Diffusion Augmentation for Sequential Recommendation

Qidong Liu
Xi'an Jiaotong University & City
University of Hong Kong
liuqidong@stu.xjtu.edu.cn

Fan Yan
Huawei Noah's Ark Lab
yanfan6@huawei.com

Xiangyu Zhao ✉
City University of Hong Kong
xianzhao@cityu.edu.hk

Zhaocheng Du
Huawei Noah's Ark Lab
zhaochengdu@huawei.com

Huifeng Guo
Huawei Noah's Ark Lab
huifeng.guo@huawei.com

Ruiming Tang ✉
Huawei Noah's Ark Lab
tangruiming@huawei.com

Feng Tian
Xi'an Jiaotong University
fengtian@mail.xjtu.edu.cn

ABSTRACT

Sequential recommendation (SRS) has become the technical foundation in many applications recently, which aims to recommend the next item based on the user's historical interactions. However, sequential recommendation often faces the problem of data sparsity, which widely exists in recommender systems. Besides, most users only interact with a few items, but existing SRS models often underperform these users. Such a problem, named the long-tail user problem, is still to be resolved. Data augmentation is a distinct way to alleviate these two problems, but they often need fabricated training strategies or are hindered by poor-quality generated interactions. To address these problems, we propose a **Diffu**sion **A**ugmentation for **S**equential **R**ecommendation (**DiffuASR**) for a higher quality generation. The augmented dataset by DiffuASR can be used to train the sequential recommendation models directly, free from complex training procedures. To make the best of the generation ability of the diffusion model, we first propose a diffusion-based pseudo sequence generation framework to fill the gap between image and sequence generation. Then, a sequential U-Net is designed to adapt the diffusion noise prediction model U-Net to the discrete sequence generation task. At last, we develop two guide strategies to assimilate the preference between generated and origin sequences. To validate the proposed DiffuASR, we conduct extensive experiments on three real-world datasets with three sequential recommendation models. The experimental results illustrate the effectiveness of DiffuASR. As far as we know, DiffuASR is one pioneer that introduce the diffusion model to the recommendation. The implementation code is available online [1][2].

✉ Xiangyu Zhao and Ruiming Tang are the corresponding authors.

CCS CONCEPTS

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

KEYWORDS

Sequential Recommendation; Data Augmentation; Diffusion Model

1 INTRODUCTION

The recommender system (RS) has shown promising developments in recent years. *Sequential Recommendation* (SRS), as one important branch of RS, aims to recommend the next item for users based on their historical interactions [7, 42]. Thanks to the advancement of deep learning, SRS has been widely used in many applications, such as e-commerce [34, 41] and smart education [51]. The core of SRS is to capture the general pattern from the user's interaction sequences. Some recent works combine with transformer architecture to get such patterns and boost recommending performance, such as Bert4Rec [37] and SASRec [14].

However, there exist two severe problems, which hinder the development of the sequential recommendation, *i.e.*, data sparsity and long-tail user problem. (i) **Data Sparsity Problem**. Sparsity is an inherent nature of real-world data, which means that only a few interactions occur among a large volume of users and items. It easily causes the sub-optimal problem for sequential recommendation [46]. (ii) **Long-tail User Problem**. This problem refers that users with few interactions often get worse recommendations. For illustration, we train a well-known SRS model, *i.e.*, Bert4rec [37], on Beauty, a real-world dataset, and show the performance grouped by the user's historical interaction number. In Figure 1, the histogram shows the number of users in each group and the line shows the recommending performance of each group. We can conclude that most users only own a few interactions, *i.e.*, interaction sequences shorter than 10, who are so-called long-tail users. However, existing SRS
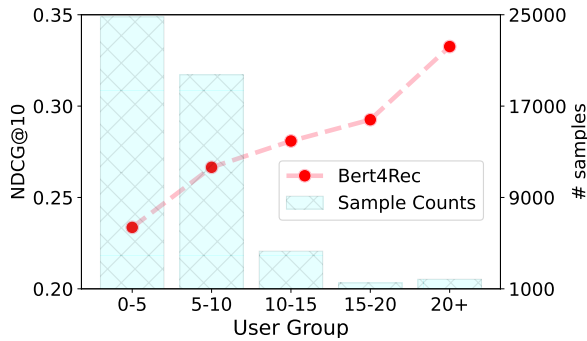
Figure 1: The illustration for long-tail user problem.

