Sparse Enhanced Network: An Adversarial Generation Method for Robust Augmentation in Sequential Recommendation

Junyang Chen[*1], **Guoxuan Zou**[*1], **Pan Zhou**[2], **Wu Yirui**[3], **Zhenghan Chen**[4], **Houcheng Su**[5], **Huan Wang**[†6], **Zhiguo Gong**[5]

[1]College of Computer Science and Software Engineering, Shenzhen University, China. [2]School of Cyber Science and Engineering, Huazhong University of Science of Technology, China. [3]College of Computer and Information, Hohai University, China. [4]Microsoft (China) Co., Ltd, China. [5]Department of Computer Information Science, University of Macau, China. [6]College of Informatics, Huazhong Agricultural University, China.
junyangchen@szu.edu.cn

Abstract

Sequential Recommendation plays a significant role in daily recommendation systems, such as e-commerce platforms like Amazon and Taobao. However, even with the advent of large models, these platforms often face sparse issues in the historical browsing records of individual users due to new users joining or the introduction of new products. As a result, existing sequence recommendation algorithms may not perform well. To address this, sequence-based data augmentation methods have garnered attention.

Existing sequence enhancement methods typically rely on augmenting existing data, employing techniques like cropping, masking prediction, random reordering, and random replacement of the original sequence. While these methods have shown improvements, they often overlook the exploration of the deep embedding space of the sequence. To tackle these challenges, we propose a Sparse Enhanced Network (SparseEnNet), which is a robust adversarial generation method. SparseEnNet aims to fully explore the hidden space in sequence recommendation, generating more robust enhanced items. Additionally, we adopt an adversarial generation method, allowing the model to differentiate between data augmentation categories and achieve better prediction performance for the next item in the sequence. Experiments have demonstrated that our method achieves a remarkable 4-14% improvement over existing methods when evaluated on the real-world datasets. (https://github.com/junyachen/SparseEnNet)

Introduction

The aim of sequential recommendation (SR) is to comprehend users' evolving preferences based on their historical behaviors, facilitating precise predictions of their forthcoming item preferences (Chen et al. 2022; Liu et al. 2021; Xie et al. 2022; Hjelm et al. 2018). SR has garnered significant interest for its effectiveness in predicting user interests.

For example, FPMC (Rendle, Freudenthaler, and Schmidt-Thieme 2010) synergizes matrix factorization and Markov chain methodologies to enhance recommender systems by deriving personalized transition matrices for users from limited observations, outperforming other models in sequential basket data. Additionally, ICL (Chen et al. 2022) develops users' intent distribution functions from unlabeled behavior sequences and optimizes SR models with contrastive self-supervised learning by incorporating the learned intents. Nonetheless, these methods struggle to effectively address next-item prediction tasks in scenarios where user historical sequences are short and sparse.

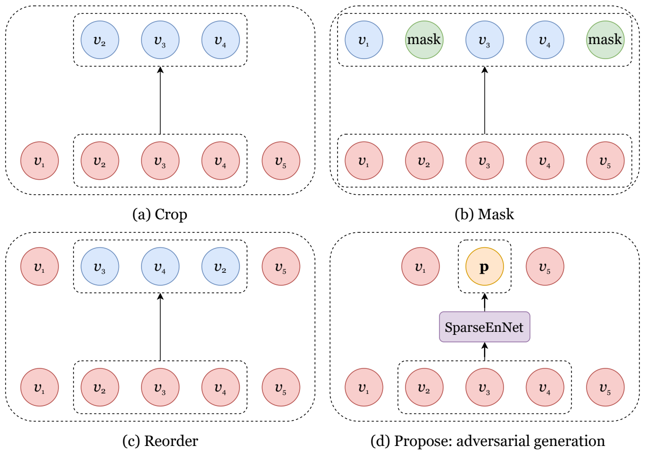


Figure 1: Examples of data augmentation techniques.

To tackle this challenge, recent studies have introduced innovative approaches. CoSeRec (Liu et al. 2021) explores the use of contrastive self-supervised learning, employing informative augmentation operators to create refined views, effectively addressing concerns like data sparsity and noisy data. In addition, CL4SRec (Xie et al. 2022) introduces three data augmentation techniques (crop/mask/reorder) to project user interaction sequences into diverse perspectives, enhancing the learning of superior user representations. Nevertheless, previous methodologies mainly focus on enhancing the surface layers of the original sequence, overlooking the interconnectedness within the hidden space of the sequence.

To illustrate this issue, we provide an example in Fig. 1,

[*]These authors contributed equally.
[†]Corresponding author

where each top row in the subgraphs represents an additional sequence generated using data augmentation methods, while the corresponding original sequence is shown in the bottom row. The figure demonstrates the following scenarios: (a) Random selection of a continuous sub-sequence from the original sequence starting at position v_2; (b) Random masking of several items within a sequence; (c) Random shuffling of a sub-sequence. Furthermore, there are alternative approaches, like substituting or inserting various items into a sequence. Collectively, we term the aforementioned approaches as shallow data augmentation.

To further comprehensively explore the latent space in sequence recommendation, we integrate previously mentioned data augmentation techniques and introduce a novel adversarial generation approach named SparseEnNet, as depicted in Fig. 1 (d). In this method, we process items v_3, v_4, and v_5 through a specially designed encoder to produce virtual item embeddings, thus enhancing the original sequence. Nevertheless, we observed that an excessive use of data augmentation methods can potentially hamper the predictive accuracy of the original model, especially for longer sequences which inherently contain rich historical information. This excessive augmentation could lead to suboptimal results.

