared with the State-Of-The-Art baselines on all datasets.

We summarize the contributions of this work below:

- ICSRec first utilizes a similar pattern hidden in cross subsequences for intent representation learning in the SR task.
- We propose two modules (i.e., coarse-grain intent learning module and fine-grain intent learning module) to model the user's intention from different dimensions.
- Extensive experiments demonstrate that ICSRec achieves state-of-the-art performances on four publicity datasets.

2 THE PROPOSED MODEL

2.1 Problem Definition

Sequential Recommendation (SR) is to recommend the next item that the user will interact with based on his/her historical interaction data. Assuming that U and I are the sets of users and items, $u \in U$ has a sequence of interacted items $S^u = \{s_1^u, s_2^u, ..., s_{|S^u|}^u\}$, and $s_t^u \in \mathcal{I}(1 \le t \le |S^u|)$ represents an interacted item at position t of user u within the sequence, where $|S^u|$ denotes the sequence length. Given the historical interactions S^u, the goal of SR is to recommend an item from the set of items $\mathcal{I}$ that the user u may interact with at the $|S^u|+1$ step, which can be formulated as follows:

$$\arg\max_{i \in \mathcal{I}} P(s_{|S^u|+1}^u = i | S^u). \tag{1}$$

2.2 Overall Framework

Figure 2 shows the overall framework of ICSRec. It mainly contains three parts, 1) Supervisory Signal Construction, 2) Intent Representation Learning, and 3) Predict Layer. In the first part, we first segment all training sequences into multiple subsequences by the $DS(\cdot)$ operation via Eq.(2). Then, we put all subsequences with the same target item into a set to construct coarse-grain intent supervisory signals. Next, we set up two auxiliary tasks for intent representation learning: Coarse-grain Intent Contrastive Learning (CICL) and Fine-grain Intent Contrastive Learning (FICL). In the CICL task, for each subsequence, we randomly sample another subsequence from the set as its positive sample with the same target item. Then, we directly make the coarse-grain intent representation of the two subsequences in the latent space closer. In the FICL task, we first cluster the coarse-grain intent representations obtained from all subsequences and then make the coarse-grain intent representation closer to its fine-grain intent prototype. In addition, we utilize the learned intent representation to predict the next item that the user may interact with. Finally, we jointly optimize these three objectives.

2.3 Supervisory Signal Construction

To investigate the same subsequence patterns across user interaction sequences, we first split the original sequence into multiple subsequences via $DS(\cdot)$ operation [36]. Specially, given a sequence $S^u = \{s_1^u, s_2^u, ..., s_{|S^u|}^u\}$, the operation can be formulated as follows:

$$DS(S^u) = \begin{cases} \{\{s_1^u, i_2^u\}, \cdots, \{s_1^u, \\ \cdots, s_{|S^u|-1}^u, i_{|S^u|}^u\}\} & |S^u| \le n+1 \\ \\ \{DS(S_{n+1}^u), \cdots, \\ \{s_{|S^u|-n}^u, \cdots, i_{|S^u|}^u\}\} & |S^u| > n+1. \end{cases} \tag{2}$$

where n represents the max sequence length and $i_k^u (1 \le k \le |S^u|)$ represents the target item of the subsequence. After the segmentation operation on all training sequences, we put all obtained

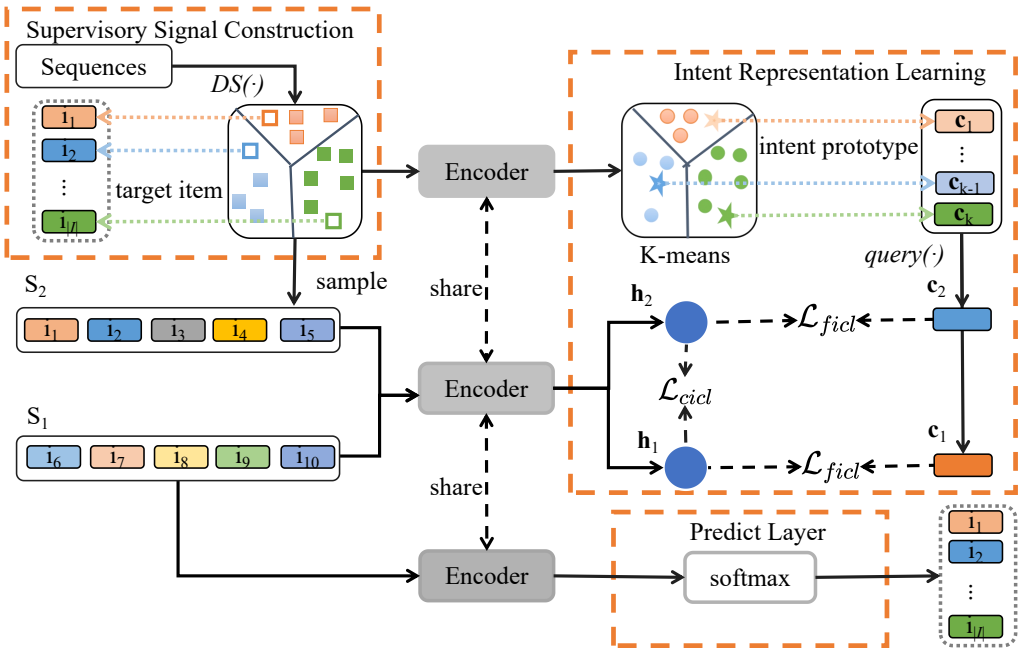

Figure 2: The model architecture of ICSRec. Where S_2 and S_1 denote two subsequences with the same target item. $\mathbf{h}_2$ and $\mathbf{h}_1$ denote two coarse-grain intentions obtained by encoder $f_\theta(\cdot)$. $\mathbf{c}_2$ and $\mathbf{c}_1$ denote two fine-grain intentions obtained via clustering.

subsequences into different sets $T = \{T_1, T_2, ..., T_{|I|}\}$, where T_a represents the subsequence set with the target item $i_a \in I (1 \le a \le |I|)$, to construct the coarse-grain intent supervisory signals.

2.4 Encoder

Firstly, the whole item sets $\mathcal{I}$ are embedded into the same space [15, 34] and generate the item embedding matrix $\mathbf{M} \in \mathbb{R}^{|\mathcal{I}| \times d}$. Given the input sequence S^u, the embedding of the sequence S^u is initialized to $\mathbf{e}^u \in \mathbb{R}^{n \times d}$, which can formulated as follows:

$$\mathbf{e}^u = \mathbf{m}^u + \mathbf{p}^u, \tag{3}$$

where $\mathbf{m}^u \in \mathbb{R}^{n \times d}$, represents the item's embedding, $\mathbf{p}^u \in \mathbb{R}^{n \times d}$ represents the position embedding and n represents the length of the sequence, respectively.