Figure 2: Two types of data augmentation methods for SRS.

models often perform quite poorly for these users [13, 26], which makes most users unsatisfied with the recommender systems.

Faced with the two problems, data augmentation is instinctive, because it can enrich the whole interaction records. We can categorize the existing data augmentation methods for SRS into two types, *i.e.,* sequence-level and item-level methods. (i) **Sequence-level Method**. As shown in Figure 2, this line of methods [3, 40, 46, 53] often generates new sequences or users to handle the problem of data sparsity. However, due to the noise contained in generated sequences, most sequence-level methods fabricate complex training strategies. For example, L2Aug [40] adopts reinforcement learning to update SRS models and sequence generators iteratively. Wang *et al.* [46] propose to optimize the sequence generator based on a pre-trained SRS model and retrain the recommender on both original and augmented sequences. Compared with training models on augmented datasets directly, sequence-level methods are not convenient. Also, they are not tailored for long-tail users specifically. (ii) **Item-level Method**. As presented in Figure 2, this category [13, 26] generates pseudo pre-order items for the interaction sequence of each user. In general, the augmented dataset by this category can be used to train the SRS models directly. Besides, item-level methods can enhance the long-tail users, because they prolong the short interaction sequence. Nevertheless, ASReP [26] and BiCAT [13] focus on how to use the augmented dataset more efficiently but ignore the augmentation quality itself. They both train a SASRec [14] reversely as an item generator and get the augmented pre-order items iteratively. However, the top-1 accuracy of the existing SRS model is relatively low, so the iterative generation may cause the challenge of poor quality augmentation and distribution drift from the original sequence. These challenges block the benefits of data augmentation for data sparsity and long-tail user problems.

To handle the challenges mentioned above, we propose a **Diffu**sion **A**ugmentation for **S**equential **R**ecommendation (**DiffuASR**). The adoption of the diffusion model has two-fold merits. First, the diffusion model has been proven that it is powerful to generate high-quality contents, such as images [31, 33] and texts [9, 20, 21], so it has the potential to generate high-quality augmented items. Second, we adopt the diffusion model to generate the sequence in one run, which can guarantee that the whole augmented sequence has a more similar distribution to the original one. However, some problems emerge while combining the diffusion model with the sequential recommendation. First, the diffusion model is initially designed to generate an image, a continuous-valued matrix. By comparison, the interaction history
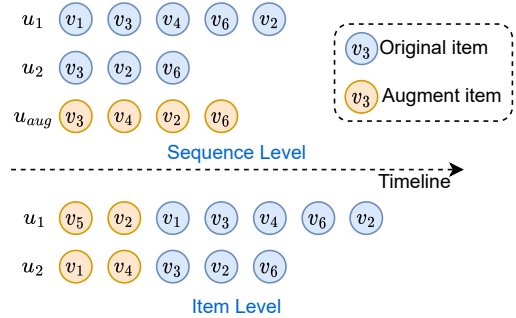
is a 1-dimensional discrete-valued sequence, which is hard to be generated directly from the diffusion model. To fill the gap between image and sequence generation, we derive a diffusion-based pseudo sequence generation framework for augmenting sequential recommendation. Then, U-Net [32] is an essential component for the general diffusion model, but it is difficult to capture the sequential information of the embedding sequence. Therefore, we design an SU-Net compatible with the embedding sequence. Besides, the augmented pseudo items should be relevant to the preference of the user's original intent, because not relevant items may cause noise for training [2, 50]. Inspired by the guided diffusion model [5, 12], we design a classifier-guide and a classifier-free strategy for DiffuASR to guide the generation process. The contributions of this paper are concluded as follows:

- We design a novel data augmentation framework based on diffusion model for the sequential recommendation. This framework fills the gap between discrete-valued item identity and generated real-valued matrix, which is one of the pioneer works to combine the diffusion model with the recommender system;
- To fit the U-Net for the sequence generation task, we devise an SU-Net that can recover the embedding sequence matrix. Also, to guide the diffusion model to generate the items that are more corresponding to the user's intent, we design a classifier-guide and a classifier-free strategy for DiffuASR;
- We conduct comprehensive experiments on three real-world datasets to illustrate the effectiveness of our augmentation method. Besides, the experimental results on three different sequential recommendation models show the generality of the DiffuASR.

2 PRELIMINARY

In this section, we will introduce the problem definition and notations used in this paper briefly. Besides, some fundamentals of the diffusion model are presented.

2.1 Problem Definition and Notations

The sequential recommendation aims to recommend the next item for users according to their historical interactions, so the records of one user are often arranged by timeline. If we let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ denote the set of users and items, respectively, the interaction history of one user u can be written as $\mathcal{S}_u = \{v_1^{(u)}, ..., v_k^{(u)}, ..., v_{n_u}^{(u)}\}$. n_u is the number of users' historical interactions. We omit the sub-script (u) in the
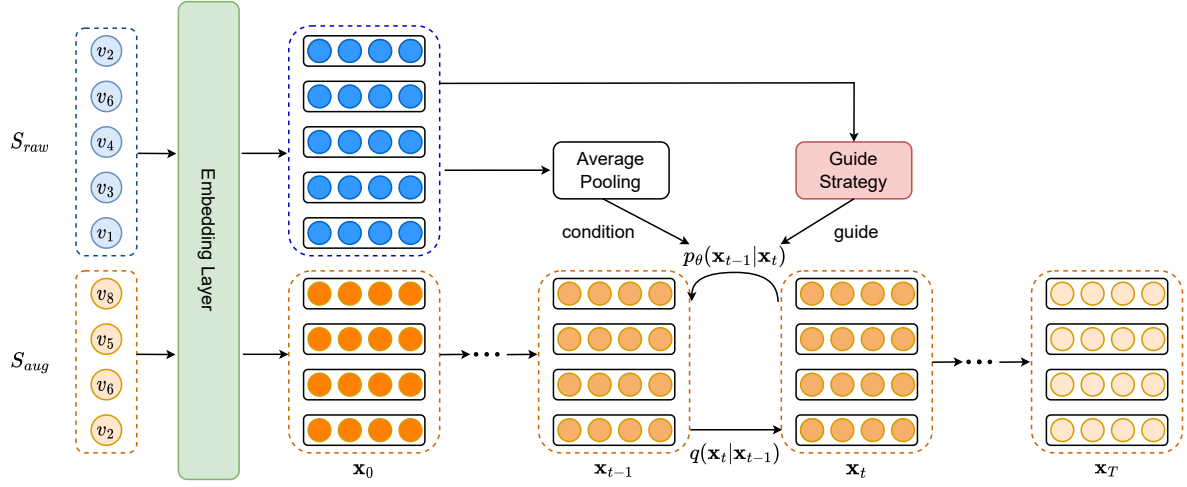
Figure 3: The overview of the proposed Diffusion Augmentation for Sequential Recommendation (DiffuASR).