Consequently, in order to enhance the robustness of the sequential recommendation performance, we design four main components in the proposed SparseEnNet. More concretely, we first exploit an augmentation discriminator to differentiate between data augmentation categories and achieve better prediction performance for the next item in the sequence. Then, we adopt a stability discriminator to stabilize the generated item embeddings from the same augmentation operation. After that, we employ a negative sample learning approach to maximize the mutual information between two positive pairs while effectively increasing the distance from negative items. Without loss of generality, we utilize a Transformer (Vaswani et al. 2017) as the sequence encoder to predict the next item. Finally, we use the self-training enhanced learning to encode all original sequences using a sequence encoder and then aggregate the sequence representations to form a set of sequence representations, for capturing consistency among similar sequences. Note that all mentioned encoders are served as the parts of the generator. The contributions of our work can be summarized as follows:

(1) We propose a robust adversarial generation method, called SparseEnNet, that can fully explore the hidden space in sequence recommendation by generating more robust enhanced items.

(2) We design four main components: an augmentation discriminator, a stability discriminator, a negative sample learning module, and a self-training enhanced learning module to achieve the above purpose. Extensive experiments are conducted on three widely used datasets to demonstrate the superiority of our method over several baselines.

Related Work

Contrastive Learning in SR

Early research in sequential recommendation (SR) often relied on Markov Chains (MC) to capture the sequential correlations among items (Zimdars, Chickering, and Meek 2001). As deep learning advanced, various deep learning architectures were incorporated into sequence recommendation tasks. For instance, (Hidasi et al. 2015) integrated the Gated Recurrent Unit (GRU) into sequence recommendation. Moreover, (Kang and McAuley 2018; Ji et al. 2020; Li, Wang, and McAuley 2020) harnessed attention mechanisms to extract contextual information, leading to more promising outcomes. Following that, contrastive learning (CL), a self-supervised task, has garnered significant attention across various domains, including computer vision (CV) (Chen et al. 2020; He et al. 2020) and natural language processing (NLP) (Fang et al. 2020). More recently, contrastive learning has also found utility in the recommendation field, enhancing performance in sequential recommendation models. For instance, (Yao et al. 2021) proposed a collaborative filtering-based recommendation method that employs DNN-based contrastive learning to enhance item features. Furthermore, several studies have showcased the potential of using contrastive learning in graph neural networks to enhance recommendation performance (Wu et al. 2021; Xia et al. 2021). In the realm of sequential recommendation, S^3-Rec (Zhou et al. 2020) employed contrastive learning for pre-training to improve item representation. Conversely, CL4SRec (Xie et al. 2022) incorporated CL into a multi-task learning framework, synergizing contrastive learning with sequential recommendation tasks for performance improvement. Besides, ICLRec (Chen et al. 2022) introduced intent contrastive self-supervised learning to capture user intents.

The sequential recommendation methods mentioned above typically depend on extensive historical user-item interactions (Chen et al. 2023). Although some earlier approaches (Xie et al. 2022; Chen et al. 2022) have integrated data augmentation techniques, they often lack an in-depth exploration of the underlying embedding space of sequences. As such, they may not effectively generate more resilient item embeddings.

Adversarial Learning in SR

Adversarial learning, originally introduced by Generative Adversarial Nets (GAN) (Goodfellow et al. 2014), enhances model performance by engaging in a minimax game between the generator and the discriminator. This approach has found wide applications in domains such as domain adaptation (Ganin et al. 2016) and anomaly detection (Akcay, Atapour-Abarghouei, and Breckon 2019). In the realm of recommendation, adversarial methods have been employed to enhance the performance of base models. For instance, Adversarial Personalized Ranking (APR) (He et al. 2018) employs adversarial training to bolster the robustness of Bayesian Personalized Ranking (BPR). In sequence recommendation, MFGAN (Ren et al. 2020) employs multiple factor discriminators to assess recommendations generated by the encoder, effectively disentangling various recommendation factors. In comparison to previous models, our approach incorporates adversarial generation into data augmentation in sequence recommendation. This integration enables the model to distinguish between data augmentation categories, ultimately leading to improved prediction performance for subsequent items in the sequence.

Proposed Model

Problem Definition

We define sequential recommendation as follows: Given a user-item interaction sequence $s^u = [v_1^u, ..., v_t^u, ..., v_{|s^u|}^u]$, where $u \in \mathcal{U}$ represents the user set, and $v \in \mathcal{V}$ represents the item set. For a user u, the sequential recommendation task involves predicting the most likely item interaction at next step $|s^u| + 1$. This can be formulated as follows:

$$\arg\max_{v \in \mathcal{V}} P(v_{|s^u|+1}^u = v | s^u). \tag{1}$$

Formulated Data Augmentation

Shallow data augmentation: Without sacrificing generality, we adopt three data augmentation methods for sequence augmentation, following the approach outlined in previous work (Xie et al. 2022). The augmentation can be formulated:

Crop (C): A random subsequence of length L_C is selected from the original sequence s^u. The subsequence is cropped from position c, and this process can be defined by:

$$s_u^C = [v_c^u, v_{c+1}^u, ..., v_{c+L_C-1}^u]. \tag{2}$$

Mask (M): Items from the original sequence s^u are randomly chosen for masking. The mask operation can be described as follows:

$$s_u^M = [\hat{v}_1^u, \hat{v}_2^u, ..., \hat{v}_{|s^u|}^u], \tag{3}$$

where $\hat{v}_t^u$ denotes the randomly masked item.