Due to its superiority in modeling sequential tasks, we choose SASRec [15] as the backbone model, represented as $f_\theta(\cdot)$, which uses the Transformer [39] to learn the representation of the sequence. Given the sequence embedding $\mathbf{e}^u$, the output representation $\mathbf{H}^u \in \mathbb{R}^{n \times d}$ is calculated as:

$$\mathbf{H}^u = f_\theta(\mathbf{e}^u). \tag{4}$$

where θ represents the parameters of the sequential model. The last vector $\mathbf{h}_n^u \in \mathbb{R}^d$ in $\mathbf{H}^u = [\mathbf{h}_1^u, ..., \mathbf{h}_n^u]$ is chosen as the intent representation of the sequence.

2.5 Intent Representation Learning

Coarse-grain Intent Contrastive Learning. Since different subsequences with the same target item may represent the same intention. Thus, we directly bring the intent representations of two subsequences with the same target item closer together in the latent space via contrastive learning. Given a subsequence S_1 with the target item i_a, we first randomly sample a subsequence S_2 from T_a, and then calculate the coarse-grain intent representations, $\mathbf{h}_1$ and $\mathbf{h}_2$, of two subsequences by Eq.(4), respectively. Then, we utilize contrast learning to bring two coarse-force intent representations closer to each other in the latent space. The contrastive loss can be formulated as follows:

$$\mathcal{L}_{cicl} = \mathcal{L}_{con}(\mathbf{h}_1, \mathbf{h}_2), \tag{5}$$

and

$$\begin{aligned} \mathcal{L}_{con}(\mathbf{x}^1, \mathbf{x}^2) = &- \log \frac{e^{s(\mathbf{x}^1, \mathbf{x}^2)/\tau}}{e^{s(\mathbf{x}^1, \mathbf{x}^2)/\tau} + \sum\limits_{\mathbf{x} \notin \mathcal{F}} e^{s(\mathbf{x}^1, \mathbf{x})/\tau}} \\ &- \log \frac{e^{s(\mathbf{x}^2, \mathbf{x}^1)/\tau}}{e^{s(\mathbf{x}^2, \mathbf{x}^1)/\tau} + \sum\limits_{\mathbf{x} \notin \mathcal{F}} e^{s(\mathbf{x}^2, \mathbf{x})/\tau}}. \end{aligned} \tag{6}$$

where $(\mathbf{x}^1, \mathbf{x}^2)$ represents a pair of positive sample's embedding, $s(\cdot)$ represents the inner product, τ is the temperature parameter, and $\mathcal{F}$ is a set of negative samples that have the same label with two positive pairs in the mini-batch. Specially, we do not use InfoNCE [3, 11] directly to calculate the contrastive loss, because treating other 2(|**B**|-1) views within the same batch as negative samples may

introduce false negative problems (e.g., there have more than one pair of subsequences that have the same target item in a mini-batch.), where $|\mathbf{B}|$ denotes the batch size. So we propose a simple strategy called False Negative Mask (FNM) to mask the effects by not contrasting them [5].

Fine-grain Intent Contrastive Learning. Since the intention to buy the same item may fall into different categories in different contexts. Thus, we assume that there are K types of users' intentions and that the intention to purchase the same item in different contexts can potentially belong to different types. We first put all subsequences into the encoder to obtain all the coarse-grain intent representations via Eq.(4). Then we use all the obtained outputs for K-means clustering via faiss [14] and treat the center of the clusters as the fine-grain intent representation. The intention prototype we obtained is represented as $\mathbf{C} = \{\mathbf{c}^k\}_{k=1}^K$, where $\mathbf{c}^k \in \mathbb{R}^d$ represents the k-th intention prototype. Then, we obtain the fine-grain intent representation of the two subsequences by $query(\cdot)$ operation. Specially, we choose the nearest intention prototype $\mathbf{c}_1$ to $\mathbf{h}_1$ and $\mathbf{c}_2$ to $\mathbf{h}_2$, respectively, from the set $\mathbf{C} = \{\mathbf{c}^k\}_{k=1}^K$ as the fine-grain intent representation of $\mathbf{h}_1$ and $\mathbf{h}_2$, which is as follows:

$$\mathbf{c}_1, \mathbf{c}_2 = query(\mathbf{h}_1, \mathbf{C}), query(\mathbf{h}_2, \mathbf{C}). \tag{7}$$

To avoid introducing false negative problems (e.g., there have more than one pair of subsequences that have the same fine-grain intent representation in a mini-batch.). We use Eq.(6) to calculate the contrastive loss as follows:

$$\mathcal{L}_{ficl} = \mathcal{L}_{con}(\mathbf{h}_1, \mathbf{c}_1) + \mathcal{L}_{con}(\mathbf{h}_2, \mathbf{c}_2). \tag{8}$$

2.6 Prediction Layer

In SR, the prediction of the next item can be viewed as a classification task based on the set of all items. Therefore, we use the learned intent representation to calculate the probability of the next item that the user may interact with. Given the learned intent representation $\mathbf{h}^u \in \mathbb{R}^d$ and the item embedding matrix $\mathbf{M}$, the Eq.(1) is equivalent to minimizing the cross-entropy loss as follows:

$$\mathcal{L}_{Rec} = -1 * \hat{\mathbf{y}}[g] + \log(\sum_{i}^{I} \exp(\hat{\mathbf{y}}[i]))), \tag{9}$$

and

$$\hat{\mathbf{y}} = softmax(\mathbf{h}^u \mathbf{M}^T). \tag{10}$$

where $\hat{\mathbf{y}} \in \mathbb{R}^{|I|}$ represents the predictive score of all items, and $g \in I$ represents the ground-truth of the sequence.

2.7 Multi Task Learning

We use a multi-task learning paradigm to jointly optimize the main sequential prediction task and the other two auxiliary learning objectives. In which Eq.(9) is to optimize the main next item predict task, Eq.(5) is to optimize the CICL task, and Eq.(8) is to optimize the FICL task. The final training loss function is as follows:

$$\mathcal{L} = \mathcal{L}_{Rec} + \lambda \cdot \mathcal{L}_{cicl} + \beta \cdot \mathcal{L}_{ficl}. \tag{11}$$

where λ and β are hyper-parameters that need to be tuned.

Dataset	Sports	Beauty	Toys	ML-1M
# Users	35,598	22,363	19,412	6,040
# Items	18,357	12,101	11,924	3,416
# Actions	296,337	198,502	167,597	999,611
# Avg. Actions/User	8.3	8.8	8.6	165.4
# Avg. Actions/Item	16.1	16.4	14	292.6
Sparsity	99.95%	99.93%	99.93%	95.16%

Table 1: Statistics of the experimented datasets.