following sections for simplicity. Then, the problem of sequential recommendation can be presented as: *Given a user's interaction history S_u, predict the most possibly interacted item at step $n_u + 1$,* i.e., v_{n_u+1}. It is formulated as follows:

$$arg \max_{v_i \in \mathcal{V}} P(v_{n_u+1} = v_i | S_u) \quad (1)$$

Item-level augmentation methods generate the pre-order items for the raw sequence. For example, given a historical sequence $S_{raw} = \{v_1, ..., v_k, ..., v_{n_u}\}$, the augmentation items are inserted before the v_1. Let $S_{aug} = \{v_{-M}, v_{-M+1}, ..., v_{-1}\}$ denote the augmented sequence for S_{raw}, where M is the number of augmented items. Then, the original sequence is transformed into $S' = \{v_{-M}, ..., v_{-1}, v_1, ..., v_{n_u}\}$ by augmentation. As illustrated in Section 1, the goal in this paper is to generate high-quality S_{aug}.

2.2 Diffusion Model

Diffusion model [11] has become prevalent in recent years, due to its strong capability in various generation tasks, such as images [31, 33] and texts [9, 20, 21]. For an easier understanding of the proposed method, we will briefly introduce the fundamental knowledge of the diffusion model before presenting DiffuASR. In general, the diffusion model is composed of two processes, *i.e.,* forward process and the reverse process. The forward process is gradually adding noise to original data samples, which perturbs samples approaching a Gaussian noise. On the contrary, the reverse process recovers the original sample step by step via predicting the added noise. Then, we will detail these two processes.

Forward Process. Let $x_0 \sim q(x_0)$ denotes the original data samples. The forward process is a Markov Chain that gradually adds Gaussian noise to x_0, which can be formulated as follows:

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1})$$
$$q(x_t|x_{t-1}) := \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \quad (2)$$

where T is the number of steps to add noise and $\beta_1, ..., \beta_T$ are the variance schedule. Since all data samples are perturbed to normal Gaussian noise, the diffusion step number T and variance schedule

$\{\beta_t\}_{t=1}^{T}$ are adjustable to control how much noise is added at each diffusion step. However, it is difficult to learn the variance β_t directly, so a reparameterization trick [17] is adopted here and thus transforms the forward process as follows:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_t, (1-\bar{\alpha}_t)I) \quad (3)$$

where $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.

Reverse Process. The reverse process aims to recover the original data sample x_0 from the completely perturbed $x_T \sim \mathcal{N}(0, I)$. We can formulate the reverse process by the Markov chain as:

$$p_\theta(x_{0:T}) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t)$$
$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_t; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \quad (4)$$

where $\mu_\theta(x_t, t)$ and $\Sigma_\theta(x_t, t)$ are the mean and variance that are parameterized by θ. Based on the Gaussian distribution parameterization [11], we can set $\Sigma_\theta(x_t, t) = \sigma_t^2 I$ as constants, where $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$. As for the mean of the distribution, it can be written as:

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}})\epsilon_\theta(x_t, t) \quad (5)$$

Therefore, the diffusion models are mainly parameterized by the ϵ_θ, which is usually an U-Net [32] based architecture.

Optimization: The objective of the general diffusion model is to pull the posterior distribution in the forward process and prior distribution in the reverse process closer, so the objective function can be written as the KL divergence form. For simplification, it can also be formulated as [11]:

$$L_{simple} = \mathbb{E}_{t,x_0,\epsilon}[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2] \quad (6)$$