Reorder (R): A continuous subsequence of a given length L_R starting from r is randomly shuffled within the original sequence s^u. The resulting reordered sequence can be formulated as:

$$s_u^R = [v_1^u, ..., \hat{v}_r^u, ..., \hat{v}_{r+L_R-1}^u, ..., v_{|s^u|}^u]. \tag{4}$$

Deep data augmentation: We have observed that excessive utilization of data augmentation methods might have a negative impact on the predictive accuracy of the original model, particularly for longer sequences that inherently hold substantial historical information. This overzealous augmentation could potentially lead to suboptimal results. To address this, and in order to bolster the robustness of sequential recommendation performance, we introduce the following approaches aimed at thoroughly exploring the latent space in sequence recommendation, thereby generating more robust enhanced items:

Pooling (P): We user a pooling operator to explore the hidden information of items. The operation is described as follows:

$$s_u^P = [v_1^u, ..., v_{n-1}^u, \mathbf{p}, v_{n+L_P}^u, ..., v_{|s^u|}^u], \tag{5}$$

where $\mathbf{p} = \mathrm{mean}(v_n^u, ..., v_{n+L_P-1}^u)$ is the generated item, n is the starting position of the pool subsequence, and L_P is the number of items used for pooling.

Sparse Enhanced Network

The complete structure of SparseEnNet is depicted in Fig. 2, where specifics of each component are elucidated as follows.

Although the aforementioned data augmentation methods can alleviate the data sparsity issue in short sequences, the generated items might adversely affect the original sequences if there is a substantial disparity in the distribution between the generated items and the original ones. To reduce the impact of distribution differences on model performance, we design an adversarial generation method, allowing the model to differentiate between data augmentation categories and achieve better prediction performance for the next item in the sequence. The details are as follows.

Augmentation Discriminator: To solve the above problem, we exploit an augmentation discriminator with a minimax game. To augment the encoder's ability to capture latent invariance across different augmentations, the augmentation discriminator's purpose is to differentiate the source augmentation method of the sequence representation. In this context, a single augmented sequence representation is input into the discriminator to determine the employed augmentation operation from the augmentation set. Denoted as a non-linear function, the augmentation discriminator is represented by $f_{AD}(\cdot)$. The loss function for this module employs cross entropy and is defined as follows:

$$\mathcal{L}_{AD} = -\sum\nolimits_{u \in \mathcal{U}} \sum\nolimits_{a_j \in \mathcal{A}} j \log(f_{AD}(\mathbf{h}_{a_j}^u)), \tag{6}$$

where a_j is the label in the augmentation operation set $\mathcal{A}$ (as mentioned in "Formulated Data Augmentation" Section), and $\mathbf{h}_{a_j}^u$ represents the sequence representation derived from the generator $f_E(\cdot)$[1]. The parameter θ_{AD} in the discriminator is optimized by:

$$\hat{\theta}_{AD} = \arg\min_{\theta_{AD}} \mathcal{L}_{AD}(\theta_E, \theta_{AD}), \tag{7}$$

where θ_E denotes the parameters in the encoder. Correspondingly, we optimize it as follows:

$$\hat{\theta}_E = \arg\max_{\theta_E} \mathcal{L}_{AD}(\theta_E, \theta_{AD}). \tag{8}$$

The classification loss of Eq. (6) on the augmented operations indirectly assesses the encoder's capacity to capture latent invariance across various augmentations. A larger value of the loss $\mathcal{L}_{AD}$ indicates improved ability of the encoder to extract invariance among different augmentation methods. Conversely, a smaller loss signifies effective discrimination by the discriminator across data augmentation categories, resulting in enhanced predictive performance for the subsequent item in the sequence.

Stability Discriminator: In this part, we aim to stabilize the generated item embeddings from the same augmentation operation. We design the stability discriminator as a nonlinear function $f_{SD}(\cdot)$ and define the classification loss of the discriminator by cross entropy as follows:

$$\mathcal{L}_{SD} = -\sum\nolimits_{u \in \mathcal{U}} \sum\nolimits_{a_j, a_j' \in \mathcal{A}} \log(f_{SD}(\mathbf{h}_{a_j}^u \,||\, \mathbf{h}_{a_j'}^u)), \tag{9}$$

[1]In this paper, all mentioned encoders are served as the parts of the generator.