3 EXPERIMENTS

In this section, we present our experimental setup and empirical results. Our experiments are designed to investigate the following research questions(**RQ**s):

- **RQ1:** How does ICSRec perform compared to the State-Of-The-Art (SOTA) SR models?
- **RQ2:** How effective are the key model components (e.g., intent representation learning, FNM) in ICSRec?
- **RQ3:** How does the robustness (e.g., hyper-parameters sensitivity, against noise interactions) of ICSRec?

3.1 Experimental Settings

Datasets. We evaluate the effectiveness of our proposed model on four datasets from two real-world applications:

- **Amazon**[2]**:** A series of datasets that collect user reviews on products from *Amazon.com*. The dataset can be divided into many subsets according to the various product categories, which have relatively short sequence lengths. In this paper, we pick **Sports**, **Beauty**, and **Toys** as three different experimental datasets from the Amazon dataset.
- **MovieLens**[3]**:** It contains users' behavior logs on movies, which have very long sequences. We pick the version MovieLens-1M (**ML-1M**), which includes 1 million user ratings, as an experimental dataset.

To ensure the data quality, users or items appearing less than five times are removed [15, 34]. For all datasets, we aggregate each user's interaction records and sort them by the action timestamps in chronological order. For evaluation purposes, we follow [5, 29] to split the data into training, validation, and testing datasets based on timestamps given in the datasets. Specifically, the last item is used as the label for testing, the second-to-last item is used as the label for validating, and the others for training. Table 1 presents the detailed statistics of the four datasets.

Baseline Models. We compare ICSRec with the following representative SR models:

- **Non-sequential model: BPR** [30] first employs Bayesian Personalized Ranking (BPR) loss to optimize the matrix factorization model.
- **General sequential models: GRU4Rec** [13] introduces a Gated Recurrent Unit (GRU) to model sequences and first leverages Recurrent Neural Networks (RNN) for SR. **Caser** [37] first introduces Convolutional Neural Network (CNN) to SR, which

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

Table 2: Performance comparisons of different methods. The results of the best baseline are underlined in each row. The last column is the relative improvements compared with the best baseline results.

DataSet	Metric	BPR	GRU4Rec	Caser	SASRec	BERT4Rec	S^3-Rec$_{MIP}$	CL4SRec	CoSeRec	DuoRec	DSSRec	SINE	ICLRec	IOCRec	ICSRec	impro.
Sports	HR@5	0.0123	0.0162	0.0154	0.0214	0.0217	0.0121	0.0231	0.0290	0.0312	0.0209	0.0240	0.0290	0.0293	**0.0403**	29.17%
	HR@10	0.0215	0.0258	0.0261	0.0333	0.0359	0.0205	0.0369	0.0439	0.0466	0.0328	0.0389	0.0437	0.0452	**0.0565**	21.24%
	HR@20	0.0369	0.0421	0.0399	0.0500	0.0604	0.0344	0.0557	0.0636	0.0696	0.0499	0.0610	0.0646	0.0684	**0.0794**	14.10 %
	NDCG@5	0.0076	0.0103	0.0114	0.0144	0.0143	0.0084	0.0146	0.0196	0.0195	0.0139	0.0152	0.0191	0.0169	**0.0283**	44.39 %
	NDCG@10	0.0105	0.0142	0.0135	0.0177	0.0190	0.0111	0.0191	0.0244	0.0244	0.0178	0.0199	0.0238	0.0220	**0.0335**	37.30%
	NDCG@20	0.0144	0.0186	0.0178	0.0218	0.0251	0.0146	0.0238	0.0293	0.0302	0.0221	0.0255	0.0291	0.0279	**0.0393**	30.13%
Beauty	HR@5	0.0178	0.0180	0.0251	0.0377	0.0360	0.0189	0.0401	0.0504	0.0561	0.0408	0.0354	0.0500	0.0511	**0.0698**	24.42%
	HR@10	0.0296	0.0284	0.0342	0.0624	0.0601	0.0307	0.0642	0.0725	0.0851	0.0616	0.0612	0.0744	0.0774	**0.0960**	12.81%
	HR@20	0.0474	0.0478	0.0643	0.0894	0.0984	0.0487	0.0974	0.1034	0.1228	0.0894	0.0963	0.1058	0.1146	**0.1298**	5.70%
	NDCG@5	0.0109	0.0116	0.0145	0.0241	0.0216	0.0115	0.0268	0.0339	0.0348	0.0263	0.0213	0.0326	0.0311	**0.0494**	41.95%
	NDCG@10	0.0147	0.0150	0.0226	0.0342	0.0300	0.0153	0.0345	0.0410	0.0441	0.0329	0.0296	0.0403	0.0396	**0.0579**	31.29%
	NDCG@20	0.0192	0.0186	0.0298	0.0386	0.0391	0.0198	0.0428	0.0487	0.0536	0.0399	0.0384	0.0483	0.0490	**0.0663**	23.69%
Toys	HR@5	0.0122	0.0121	0.0205	0.0429	0.0371	0.0456	0.0503	0.0533	0.0655	0.0447	0.0385	0.0597	0.0542	**0.0788**	20.30%
	HR@10	0.0197	0.0184	0.0333	0.0652	0.0524	0.0689	0.0736	0.0755	0.0959	0.0671	0.0631	0.0834	0.0804	**0.1055**	10.01%
	HR@20	0.0327	0.0290	0.0542	0.0957	0.0760	0.0940	0.0990	0.1037	0.1293	0.0942	0.0957	0.1139	0.1132	**0.1368**	5.80%
	NDCG@5	0.0076	0.0077	0.0125	0.0245	0.0259	0.0314	0.0264	0.0370	0.0392	0.0297	0.0225	0.0404	0.0297	**0.0571**	41.34%
	NDCG@10	0.0100	0.0097	0.0168	0.0320	0.0309	0.0388	0.0339	0.0442	0.0490	0.0369	0.0304	0.0480	0.0381	**0.0657**	34.08%
	NDCG@20	0.0132	0.0123	0.0221	0.0397	0.0368	0.0452	0.0404	0.0513	0.0574	0.0437	0.0386	0.0557	0.0464	**0.0736**	28.22%
ML-1M	HR@5	0.0247	0.0806	0.0912	0.1078	0.1308	0.1078	0.1142	0.1128	0.2098	0.1371	0.0990	0.1382	0.1796	**0.2445**	16.54%
	HR@10	0.0412	0.1344	0.1442	0.1810	0.2219	0.1952	0.1815	0.1861	0.3078	0.2243	0.1694	0.2273	0.2689	**0.3368**	9.42%
	HR@20	0.0750	0.2081	0.2228	0.2745	0.3354	0.3114	0.2818	0.2950	0.4098	0.3275	0.2705	0.3368	0.3831	**0.4518**	10.25%
	NDCG@5	0.0159	0.0475	0.0565	0.0681	0.0804	0.0616	0.0705	0.0692	0.1433	0.0898	0.0586	0.0889	0.1201	**0.1710**	19.33%
	NDCG@10	0.0212	0.0649	0.0734	0.0948	0.1097	0.0917	0.0920	0.0915	0.1749	0.1179	0.0812	0.1175	0.1487	**0.2007**	14.75%
	NDCG@20	0.0297	0.0834	0.0931	0.1156	0.1384	0.1204	0.1170	0.1247	0.2007	0.1440	0.1066	0.1450	0.1775	**0.2297**	14.45%

Table 3: The HR@20 and NDCG@20 performances achieved by ICSRec variants and SASRec on four datasets.