3 DIFFUASR

In this section, we will detail the design of our Diffusion Augmentation for Sequential Recommendation (DiffuASR). As a start, we present the overview of the augmentation framework in Section 3.1. In Section 3.2, we will refer to the proposed SU-Net architecture used to recover the embedding sequence. Next, we introduce how to guide the diffusion model to generate pseudo items corresponding
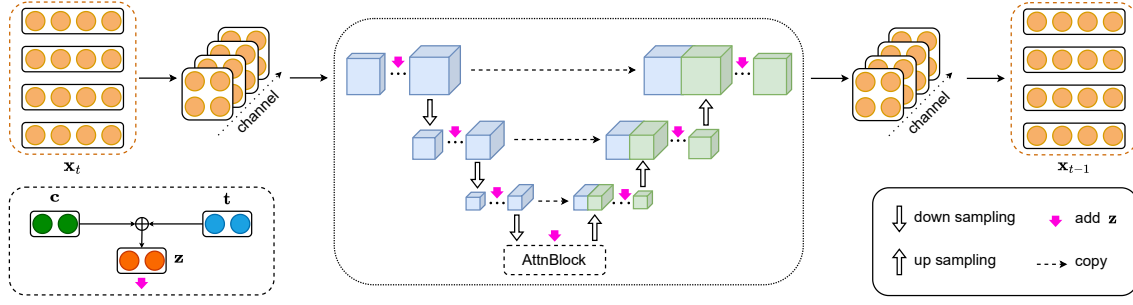
Figure 4: The architecture of the proposed SU-Net.

to the raw sequence in Section 3.3. Finally, the train and augment process are illustrated in Section 3.4.

3.1 Framework

As Figure 3 shows, we design the DiffuASR framework for data augmentation. The augmented sequence S_{aug} is the generative objective of the diffusion model, while the S_{raw} is the raw sequence of the user. The task of data augmentation for sequential recommendation is to generate a 1-dimensional discrete-valued sequence, while general diffusion models produce a 2-dimensional continuous-valued image matrix. To fill this gap, we derive the embedding procedure for the forward process and the rounding procedure for the reverse process. Then, the augmented sequence S_{aug} should have a similar distribution of S_{raw} to avoid noisy training [2, 50], so we devise the guide procedure for reverse process to assimilate them. In the rest of this section, we will detail the DiffuASR framework according to the forward process, reverse process and guide procedure.

3.1.1 **Forward Process**. As illustrated in Section 2, the augmented sequence is a series of item identities, which can be written as $S_{aug} = \{v_{-M}, v_{-M+1}, ..., v_{-1}\}$. To get the representation of each item, we adopt a learnable embedding table to transform the item identity v_i into dense representation $\mathbf{e}_i$. Let $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ denotes the embedding table, where d is the dimension of embedding. After the embedding table, we can get the stacked item embedding (an embedding sequence), denoted as $\mathbf{x}_0 = \{\mathbf{e}_{-M}, \mathbf{e}_{-M+1}, ..., \mathbf{e}_{-1}\}$. Here, the 1-dimensional discrete-valued sequence is converted to a 2-dimensional continuous-valued matrix, similar to the input type of the general diffusion model. Then, following the Equation (2), we add the Gaussian noise to the embedding sequence $\mathbf{x}_t$ iteratively and get the $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I}^{M \times d})$. During the forward process, except for the diffusion step T, the variance schedule is also important for controlling the noise intensity in each step. The linear schedule is adopted in DiffuASR due to better performance.

3.1.2 **Reverse Process**. The task of the reverse process is to recover the embedding sequence $\mathbf{x}_0$ from $\mathbf{x}_T$. According to Equation (4), this process removes the Gaussian noise step by step, so the core of the reverse process is to predict the added noise for each diffusion step. After simplification, the noise can be predicted by the $\epsilon_\theta(\mathbf{x}_t, t)$ as the Equation (5) shows. In general, $\epsilon_\theta(\mathbf{x}_t, t)$ is implemented by U-Net [32]. However, U-Net often absorbs the image matrix instead of sequence embedding, which contains different meanings along the column and row dimensions. Therefore, we propose a sequential U-Net (SU-Net) for ϵ_θ, which will be referred

to in Section 3.2. Finally, the predicted $p_\theta(x_{t-1}|x_t)$ helps recover the augmented embedding sequence $\hat{\mathbf{x}}_t = \{\hat{\mathbf{e}}_{-M}, \hat{\mathbf{e}}_{-M+1}, ..., \hat{\mathbf{e}}_{-1}\}$. To get the augmented sequence, we design a rounding procedure. This procedure uses the cosine similarity function to decide the augmented item for each position $j \in [-M, -1]$:

$$v_j = \max_{v_i \in \mathcal{V}} \text{sim}(\hat{\mathbf{e}}_j, \mathbf{e}_i) \tag{7}$$

where $\mathbf{e}_i$ is the embedding of item $v_i \in \mathcal{V}$ and $\text{sim}(\cdot, \cdot)$ represents the cosine similarity function. The reason for using cosine similarity here is that the generated embedding has a different embedding length compared with the original item embedding.