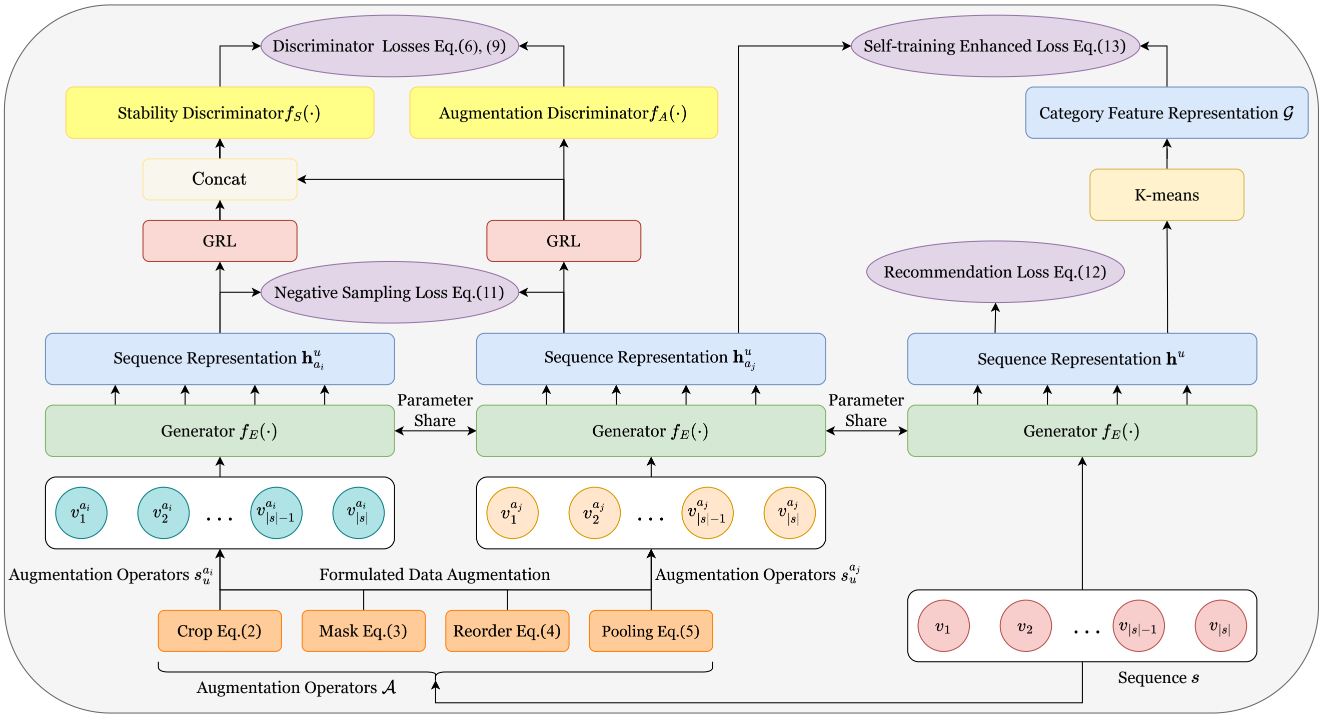

Discriminator Losses Eq.(6), (9)
Self-training Enhanced Loss Eq.(13)
Stability Discriminator $f_S(\cdot)$
Augmentation Discriminator $f_A(\cdot)$
Category Feature Representation $\mathcal{G}$
Concat
K-means
GRL
GRL
Recommendation Loss Eq.(12)
Negative Sampling Loss Eq.(11)
Sequence Representation $\mathbf{h}^u_{a_i}$
Sequence Representation $\mathbf{h}^u_{a_j}$
Sequence Representation $\mathbf{h}^u$
Parameter Share
Parameter Share
Generator $f_E(\cdot)$
Generator $f_E(\cdot)$
Generator $f_E(\cdot)$
Augmentation Operators $s^{a_i}_u$
Formulated Data Augmentation
Augmentation Operators $s^{a_j}_u$
Crop Eq.(2)
Mask Eq.(3)
Reorder Eq.(4)
Pooling Eq.(5)
Augmentation Operators $\mathcal{A}$
Sequence s

Figure 2: Overall structure of SparseEnNet. When examining the structure from bottom to top, two augmentation operators, denoted as a_i and a_j, are randomly chosen from the augmentation set $\mathcal{A} = \{C, M, R, P\}$. Then, the Encoder processes both the augmented sequences and the original sequence, generating sequence representations used for calculating the losses.

where $a_j^{'}$ denotes the same data augmentation type as a_j, and $||$ denotes the concatenation. We repeat the same enhancement operation twice on the identical sequence, concurrently reducing the discriminator loss during the loss function optimization process. This strategy ensures stable learning for the encoder. Similarly, we update the parameter θ_{SD} in the discriminator as follows:

$$\hat{\theta}_{SD} = \arg\min_{\theta_{SD}} \mathcal{L}_{SD}(\theta_E, \theta_{SD}). \tag{10}$$

Negative Sample Learning: Lately, self-supervised contrastive methods (Xie et al. 2022; Liu et al. 2021) have garnered attention in the recommendation field due to their remarkable success in enhancing negative sample learning during batch training. We employ this negative sample learning approach to maximize the mutual information between two positive pairs while effectively increasing the distance from negative items. The InfoNCE loss is utilized as the loss function, as shown below:

$$\mathcal{L}_{NSL} = \sum\nolimits_{u \in \mathcal{U}} \sum\nolimits_{a_i, a_j \in \mathcal{A}} -\log \frac{\exp(\text{sim}(\mathbf{h}^u_{a_i}, \mathbf{h}^u_{a_j})/\tau)}{\sum_{\bar{u} \in neg} \exp(\text{sim}(\mathbf{h}^u_{a_i}, \mathbf{h}^{\bar{u}}_{a_j})/\tau)}, \tag{11}$$

where $\text{sim}(\cdot)$ represents the dot product, τ is the scale factor, and $\mathbf{h}^u_{a_j}$ and $\mathbf{h}^u_{a_k}$ denote the sequence representations of $s^u_{a_i}$ and $s^u_{a_i}$, which have been independently augmented by two distinct augmentation operators a_i and a_j. The pair formed by $\mathbf{h}^u_{a_j}$ and $\mathbf{h}^u_{a_k}$ is considered as a positive pair, while $s^{\bar{u}}_{a_j}$ represents the augmented sequence that does not belong to user u but resides within the same training batch. In this context, it is treated as a negative pair.