Model	Dataset							
	Sports		Beauty		Toys		ML-1M	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
(A) ICSRec	**0.0794**	**0.0393**	**0.1298**	**0.0663**	**0.1368**	**0.0736**	**0.4518**	**0.2297**
(B) w/o CICL	0.0754	0.0374	0.1268	0.0627	0.1293	0.0701	0.4468	0.2174
(C) w/o FICL	0.0746	0.0368	0.1252	0.0616	0.1346	0.0716	0.4455	0.2162
(D) w/o FNM	0.0786	0.0384	0.1290	0.0658	0.1336	0.0718	0.4315	0.2157
(E) SASRec	0.0500	0.0218	0.0894	0.0386	0.0957	0.0397	0.2745	0.1156

leverages both horizontal and vertical convolution to model the sequence. **SASRec** [15] first utilizes the attention mechanism to model sequences, which greatly improves the performance of SR.

- **SSL-based sequential models: BERT4Rec** [34] first introduces BERT [6] to model the sequence, which leverages MIP (Mask Item Predict) task to capture the potential relationships between items and sequences. **S^3-Rec$_{MIP}$** [51] first introduces self-supervised learning to capture the potential relationships between items. And since we have no attribute information, only the MIP task, called S^3-Rec$_{MIP}$, is used for training. **CL4SRec** [47] first integrates data augmentation and contrastive learning to SR, which further improves the robustness of the model to noise and sparsity **CoSeRec** [23] further improves CL4SRec by introducing two more robust data augmentation. **DuoRec** [29] introduces a model augmentation method and a supervised sampling strategy for the first time.
- **Intent-guided sequential models: DSSRec** [26] first introduces a novel seq2seq training strategy and an intention-disentanglement layer for SR. **SINE** [35] designs an adaptive interest aggregation module to model users' multiple interests for SR. **ICLRec** [5] learns users' latent intents from the behavior sequences via clustering and integrates the learned intents into the model via an auxiliary contrastive loss. **IOCRec** [21] first introduces intent-level contrastive learning for denoising problems of the SR task.

Evaluation Metrics. We follow [17, 42] to rank the whole item set without negative sampling and use two widely-used evaluation metrics to evaluate the model, including Hit Ratio @k (HR@k) and Normalized Discounted Cumulative Gain @k (NDCG@k) where $k \in \{5, 10, 20\}$.

Implementation Details. The implementations of Caser, S^3-Rec, BERT4Rec, CoSeRec, ICLRec, DuoRec, and IOCRec are provided by the authors. BPR, GRU4Rec, SASRec, CL4SRec, DSSRec, and SINE are implemented based on public resources. All parameters in these methods are used as reported in their papers and the optimal settings are chosen based on the model performance on validation data. For ICSRec, the number of the self-attention blocks and attention heads is set as 2. The batch size is set to 256. We set d as 64, n as 50, τ as 1.0. λ and β are selected from $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. We use Adam [16] as the optimizer and set the learning rate to 10^{-3}. The number of clusters is set in the range of $\{64, 128, 256, 512, 1024\}$ and the dropout rate in a range of $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. We train the model with an early stopping strategy based on the performance of validation data. All experiments are implied on NVIDIA GeForce RTX 2080 Ti GPU.

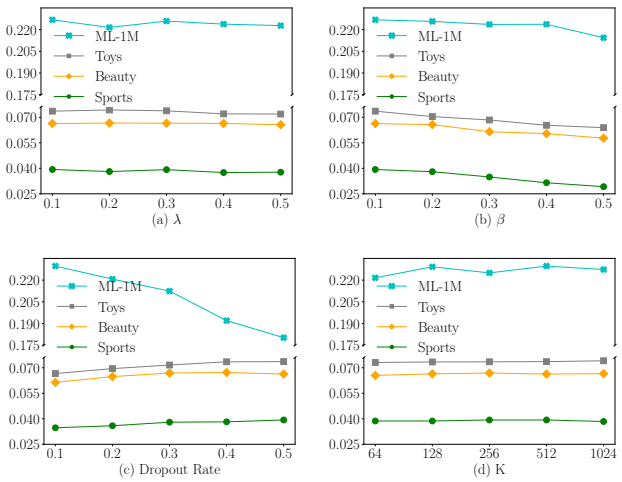

Figure 3: Performances of ICSRec w.r.t. different hyper-parameters (λ, β, dropout rate and K). on NDCG@20.

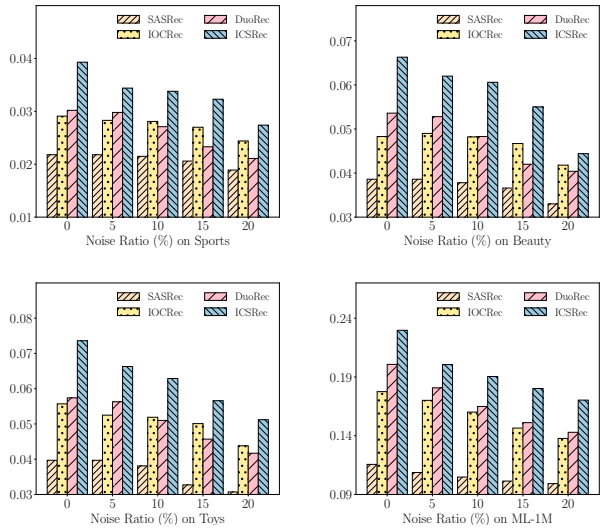

Figure 4: Performance comparison w.r.t. noise ratio on four datasets. The bar chart shows the performance in NDCG@20.