3.1.3 **Guide Procedure**. Only forward and reverse processes cannot guarantee that the generated contents are desired, because no control signal is injected while generating. Thus, a guide procedure is needed. Specifically, we hope the generated items can correspond to the preference of users. Users' preference is often contained in their historical interactions, so we utilize S_{raw} to guide the generation process. The guide procedure is in two folds. On the one hand, we add the information of S_{raw} to ϵ_θ, because the predicted noise controls the direction of generation. In detail, the embedding sequence is converted to conditional vector $\mathbf{c} = \text{Avg}(\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_{n_u})$ and injected to $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{c})$. On the other hand, we design two guide strategies for further guidance, which will be detailed in Section 3.3.

3.2 SU-Net

In this section, we will introduce the proposed Sequential U-Net (SU-Net). U-Net is widely used in the diffusion model, but it is devised for images and may lose the sequence information if we use it to handle the embedding sequence directly. Therefore, we propose altering the U-Net input to fit the embedding sequence.

The architecture of SU-Net is shown in Figure 4. To capture the sequential information of $\mathbf{x}_t$, we set the sequence dimension as the channel of the image, because the channel is modeled independently in the convolutional networks. However, the input of the image matrix becomes 1-dimensional, *i.e.,* only the embedding dimension. For full usage of convolutional neural networks, we reshape each 1-dimensional embedding into a matrix with the shape of $(\sqrt{d}, \sqrt{d})$. For example, if the size of the embedding vector is 64, the embedding can be converted to a 8×8 matrix. As a result, the input embedding sequence is transformed into an M channels matrix with $\sqrt{d}$ length and width, which will be input into the main architecture of SU-Net.

The main architecture of SU-Net is similar to the U-Net [32]. It contains several steps of down-sampling and up-sampling. Before the down-sampling, multiple resnet blocks with convolutional

neural networks are imposed on the multi-channel matrix. Then, a middle AttnBlock equipped with a resnet and attention layer is used to remove the redundant information. During the up-sampling process, the matrix from down-sampling is copied and concatenated to fuse more features. The output matrix has the same size as the input, so we reshape each $(\sqrt{d}, \sqrt{d})$ matrix back to the 1-dimensional embedding vector to get the $\mathbf{x}_{t-1}$.

The diffusion step t and conditional vector $\mathbf{c}$ are also vital for $\epsilon_\theta(\mathbf{x}_t, t, \mathbf{c})$ to learn the noise. To represent the diffusion step, we convert the scalar t into a vector $\mathbf{t}$ by sinusoidal embedding technique [39], which has the same size of $\mathbf{c}$. Then, we add the step vector and conditional vector, $i.e.$, $\mathbf{z} = \mathbf{c} + \mathbf{t}$. Finally, $\mathbf{z}$ is added to each resnet block of SU-Net architecture for the injection of these two types of information. Specifically, $\mathbf{z}$ is transformed to the same size of the input matrix by linear layer and added to the input matrix before the convolution layer. Thus, the diffusion step and conditional vector can control the ϵ_θ to predict the noise.

3.3 Guide Strategy

High-quality augmentation should correspond to the preference contained in the raw user's interaction sequence, because unrelated items can be considered as noisy interactions and thus harm the sequential recommendation model [2, 50]. To make the augmented items more relevant to the original sequences, we design two guide strategies at the reverse process for DiffuASR, $i.e.$, classifier guide and classifier free. Then, we will detail these two strategies.