Next Item Prediction: Without loss of generality, we utilize a Transformer (Vaswani et al. 2017) as the sequence encoder to extract sequential information for predicting the next item. We employ a log-likelihood loss function to optimize the prediction at time step t:

$$\begin{aligned}\mathcal{L}_{NIP}(s^u, t) = &- \log(\sigma(\mathbf{h}^u_t \cdot \mathbf{e}_{v_{t+1}})) \\ &- \sum_{v_j \notin s^u} \log(1 - \sigma(\mathbf{h}^u_t \cdot \mathbf{e}_{v_j})),\end{aligned} \tag{12}$$

where $\mathbf{h}^u_t$ represents the output of the Transformer encoder at position t, $e_{v_{t+1}}$ stands for the actual next item at time step t, σ denotes the Sigmoid function, and v_j corresponds to a randomly selected negative item not present in sequence s^u and drawn from the batch (Chen et al. 2022).

Self-training Enhanced Learning: As mentioned in the "Augmentation Discriminator" Section, we consider two different augmentations of the same sequence as positives and treats augmentations of different sequences as negatives, aiming to bring positives closer together and push negatives farther apart. However, even if two different sequences are somewhat similar, the discriminator might still treat them as negatives and attempt to separate them, which could hinder the feature encoder's ability to capture consistency among similar sequences. Inspired by (Chen et al. 2022), we encode all original sequences using a sequence encoder and then aggregate the sequence representations to form a set of sequence representations. Then, appling k-means clustering to obtain pseudo clusters of embedding representations for different categories. Clustering allows us to group feature-similar sequences into the same category, which we call it as the self-training enhanced learning. We utilize the mean of each category as the category feature representation and apply the sequence within this category to the following loss:

$$\mathcal{L}_{SEL} = \sum\nolimits_{\mathbf{g}\in\mathcal{G}} \sum\nolimits_{u\in\mathcal{U}_g} -\log\frac{\exp(\text{sim}(\mathbf{h}_a^u,\mathbf{g})/\tau)}{\sum_{\bar{\mathbf{g}}\in\mathcal{G}}\exp(\text{sim}(\mathbf{h}_{a_j}^u,\bar{\mathbf{g}})/\tau)}, \quad (13)$$

where $\mathcal{U}_g$ is the set of users belonging to the cluster g, $\mathbf{g}$ is the cluster feature representation.

Unified Training Loss

In essence, our goal is to minimize the recommendation loss for the next item prediction as indicated in Eq. (12). Additionally, we aim to employ negative sample learning to bolster the recommendation task and employ self-training for enhanced learning to mitigate the issue of false negatives, as expressed through Eq. (11) and Eq. (13). Furthermore, the integration of an adversarial mechanism is deemed necessary to enhance the semantic consistency between augmented sequences, and this is realized through Eq. (9) and Eq. (6). Thus, the joint loss function for SparseEnNet is defined as follows:

$$\mathcal{L} = \mathcal{L}_{NIP} + \lambda_1\mathcal{L}_{NSL} + \lambda_2\mathcal{L}_{SEL} + \lambda_3(\mathcal{L}_{SD} + \mathcal{L}_{AD}) \quad (14)$$

where λ_1, λ_2, and λ_3 serve as balancing weights to regulate the emphasis on multi-tasks.

GRL for Adversarial Training: To facilitate the end-to-end training of adversarial methods, we incorporate a gradient reversal layer (GRL) (Ganin et al. 2016) between the encoder and the discriminators. GRL reverses the gradient during the backpropagation process, causing the parameters before the GRL to be optimized to increase the loss, while the gradient direction of the parameters after the GRL remains unchanged to optimize for decreasing the loss. The optimization objectives of the parameters before and after the GRL are contrary, thus achieving the goal of adversarial learning. The parameter update process is as follows:

$$\theta = \theta - \mu\frac{\partial\mathcal{L}}{\partial\theta}, \quad (15)$$

where $\theta \in \theta_E, \theta_I, \theta_T$ represents parameters within the encoder and discriminators, and μ signifies the learning rate.

Experiment

In this section, we conduct comprehensive experiments on SparseEnNet to address the following inquiry:

RQ1: Does the proposed method outperform the baseline models? **RQ2:** How do various augmentation methods impact the model's performance? **RQ3:** How do different components influence the performance of SparseEnNet? **RQ4:** Does SparseEnNet mitigate the cold-start problem caused by data sparsity? **RQ5:** How do various hyper-parameters influence the performance of our model? (We have included these results in the supplementary material.)

Table 1: Dataset statistics

Dataset	#Users	#Items	#Actions	Avg.length	Sparsity
Beauty	22,363	12,101	198,502	8.9	99.93%
Toys	19,412	11,924	167,597	8.6	99.93%
Yelp	30,431	20,033	316,354	10.4	99.95%

Datasets: We perform experiments on three publicly available datasets sourced from real-world data. The Beauty and Toys subsets are extracted from the Amazon reviews dataset (McAuley et al. 2015), derived from one of the world's largest e-commerce platforms. Additionally, we utilize the Yelp dataset[2], which is a frequently employed resource in recommendation tasks and originates from a business platform. We adhere to the established convention (Kang and McAuley 2018; Xie et al. 2022) for dataset processing. Then, we organize each user's reviews in chronological sequence and transform them into a sequence of user-item interactions suitable for the recommendation models.