3.2 Performance Comparison (RQ1)

We compare the performance of all baselines with ICSRec. Table 2 shows the experimental results of all the compared models on four datasets, and the following findings can be seen through it:

- Models based on the attention mechanism are better than models based on other networks, such as GRU4Rec and Caser, which demonstrates the superiority of the attention mechanism in modeling sequential tasks. In addition, the self-supervised based models perform more effectively than classical models, such as SASRec. Among them, different from BERT4Rec and S^3-Rec$_{MIP}$ that use MIP tasks to train the model, CL4SRec, CoSeRec, and DuoRec utilize data augmentation and Contrastive Learning (CL) for training, which leads to generally better results than BERT4Rec and S^3-Rec$_{MIP}$. That indicates the CL paradigm may generate more expressive embeddings for users and items by maximizing the mutual information.
- The intent-based models perform better than most self-supervised models, such as BERT4Rec, S^3-Rec$_{MIP}$, and CL4SRec. Among them, ICLRec and IOCRec achieve better results in this group. Different from DSSRec and SINE, ICLRec and IOCRec utilize a contrastive SSL task to learn the intent representation for SR, and thus improve the performance by a large margin. But they are not as effective as DuoRec, probably because introducing stochastic data augmentation destroys the intention of the original sequences and thus reduces the performance [10]. That motivates us to further investigate how to better combine CL and intent modeling for SR.
- Benefiting from introducing intent supervisory signal for intent representation learning, ICSRec significantly outperforms other methods on all metrics across the different datasets. For instance, ICSRec improves over the best baseline result w.r.t. NDCG by 23.69-44.39% and 14.45-19.33% on three sparse datasets and the dense dataset ML-1M, respectively. The reason may be that the user's interaction sequences are longer in dense dataset ML-1M, which makes the users' intentions more diverse and less easy to distinguish.

Table 4: The HR@20 and NDCG@20 performances of GRU4Rec, SASRec, and ICSRec with different sequence encoders.

Model	Dataset							
	Sports		Beauty		Toys		ML-1M	
	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG
(A) GRU4Rec	0.0421	0.0186	0.0478	0.0186	0.0290	0.0123	0.2081	0.0834
(B) ICSRec$_{GRU}$	0.0595	0.0280	0.1014	0.0506	0.0950	0.0510	0.4045	0.1964
(C) SASRec	0.0500	0.0218	0.0894	0.0386	0.0957	0.0397	0.2745	0.1156
(D) ICSRec$_{SAS}$	**0.0794**	**0.0393**	**0.1298**	**0.0663**	**0.1368**	**0.0736**	**0.4518**	**0.2297**

3.3 Ablation Study (RQ2)

To analyze the effectiveness of each component of our model, we conduct several ablation experiments about ICSRec, where w/o denotes without. (A) denotes ICSRec, (B) removes the coarse-grain intent learning module by setting λ to 0 in Eq.(11), (C) removes the fine-grain intent learning module by setting β to 0 in Eq.(11), and (D) removes the false negative mask component in Eq.(6). (E) denotes the base encoder SASRec [15].

Tabel 3 summarizes HR@20 and NDCG@20 performances of ICSRec variants and SASRec on four datasets. From the table, we can find that ICSRec achieves the best results on all datasets, which indicates all components are effective for our framework. By comparing (A) with (B) and (C), we find that CICL and FICL could significantly improve the model accuracy, which is consistent with our statements. By comparing (B) and (C), it can be observed that FICL is more efficient than CICL. By comparing (A) and (D), we can find that the mask of the false negative sample could improve the

User1	{185, T}	{3307, D&Co}	{1287, D&A}	{2283, D}	{1363, D}	{1252, T}	{1276, D&Co}	{3007, Do}	{627, D&T}	{3791, D}	{ 2579, D}
User2	{3889, A}	{1991, H}	{3525, Co}	{2335, Co}	{2920, D}	{265, D}	{3773, Co}	{381, D}	{3246, D}	{3028, Co}	
User3	{546, A&C}	{1431, A&Co}	{2412, A&D}	{2153, A}	{694, A}	{3113, A&T}	{1497, A}	{1772, A&Co}	{2487, A&Co}	{2568, A}	{2720, A&C}
User4	{2142, C&Co}	{2162, C}	{126, C}	{546, C&A}	{631, C&M}	{87, C&Co}	{1702, C&Co}	{575, C&Co}	{3054, C}	{1822, C&Co}	

Table 5: The case study explains the motivation of our proposed model. The digit means the movie id from the ML-1M datasets. The capital letter means the film genre (A: Action; Co: Comedy; C: Children; D: Drama; T: Thriller; M: Musical; Do: Documentary; H: Horror). User1 and User2, User3 and User4 have the same target movie.

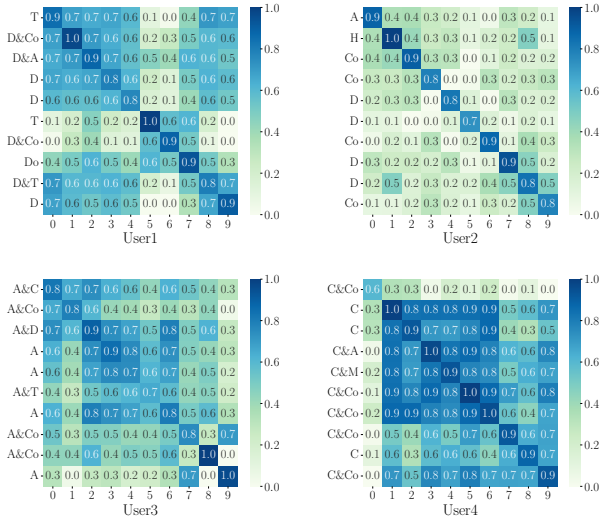

Figure 5: Visualization of the intents heat map for four users.

performance. By comparing (A) and (E), we can find that ICSRec outperforms the backbone model SASRec on four datasets. This indicates that leveraging the intent supervisory signals, obtained from cross subsequences, for intent representation learning can help improve SR performance.

3.4 Further Analysis (RQ3)

We conduct various experiments on four datasets to verify the robustness of ICSRec. For all models in the following experiments, we only change one variable at one time while keeping other hyper-parameters optimal.

Impact of $\mathcal{L}_{cicl}$ and $\mathcal{L}_{ficl}$ weights. The final loss function of ICSRec in Eq.(11) is a multi-task learning loss. Figure 3(a) and (b) shows the effect of assigning different weights to λ and β on the model, respectively. We observe that the performance of ICSRec gets peak value to different β and λ, which demonstrates the effectiveness of the proposed framework and manifests that introducing suitable weights can boost the performance of recommendation. From these figures, $\beta = 0.1$ and $\lambda = 0.3$ for Sports, $\beta = 0.1$ and $\lambda = 0.3$ for Beauty, $\beta = 0.1$ and $\lambda = 0.2$ for Toys, and $\beta = 0.1$ and $\lambda = 0.1$ for ML-1M are generally proper to ICSRec. The weight of $\mathcal{L}_{cicl}$, i.e., λ, is commonly larger than β, which demonstrates that CICL gains more importance than FICL.

Impact of the dropout rate. The dropout can be seen as a regularization technique to mitigate model overfitting, and the magnitude of its value is closely related to the data [33, 43]. From Figure 3(c), dropout rate=0.5 for three sparse datasets and dropout rate=0.1 for the ML-1M dataset are generally proper to ICSRec. The reason may be that the ML-1M dataset contains longer sequences and the loss of more neurons affects the modeling of user intention.