3.3.1 Classifier Guide Strategy.
In the reverse process, each step updates towards the gradient direction of raw data distribution. For this reason, Dhariwal $et\ al.$ [5] propose to add a gradient of image classifier to ϵ to guide the generation towards a specific class, so the noise prediction can be reformulated as follows:

$$\hat{\epsilon} = \epsilon_\theta(\mathbf{x}_t) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \nabla_{\mathbf{x}_t} log\ p_\phi(y|\mathbf{x}_t) \tag{8}$$

where γ is the scale for guidance and $p_\phi(\cdot)$ represents the classifier. However, the recommendation task is different from the classification task, which makes the original classifier unsuitable. The augmented sequence S_{aug} is the pre-order sequence of the first item in S_{raw}, so v_1 can be regarded as the next recommendation item of S_{aug}. Inspired by this thought, we propose the sequential recommendation model as the "classifier" and the Equation (8) in DiffuASR can be written as follows:

$$\hat{\epsilon} = \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}) - \gamma \cdot \sqrt{1 - \bar{\alpha}_t} \nabla_{\mathbf{x}_t} log\ p_\phi(v_1|S_{aug}) \tag{9}$$

where ϕ is a pre-trained sequential recommendation model and $p(\cdot)$ is binary cross-entropy loss instead of cross-entropy loss in [5].

3.3.2 Classifier Free Strategy.
However, the classifier guide strategy needs a pre-trained "classifier", which costs much time and computation. To free from the laborious pre-training procedure, we propose to utilize the classifier free strategy [12] as the other variant of our DiffuASR. The predicted noise of classifier free strategy in the reverse process is formulated as follows:

$$\hat{\epsilon} = (1 + \gamma) \cdot \epsilon_\theta(\mathbf{x}_t, t, \mathbf{c}) - \gamma \cdot \epsilon_\theta(\mathbf{x}_t, t, \mathbf{e}_{padding}) \tag{10}$$

where γ is the scale for guidance and $\mathbf{e}_{padding}$ is the padding vector.

Algorithm 1 Train and Augment process of DiffuASR

1: Indicate the augmented number M for each sequence.
2: Indicate the variance schedule and calculate $\{\beta_t\}_{t=1}^T$.
3: Indicate the guidance scale γ and guide strategy.
4: Pre-train a sequential recommendation model ϕ.
5: Select out the sequence with $n_u > M$ from $\mathcal{D}$ to form $\mathcal{D}_T$.

Train Process

6: **for** S in $\mathcal{D}_T$ **do**
7:     Take out the first M items as S_{aug} and the rest as S_{raw}.
8:     Conduct the forward process by Equation (3).
9:     **if** Guide Strategy is Classifier Guide **then**
10:         Calculate the $\hat{\epsilon}$ by Equation (9).
11:     **else**
12:         Calculate the $\hat{\epsilon}$ by Equation (10).
13:     **end if**
14:     Conduct the reverse process by Equation (4).
15:     Calculate the loss by Equation (6) and update the parameters θ by gradient descent.
16: **end for**

Augment Process

17: Set the augmented dataset $\mathcal{D}_A$
18: **for** S in $\mathcal{D}$ **do**
19:     **if** Guide Strategy is Classifier Guide **then**
20:         Calculate the $\hat{\epsilon}$ by Equation (9).
21:     **else**
22:         Calculate the $\hat{\epsilon}$ by Equation (10).
23:     **end if**
24:     Conduct the reverse process by Equation (4) based on S and ϵ.
25:     Get the augment sequence $\hat{S}_{aug}$ and add $[\hat{S}_{aug}, S]$ to $\mathcal{D}_A$.
26: **end for**
27: Get the augmented dataset $\mathcal{D}_A$.