Evaluation Metrics: We employ the leave-one-out strategy (Kang and McAuley 2018; Zhou et al. 2020; Xie et al. 2022), a widely employed approach in sequence recommendation, to partition the datasets. In this strategy, for each user-item interaction sequence, we treat the last item as the test data, and the item immediately preceding it as the validation data. The remaining items are utilized for training the model. To comprehensively assess all models, we employ the Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) as evaluation metrics for recommendation performance. HR@top-k quantifies the proportion of the target item in the top-k recommended items. In comparison, NDCG@top-k takes into consideration the rank of the target item among the top-k recommendations.

Baseline Methods: The selected baseline methods encompass the following: **GRU4Rec** (Hidasi et al. 2015) integrates the GRU architecture into session-based recommendation tasks, augmenting model performance through novel loss function design and efficient sampling strategies. **SASRec** (Kang and McAuley 2018) leverages a Transformer-based one-way attention mechanism to capture sequential patterns and make next-item recommendations. **BERT4Rec** (Sun et al. 2019) adapts the BERT technique (Devlin et al. 2018), originally successful in NLP, to sequence recommendation. It employs a two-way self-attention mechanism and mask operation for enhanced performance. **CL4SRec** (Xie et al. 2022) employs contrastive learning within sequence recommendation, introducing crop, mask, and reorder augmentation techniques for the contrastive learning framework. **ICLRec** (Chen et al. 2022) engages unsupervised modeling of user intentions and incorporates these intentions into contrastive sequential recommendation tasks.

Implementation Details: We evaluate the above baselines by either using the RecStudio[3] (a widely used gen-

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

[3]https://github.com/ustcml/RecStudio

Table 2: Performance comparison of various methods on top-N recommendation. The best score for each metric is indicated in bold, and the second-best score is underlined. The final row displays improvements over the best baseline on each dataset.

Datasets	Methods	Hit@5	NDCG@5	Hit@10	NDCG@10	Hit@20	NDCG@20
Beauty	GRU4Rec	0.0256	0.0164	0.0426	0.0218	0.0690	0.0285
	SASRec	0.0338	0.0222	0.0532	0.0285	0.0828	0.0359
	BERT4Rec	0.0293	0.0183	0.0477	0.0242	0.0688	0.0295
	CL4SRec	0.0427	0.0278	0.0648	0.0349	0.0957	0.0427
	ICLRec	<u>0.0461</u>	<u>0.0304</u>	<u>0.0728</u>	<u>0.0389</u>	<u>0.1054</u>	<u>0.0471</u>
	SparseEnNet	**0.0516**	**0.0348**	**0.0762**	**0.0426**	**0.1103**	**0.0512**
	Improv.	11.93%	14.47%	4.67%	9.51%	4.65%	8.70%
Toys	GRU4Rec	0.0211	0.0145	0.0337	0.0186	0.0536	0.0236
	SASRec	0.0399	0.0264	0.0584	0.0324	0.0832	0.0387
	BERT4Rec	0.0304	0.0199	0.0461	0.0248	0.0689	0.0305
	CL4SRec	0.0541	0.0449	0.0772	0.0374	0.1063	0.0522
	ICLRec	<u>0.0579</u>	<u>0.0395</u>	<u>0.0820</u>	<u>0.0472</u>	<u>0.1131</u>	<u>0.0550</u>
	SparseEnNet	**0.0619**	**0.0423**	**0.0855**	**0.0499**	**0.1162**	**0.0576**
	Improv.	6.91%	7.09%	4.27%	5.72%	2.74%	4.73%
Yelp	GRU4Rec	0.0152	0.0091	0.0248	0.0124	0.0371	0.0145
	SASRec	0.0160	0.0101	0.0260	0.0133	0.0443	0.0179
	BERT4Rec	0.0196	0.0121	0.0339	0.0167	0.0564	0.0223
	CL4SRec	0.0227	0.0143	0.0384	0.0194	0.0623	0.0254
	ICLRec	<u>0.0234</u>	<u>0.0145</u>	<u>0.0401</u>	<u>0.0199</u>	<u>0.0645</u>	<u>0.0260</u>
	SparseEnNet	**0.0244**	**0.0154**	**0.0414**	**0.0209**	**0.0678**	**0.0275**
	Improv.	4.27%	6.21%	3.24%	5.03%	5.12%	5.77%

eral recommendation library) or adopting the codes from the papers. We follow the general implementation method and set the maximum length of the sequence to 50, the embedding dimension of the model to 64, and the batch size to 256. For SparseEnNet, we set attention heads of the encoder as 2 and tune the self-attention layer within $\{1, 2, 3, 4\}$. We test the hyper-parameters $\lambda_1, \lambda_2, \lambda_3$ among $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5\}$, and set the cluster number K as 256. We apply dropout in our model and set dropout ratio between $0.1, 0.3, 0.5, 0, 7$. We use the Adam optimizer (Kingma and Ba 2014) to optimize the model trainable parameters with a learning of 0.001, $\beta_1 = 0.9$, $\beta_2 = 1$.

Performance Comparison (RQ1)

Table. 2 presents a comprehensive performance comparison of all methods. From the table, we can observe that:

(1) The proposed SparseEnNet outperforms all other baselines across all datasets, underscoring the effectiveness of our proposed model. Notably, SparseEnNet demonstrates remarkable performance improvements, showcasing a 11.93%, 6.91%, and 4.27% enhancement over the second-best model in terms of Hit@5 on the Beauty, Toys, and Yelp datasets, respectively. Moreover, SparseEnNet demonstrates substantial improvements over the second-best performing model across all datasets in terms of NDCG@5, with gains of 14.47%, 7.09%, and 6.21% observed on the Beauty, Toys, and Yelp datasets, respectively. Comparable enhancements are also evident across other evaluation metrics.