Impact of the cluster number K. Figure 3(d) shows the effect of different numbers of intention prototype clusters on the model. This figure shows that the number of users' intentions in different datasets is different, which shows the diversity of users' intentions. In addition, K=256 for Sports, K=256 for Beauty, K=1024 for Toys, and K=512 for ML-1M are generally proper to ICSRec.

Robustness to Noise Data. To verify the robustness of ICSRec against noise interactions, we randomly add a certain proportion(i.e, 5%, 10%, 15%, 20%) of negative items into the input sequences during testing, and examine the final performance of ICSRec and other baselines. From Figure 4 we can see that adding noise data deteriorates the performances of four models. By comparing SASRec and other models, it can be seen that adding a contrastive auxiliary task can significantly improve the model's robustness. By comparing ICSRec and other models, it can be seen that our model performs better than other models. Especially, with 10% noise proportion, our model can even outperform other models without noise dataset on four datasets. The reason might be that ICSRec leverages contrastive learning to model intentions more accurately from cross subsequences, thus increasing the robustness. By looking at the performance of the four models on sparse datasets (sports, beauty, and toys), it can be seen that SASRec and DuoRec are much lower than the intent-based models IOCRec and ICSRec when the proportion of noise data is 20. This shows the robustness of the model can be improved by leveraging the user's intention on sparse datasets. In addition, comparing the performances of the four models on the dense dataset ml-1m, we can see that IOCRec performs less robustly than DuoRec on ML-1M, probably because users' intentions are more diverse in dense datasets, and IOCRec introduces stochastic data augmentation to model the intention, which may destroy the user's original intentions [10].

Impacts of encoder $f_\theta(\cdot)$. To further investigate the effectiveness of our proposed methods, we use other models, such GRU4Rec, as the backbone encoder. Specially, we consider the following settings of ICSRec as the backbone encoder for experiments: (B) ICSRec$_{GRU}$: we use the GRU4Rec [13] as the backbone encoder, (D) ICSRec$_{SAS}$: the default model that uses SASRec [15] as the backbone encoder. Table 4 shows the performance of ICSRec with different encoders, as well as the performance of backbone models. It can be seen that

(B) and (D) outperform the respective backbone encoder models. This indicates that the intent representation learning module is a general module that can help improve the performance of existing SR methods. Comparing (A) and (C), (B) and (D), we can find that the encoder dominates the performance of ICSRec, and the intent representation learning module is a complementary part that can help further improve the performance.

4 RELATED WORK

4.1 Sequential Recommendation

Sequential recommendation (SR) aims to predict successive preferences according to one's historical interactions, which has been heavily researched in academia and industry [9, 40]. Classical FPMC [31] fuses both sequential patterns and users' general interests. FOSSIL [12] improves the robustness against sparse data by combining similarity-based methods and high-order Markov Chains. With the recent advancements in deep learning, many SR models are gradually being combined with neural networks [2, 44]. Such as Recurrent Neural Networks (RNN) based [13, 44] and Convolutional Neural Networks (CNN) based [37]. The recent success of Transformer [39] advances the development of SR. SASRec [15] first utilizes a transformer to model the users' ordered historical interactions, which has made an immense success. LSAN [22] devises a temporal context-aware embedding and proposes a twine-attention sequential framework. STOSA [8] introduces stochastic embeddings and proposes a Wasserstein Self-Attention in SR. However, these models are commonly limited by sparse and noisy problems.

4.2 Self-supervised Learning for Sequential Recommendation

Motivated by the immense success of Self-Supervised Learning (SSL) in the field of Natural Language Process (NLP) [6] and Computer Vision (CV) [7] and its effectiveness in solving data sparsity problems, a growing number of works are now applying SSL to recommendation. BERT4Rec [34] leverages pre-trained BERT to generate the representation of the target item from the user's historical interactions. S^3-Rec [51] utilizes four auxiliary self-supervised tasks to capture the sequential information by maximizing the mutual information of different views. More interesting works [23, 27, 29, 47] introduce Contrastive Learning (CL), which leverages the information from augmented views to boost the effectiveness of learned representations [48], into SR to alleviate the noise and sparse problems. MMInfoRec [28] applies an end-to-end contrastive learning scheme for feature-based SR. CL4SRec [47] learns the representations of users by maximizing the agreement between different augmented views of the same user's chronological interactions and optimizing the contrastive loss with the main task simultaneously. CoSeRec [23] further improves CL4SRec by introducing two more robust data augmentation methods for generating contrastive pairs. More prior works [45, 46, 52] also explore the application of SSL to graph-based recommendation. SGL [45] generates two augmented views with graph augmentation and optimizes the node-level contrastive loss. Different from constructing views by adopting data augmentation, DuoRec [29] chooses to construct view pairs with model augmentation, which could maintain the sequential correlations in the process of training. However, these augmentation operations are all hand-crafted and may change the original intention behind the sequence.

4.3 Intent-guided Recommender Systems

Many recent approaches have turned their attention to studying users' intentions to improve the performance and robustness of recommender systems [2, 4, 18, 19, 41]. Some exciting works [2, 20, 38] construct a complementary task for learning the user's intentions by introducing auxiliary information (e.g., predicting the next item's category, predicting the user's next action type, etc.). Since such information may not always be available or truly express the user's intentions, some interesting works have been proposed to model users' intents in the latent space [5]. DSSRec [26] leverages a seq2seq training strategy to extract supervision signals from multiple future interactions and introduces an intent variable to capture mutual information between a user's historical and future behavior sequences. SINE [35] proposes a sparse interest extraction module to infer the interacted intentions of a user from a large pool of intention groups. ICLRec [5] extracts users' intent distributions from all user behavior sequences via clustering and integrates the learned intent into the SR model via a contrastive SSL loss. IOCRec [21] improves ICLRec by introducing two new modules, global and local modules, for modeling intentions, and it leverages these two modules to alleviate the noise problem of the contrastive learning task. MITGNN [24] proposes a multi-intent translation graph neural network to mine users' multiple intents by considering the correlations of the intents. However, these methods all ignore the intent supervisory signals hidden in the user interaction sequence [50]. In addition, introducing stochastic data augmentation into intent modeling may change the original intent hidden in the interaction sequence [10]. Instead, our method extracts supervisory signals from users' interactions and leverages Contrastive Learning (CL) to learn users' intentions.

5 CONCLUSION

In this paper, we propose a novel contrastive learning based sequential recommendation system called ICSRec. ICSRec extracts coarse-grain intent supervisory signals from all users' historical interaction sequences and then utilizes these intent supervisory signals to construct two auxiliary learning objectives for intent representation learning. This approach helps alleviate the sparsity problem of interaction data and presents more suitable items for users with different intentions. Finally, the outstanding performance of our model on four real-world datasets demonstrates our mentioned model's superiority over other methods. Extensive experiments show that IOCRec achieves state-of-the-art performance against a series of SOTA solutions. For future work, we would like to develop novel auxiliary learning objects for improving the performance of ICSRec. Moreover, we are also interested in applying ICSRec to improve the performance of other sequential recommendation models.

6 ACKNOWLEDGMENTS

This research was partially supported by the NSFC (62376180, 62176175), the major project of natural science research in Universities of Jiangsu Province (21KJA520004), Suzhou Science and

Technology Development Program(SYG202328) and the Priority Academic Program Development of Jiangsu Higher Education Institutions.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *TKDE* 17, 6 (2005), 734–749.

[2] Renqin Cai, Jibang Wu, Aidan San, Chong Wang, and Hongning Wang. 2021. Category-aware Collaborative Sequential Recommendation. In *SIGIR*. 388–397.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*. 1597–1607.

[4] Wanyu Chen, Pengjie Ren, Fei Cai, Fei Sun, and Maarten de Rijke. 2020. Improving End-to-End Sequential Recommendations with Intent-aware Diversification. In *CIKM*. 175–184.

[5] Yongjun Chen, Zhiwei Liu, Jia Li, Julian J. McAuley, and Caiming Xiong. 2022. Intent Contrastive Learning for Sequential Recommendation. In *WWW*. 2172–2182.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*.

[8] Ziwei Fan, Zhiwei Liu, Yu Wang, Alice Wang, Zahra Nazari, Lei Zheng, Hao Peng, and Philip S. Yu. 2022. Sequential Recommendation via Stochastic Self-Attention. In *WWW*. 2036–2047.

[9] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. 2020. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *TOIS* 39, 1 (2020), 1–42.

[10] Wei Guo, Can Zhang, Zhicheng He, Jiarui Qin, Huifeng Guo, Bo Chen, Ruiming Tang, Xiuqiang He, and Rui Zhang. 2022. MISS: Multi-Interest Self-Supervised Learning Framework for Click-Through Rate Prediction. In *ICDE*. 727–740.

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*. 9726–9735.

[12] Ruining He and Julian J. McAuley. 2016. Fusing Similarity Models with Markov Chains for Sparse Sequential Recommendation. In *ICDM*. 191–200.

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR*.

[14] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-Scale Similarity Search with GPUs. *Big Data* 7, 3 (2021), 535–547.

[15] Wang-Cheng Kang and Julian J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *ICDM*. 197–206.

[16] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

[17] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD*. 1748–1757.

[18] Chao Li, Zhiyuan Liu, Mengmeng Wu, Yuchi Xu, Huan Zhao, Pipei Huang, Guoliang Kang, Qiwei Chen, Wei Li, and Dik Lun Lee. 2019. Multi-Interest Network with Dynamic Routing for Recommendation at Tmall. In *CIKM*. 2615–2623.

[19] Haoyang Li, Xin Wang, Ziwei Zhang, Jianxin Ma, Peng Cui, and Wenwu Zhu. 2022. Intention-Aware Sequential Recommendation With Structured Intent Transition. *TKDE* 34, 11 (2022), 5403–5414.

[20] Jiacheng Li, Tong Zhao, Jin Li, Jim Chan, Christos Faloutsos, George Karypis, Soo-Min Pantel, and Julian J. McAuley. 2022. Coarse-to-Fine Sparse Sequential Recommendation. In *SIGIR*. 2082–2086.

[21] Xuewei Li, Aitong Sun, Mankun Zhao, Jian Yu, Kun Zhu, Di Jin, Mei Yu, and Ruiguo Yu. 2023. Multi-Intention Oriented Contrastive Learning for Sequential Recommendation. In *WSDM*. 411–419.

[22] Yang Li, Tong Chen, Peng-Fei Zhang, and Hongzhi Yin. 2021. Lightweight Self-Attentive Sequential Recommendation. In *CIKM*. 967–977.

[23] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479* (2021).

[24] Zhiwei Liu, Xiaohan Li, Ziwei Fan, Stephen Guo, Kannan Achan, and Philip S. Yu. 2020. Basket Recommendation with Multi-Intent Translation Graph Neural Network. In *BigData*. 728–737.

[25] Anjing Luo, Pengpeng Zhao, Yanchi Liu, Fuzhen Zhuang, Deqing Wang, Jiajie Xu, Junhua Fang, and Victor S. Sheng. 2020. Collaborative Self-Attention Network for Session-based Recommendation. In *IJCAI*. 2591–2597.

[26] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled Self-Supervision in Sequential Recommenders. In *KDD*. ACM, 483–491.

[27] Xiuyuan Qin, Huanhuan Yuan, Pengpeng Zhao, Junhua Fang, Fuzhen Zhuang, Guanfeng Liu, Yanchi Liu, and Victor S. Sheng. 2023. Meta-optimized Contrastive Learning for Sequential Recommendation. In *SIGIR*. 89–98.

[28] Ruihong Qiu, Zi Huang, and Hongzhi Yin. 2021. Memory Augmented Multi-Instance Contrastive Predictive Coding for Sequential Recommendation. In *ICDM*. 519–528.

[29] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive learning for representation degeneration problem in sequential recommendation. In *WSDM*. 813–823.

[30] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *UAI*. 452–461.

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

[32] Deepjyoti Roy and Mala Dutta. 2022. A systematic review and research perspective on recommender systems. *Journal of Big Data* 9, 1 (2022), 59.

[33] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *JMLR* 15, 1 (2014), 1929–1958.

[34] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *CIKM*. 1441–1450.

[35] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. Sparse-Interest Network for Sequential Recommendation. In *WSDM*. 598–606.

[36] Yong Kiam Tan, Xinxing Xu, and Yong Liu. 2016. Improved recurrent neural networks for session-based recommendations. In *DLRS@RecSys*. 17–22.

[37] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *WSDM*. 565–573.

[38] Md. Mehrab Tanjim, Congzhe Su, Ethan Benjamin, Diane Hu, Liangjie Hong, and Julian J. McAuley. 2020. Attentive Sequential Models of Latent Intent for Next Item Recommendation. In *WWW*. 2528–2534.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NeurIPS*. 5998–6008.

[40] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet A. Orgun. 2019. Sequential Recommender Systems: Challenges, Progress and Prospects. In *IJCAI*. 6332–6338.

[41] Shoujin Wang, Liang Hu, Yan Wang, Quan Z Sheng, Mehmet Orgun, and Longbing Cao. 2019. Modeling multi-purpose sessions for next-item recommendations via mixture-channel purpose routing networks. In *IJCAI*. 3771–3777.

[42] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*. 165–174.

[43] David Warde-Farley, Ian J. Goodfellow, Aaron C. Courville, and Yoshua Bengio. 2014. An empirical analysis of dropout in piecewise linear networks. In *ICLR*.

[44] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *WSDM*. 495–503.

[45] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised Graph Learning for Recommendation. In *SIGIR*. 726–735.