(2) BERT4Rec consistently outperforms GRU4Rec, affirming the efficacy of the self-attention mechanism in extracting sequence features for sequential recommendation tasks. Furthermore, SASRec exhibits superior performance compared to BERT4Rec. This divergence could potentially be attributed to the fact that the masking operation in sparse datasets leads to a more pronounced loss of contextual information within such sparse data.

(3) CL4SRec and ICLRec, tailored for sequential data augmentation, outperform the other baseline models. Notably, CL4SRec achieves the second-best performance across all datasets, underscoring the significance and effectiveness of data augmentation techniques in the realm of sequential recommendation.

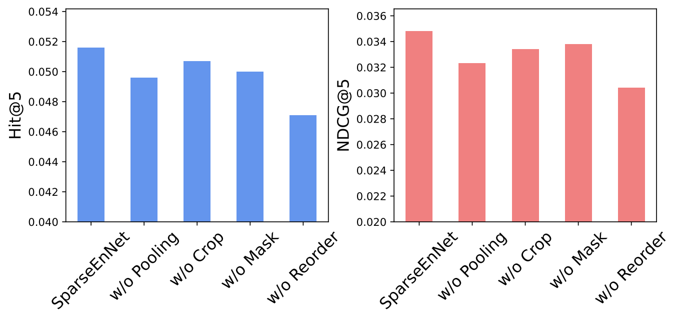


Figure 3: Different data augmentation performance in terms of Hit@5 and NDCG@5 on Beauty.

Table 3: Ablation study for SparseEnNet

Dataset	Beauty		Toys	
Metric	Hit@10	NDCG@10	Hit@10	NDCG@10
SparseEnNet	0.0762	0.0426	0.0855	0.0499
w/o D	0.0724	0.0406	0.0865	0.0497
w/o SD	0.0737	0.0407	0.0847	0.0490
w/o AD	0.0738	0.0415	0.0870	0.0500
w/o SEL	0.0713	0.0401	0.0805	0.0451
w/o NSL	0.0649	0.0347	0.0775	0.0427

Different Data Augmentation Analysis (RQ2)

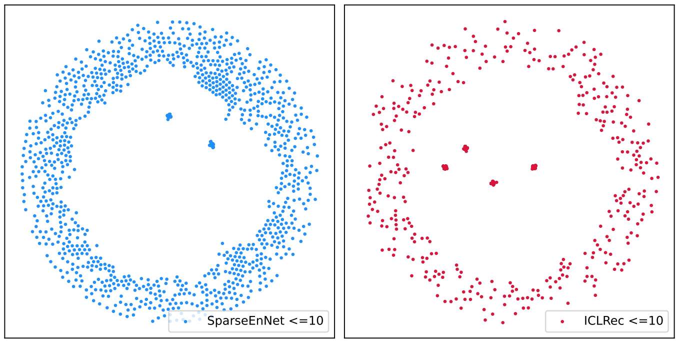
To investigate the impact of different data augmentation techniques on model performance, we present the comparative results in Fig. 3. Generally, the ranking of the significance of various augmentation methods is as follows: $Recoder > Pooling > Mask \approx Crop$. SparseEnNet achieves scores of 0.0516 and 0.0348 on Hit@5 and NDCG@5, respectively. In contrast, the performance of the model without $reorder$ data augmentation is the lowest, with corresponding scores of 0.0471 and 0.0304. Notably, SparseEnNet consistently outperforms these individual methods, underscoring the effectiveness of its holistic approach that considers a range of enhancement techniques.

Ablation Study (RQ3)

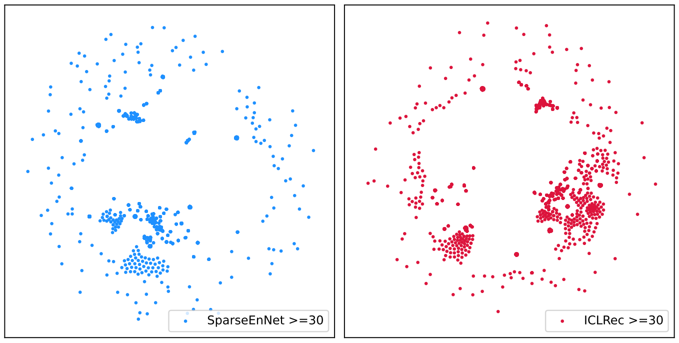
We investigated the impact of various modules on the overall performance of the model. The variants of SparseEnNet encompass the following configurations: **(1) w/o D**: removing all discriminators of the model and leaving the NSL and SEL parts. **(2) w/o SD**: removing the stability discriminator. **(3) w/o AD**: removing the augmentation discriminator. **(4) w/o SEL**: removing self-training enhanced learning. **(5) w/o NSL**: removing negative sample learning module. Table 3 presents a performance comparison between SparseEnNet and its five variants on the beauty and toys datasets. Importantly, it is evident that the NSL component plays a critical role within SparseEnNet, resulting in significant improvements in terms of Hit@10 and NDCG@10 scores. Specifically, on the Beauty dataset, NSL contributes to achieving values of 0.0649 for Hit@10 and 0.0347 for NDCG@10. Similarly, on the Toys dataset, NSL contributes to achieving values of 0.0775 for Hit@10 and 0.0427 for NDCG@10. Overall, this table underscores the efficacy of all the designed components within our model.