[46] Xin Xia, Hongzhi Yin, Junliang Yu, Qinyong Wang, Lizhen Cui, and Xiangliang Zhang. 2021. Self-Supervised Hypergraph Convolutional Networks for Session-based Recommendation. In *AAAI*. 4503–4511.

[47] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. 2022. Contrastive Learning for Sequential Recommendation. In *ICDE*. 1259–1273.

[48] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Jundong Li, and Zi Huang. 2023. Self-supervised learning for recommender systems: A survey. *TKDE* (2023).

[49] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *CSUR* 52, 1 (2019), 1–38.

[50] Yixin Zhang, Yong Liu, Yonghui Xu, Hao Xiong, Chenyi Lei, Wei He, Lizhen Cui, and Chunyan Miao. 2022. Enhancing Sequential Recommendation with Graph Contrastive Learning. In *IJCAI*. 2398–2405.

[51] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*. 1893–1902.

[52] Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph Contrastive Learning with Adaptive Augmentation. In *WWW*. 2069–2080.

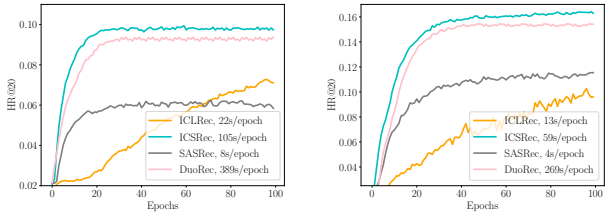

Figure 6: Training efficiency on Sports and Toys datasets.

7 APPENDIX

7.1 Case Study

In this subsection, we will briefly analyze cases that occur in real-world data. Specifically, we randomly selected two pairs of users in ML-1M datasets. As shown in Table 5, User1 and User2 may have the same intention with the same target item (e.g., watch a drama movie.). User3 and User4 have the same target item, but their intentions may be different (e.g., User3 is more inclined to watch the 'Action' movie, and User4 is more inclined to watch the 'Children' movie.). To analyze the effectiveness of our model, we put four sequences into the trained model and put the representation of the four users' output from the model into the visualization. The intent heat maps of four users are shown in Figure 5. To contrast the heat maps, we use the inner product to calculate the similarity among different intents from the same user. Specifically, we use $\mathbf{H}^u \cdot (\mathbf{H}^u)^T$ to represent the intent relation among different steps. Comparing User1, User2, User3, and User4, we can find that all four users have different intention distributions, which indicates that each user has their own preferences. In addition, it can be seen that User1, User3, and User4 all have relatively stable preferences, so their intention representations are more dispersed (i.e., most intentions are related.). User2 has more diverse intentions, so his intention representations are more concentrated (i.e., most intentions are unrelated). The above findings demonstrate the effectiveness of our model in modeling user intention.

7.2 Complexity Analysis

In this subsection, we conduct a detailed complexity analysis of ICSRec. We choose SASRec [15] as our base encoder. Since our ICSRec framework does not introduce any auxiliary learnable parameters. The model size of ICSRec is identical to SASRec. The learned parameters in ICSRec are from embedding and parameters in the self-attention layers, feed-forward networks, and layer normalization. The total number of parameters is $O(|\mathcal{I}|d + nd + d^2)$.

Then, we analyze the time complexity of the training procedure of ICSRec framework. In our supervisory signal construction module, we use the dynamic slide window operation to split the original sequence into a set of sub-sequences. We assume the length of each user's sequence is L_a for simplicity. Let $|B|$ denote the size of the mini-batch, and d denote the embedding size. The additional operations mainly come from three components: applying data split operation on original user sequences, deriving intent representations from their interactions in the E-step, and multi-task learning in the M-step. First, we operate the data by dynamic slide window in the offline. The total complexity is $O(L_a \times (L_a - 1))|U|$. Second, for the E-step, the time complicity is $O(NmKd)$ from clustering, where N represents the number of the processed data after sliding operation, m is the maximum iteration number in clustering($m = 20$ in this paper), K is the cluster number and d is the dimensionality of the embedding. Third, for the M-step, since we have three objectives to optimize the encoder $f_\theta(\cdot)$, the time complexity is $3 \times O(N^2d + Nd^2)$. The overall complexity is dominated by the term $O(3 \times (N^2d))$, which is 3 times of Transformer based SR with only the next item prediction objective, e.g., SASRec. Fortunately, the model can be benefactive parallized because f_θ is Transformer. In the testing phase, the proposed CICL and FICL objectives are no longer needed, which yields the model to have the same time complexity as SASRec($O(d|\mathcal{I}|)$). The empirical time spending comparisons are reported in the experiments.

Algorithm 1 The ICSRec Algorithm

Input: training dataset $\{S^u\}_{u=1}^{|U|}$, sequence encoder $f_\theta(\cdot)$, hyper-parameters K, β, λ, max training epochs E, batch size $|B|$.
Output: $f_\theta(\cdot)$.
Split $\{S^u\}_{u=1}^{|U|}$ into several subsequences via Eq. (2).
while $epoch \leq E$ **do**
  // Update intent prototype representation C
  C = Clustering$(f_\theta(\{\{S_t^u\}_{t=1}^{|S^u|}\}_{u=1}^{|U|}), K)$
  for a minibatch$\{s_u\}_{u=1}^{|B|}$ **do**
    for $u \in \{1, 2, 3, ..., |B|\}$ **do**
      // Get sub-sequence s_2 with the same target item to s_1.
      $s_2 = Sample(T_{s_1})$.
      // Encoding via $f_\theta(\cdot)$.
      $\mathbf{h}^u = f_\theta(S_1)$
      $\mathbf{h}_1 = f_\theta(s_1)$,$\mathbf{h}_2 = f_\theta(s_2)$
      // Query the intent prototype representations
      $\mathbf{c}_1 = query(\mathbf{h}_1)$,$\mathbf{c}_2 = query(\mathbf{h}_2)$
      Calculate $\mathcal{L}_{cicl}$ by Eq. (5)
      Calculate $\mathcal{L}_{ficl}$ by Eq. (8)
    end for
    // Optimization the encoder via Eq. (11)
    $\mathcal{L} = \mathcal{L}_{Rec} + \lambda \cdot \mathcal{L}_{cicl} + \beta \cdot \mathcal{L}_{ficl}$
    Update network $f_\theta(\cdot)$ to minimize $\mathcal{L}$
  end for
end while

7.3 Training Efficiency

Figure 6 demonstrates the efficiency of four compared methods with GPU acceleration. On the one hand, SASRec [15] exhibits the fastest computational speed. On the other hand, SASRec [15] tends to have a less favorable performance compared to other models. This indicates that incorporating CL auxiliary tasks not only increases the computational overhead of the model but also enables the model to learn more features, thereby enhancing its recommendation performance. Upon comparing ICSRec with other models, it becomes evident that the computational overhead of ICSRec is within a reasonable and acceptable range, and its performance improvement is significant, affirming the superiority of ICSRec.