Cold-start Problem by Data Sparsity (RQ4)

For the cold-start performance assessment, we have chosen ICLRec, the second-best baseline. Initially, we randomly selected 2000 items that appear in user-item interactions with a count of less than or equal to 10. This was done to simulate cold-start item embeddings. As evident from Figure 4a, it is evident that SparseEnNet is able to encode items into a more condensed embedding space compared to ICLRec. Additionally, we conducted a comparative learning by selecting 2000 items that occur in user-item interactions with a count of 30 or more to simulate embeddings for popular items. The findings from Figure 4b indicate that our model performs at a similar level to ICLRec in this scenario. These outcomes underscore the effectiveness of our proposed model in addressing the cold-start issue.



(a) Simulating cold-start item embeddings for user-item interactions $<=$ 10 on SparseEnNet and ICLRec.



(b) Simulating popular item embeddings for user-item interactions $>=$ 30 on SparseEnNet and ICLRec.

Figure 4: Cold-start performance in sparsity data.

Conclusion

In this paper, we present SparseEnNet, an innovative and robust adversarial generation method designed to thoroughly explore the latent space in sequence recommendation by generating enhanced items that are more robust. The SparseEnNet framework consists of four essential components: an augmentation discriminator, a stability discriminator, a negative sample learning module, and a self-training enhanced learning module. Through extensive experiments conducted on three well-known datasets, we establish the efficacy of our model. Additionally, our ablation study further validates the contribution of each individual component. Furthermore, a case study focusing on the cold-start problem confirms the ability of SparseEnNet to produce distinct item embeddings in sparse datasets.

Acknowledgments

This work was supported by NSFC (62102265), Shenzhen University Stable Support Program Project (20231120161634002), Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ, GML-KF-22-29), Natural Science Foundation of Guangdong Province (2022A1515011474), GDST (2020B1212030003).

References

Akcay, S.; Atapour-Abarghouei, A.; and Breckon, T. P. 2019. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, 622–637. Springer.

Chen, J.; Wang, J.; Dai, Z.; Wu, H.; Wang, M.; Zhang, Q.; and Wang, H. 2023. Zero-shot Micro-video Classification with Neural Variational Inference in Graph Prototype Network. In *Proceedings of the 31st ACM International Conference on Multimedia*, 966–974.

Chen, T.; Kornblith, S.; Norouzi, M.; and Hinton, G. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, 1597–1607. PMLR.

Chen, Y.; Liu, Z.; Li, J.; McAuley, J.; and Xiong, C. 2022. Intent contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*, 2172–2182.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fang, H.; Wang, S.; Zhou, M.; Ding, J.; and Xie, P. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1): 2096–2030.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. *Advances in neural information processing systems*, 27.

He, K.; Fan, H.; Wu, Y.; Xie, S.; and Girshick, R. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 9729–9738.

He, X.; He, Z.; Du, X.; and Chua, T.-S. 2018. Adversarial personalized ranking for recommendation. In *The 41st International ACM SIGIR conference on research & development in information retrieval*, 355–364.

Hidasi, B.; Karatzoglou, A.; Baltrunas, L.; and Tikk, D. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Hjelm, R. D.; Fedorov, A.; Lavoie-Marchildon, S.; Grewal, K.; Bachman, P.; Trischler, A.; and Bengio, Y. 2018. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*.

Ji, W.; Wang, K.; Wang, X.; Chen, T.; and Cristea, A. 2020. Sequential recommender via time-aware attentive memory network. In *Proceedings of the 29th ACM international conference on information & knowledge management*, 565–574.

Kang, W.-C.; and McAuley, J. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, 197–206. IEEE.

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

Li, J.; Wang, Y.; and McAuley, J. 2020. Time interval aware self-attention for sequential recommendation. In *Proceedings of the 13th international conference on web search and data mining*, 322–330.

Liu, Z.; Chen, Y.; Li, J.; Yu, P. S.; McAuley, J.; and Xiong, C. 2021. Contrastive self-supervised sequential recommendation with robust augmentation. *arXiv preprint arXiv:2108.06479*.

McAuley, J.; Targett, C.; Shi, Q.; and Van Den Hengel, A. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, 43–52.

Ren, R.; Liu, Z.; Li, Y.; Zhao, W. X.; Wang, H.; Ding, B.; and Wen, J.-R. 2020. Sequential recommendation with self-attentive multi-adversarial network. In *Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval*, 89–98.

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

Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; and Jiang, P. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, 1441–1450.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; and Xie, X. 2021. Self-supervised graph learning for recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, 726–735.

Xia, X.; Yin, H.; Yu, J.; Wang, Q.; Cui, L.; and Zhang, X. 2021. Self-supervised hypergraph convolutional networks for session-based recommendation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 4503–4511.

Xie, X.; Sun, F.; Liu, Z.; Wu, S.; Gao, J.; Zhang, J.; Ding, B.; and Cui, B. 2022. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, 1259–1273. IEEE.

Yao, T.; Yi, X.; Cheng, D. Z.; Yu, F.; Chen, T.; Menon, A.; Hong, L.; Chi, E. H.; Tjoa, S.; Kang, J.; et al. 2021. Self-supervised learning for large-scale item recommendations. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 4321–4330.

Zhou, K.; Wang, H.; Zhao, W. X.; Zhu, Y.; Wang, S.; Zhang, F.; Wang, Z.; and Wen, J.-R. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM in-*

ternational conference on information & knowledge management, 1893–1902.

Zimdars, A.; Chickering, D. M.; and Meek, C. 2001. Using temporal data for making recommendations. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, 580–588.