3.4 Train and Augment

In this section, we will introduce the train and augment process of DiffuASR. We conclude the process in Algorithm 1 to be more readable. At first, we need to indicate some hyper-parameters in advance (lines 1-3), such as augmented number M and guidance scale γ. Then, we make up the dataset $\mathcal{D}_T$ for training DiffuASR (line 5). Since we do not have the ground-truth augment sequence, the pre-order items in the raw sequence are regarded as augmentation for training. Thus, only the sequences with a length longer than the augmented number M can be used for training DiffuASR. Then, we take out the first M items in a sequence for diffusion (line 7). Next, the forward and reverse processes are conducted and the parameters θ of SU-Net are updated (lines 8-15). When the model gets convergence, the train process ends. As for the augment process, the guided reverse process is conducted based on well-trained ϵ_θ (line 19-24) to generate pre-order sequence $\hat{S}_{aug}$. Then, $\hat{S}_{aug}$ is concatenated before the raw sequence, which forms the augmented sequence. After augmentation for each raw sequence in $\mathcal{D}$, we can get the augmented dataset $\mathcal{D}_A$. Unlike the sequence-level augmentation methods, $\mathcal{D}_A$ can be directly used for training sequential recommendation models to enhance their performance.

Table 1: The Statistics of Datasets

Dataset	# Users	# Items	Sparsity	Avg.length
Yelp	30,431	20,033	99.95%	10.40
Beauty	50,498	57,289	99.99%	7.76
Steam	332,925	13,047	99.91%	11.06

4 EXPERIMENT

In this section, we will introduce the extensive experiments conducted on three real-world datasets. We analyze the experimental results to illustrate the following research problems (**RQ**):

- **RQ1:** How DiffuASR performs compared with other item-level augmentation methods? Is DiffuASR a general augmentation method for various sequential recommendation models?
- **RQ2:** Can DiffuASR alleviate the long-tail user problem?
- **RQ3:** How does the number of augmented items affect the recommending performance of DiffuASR?
- **RQ4:** How do the hyper-parameters of the diffusion model affect the performance of DiffuASR?

4.1 Experiment Settings

*4.1.1 **Dataset**.* We conduct comprehensive experiments on three representative real-world datasets, *i.e.,* Yelp, Amazon Beauty and Steam. **Yelp**[3] is a business recommendation dataset, which records the check-in history of users. We only use the interactions that occurs after Jan 1st, 2019, since the whole dataset is too large. **Beauty**[4] is an review dataset on an e-commerce platform. Amazon is a large-scale product review dataset published in [27] and Beauty is one sub-category. **Steam**[5] dataset collects the players' reviews for video games on the Steam platform. This dataset is published in [14].

We preprocess all the datasets referring to [14, 37]. For sequential recommendation task, we regard all reviewed or rated records as implicit interactions. Then, to fulfill the test split requirement and investigate the long-tail users, we only remove the users with fewer than three interactions. As for data split, we take the last item v_{n_u} as the test and the penultimate item v_{n_u-1} as the validation. The rest of the raw sequence is split into the training data. After the preprocessing, the statistics of the datasets are shown in Table 1.

*4.1.2 **Baselines**.* To show the effectiveness of the proposed augmentation method, we compare DiffuASR with some item-level baseline methods:

- **None**. No augmentation for this baseline. The recommendation models are trained directly on the original dataset $\mathcal{D}$.
- **Random**. Augment each sequence by randomly selecting items from the whole item set, *i.e.,* $v_{aug} \in \mathcal{V}$.
- **Random-Seq**. Select items from the original sequence randomly as the augmented items, *i.e.,* $v_{aug} \in \mathcal{S}_u$.
- **ASReP** [26]. This baseline train a SASRec [14] model in a reverse pattern and generate the augmentation item iteratively. The pretrain-finetune scheme used in this baseline can only be compatible with SASRec. In contrast, we want to test the generality for several SRS models and focus on the generation quality.

[3]https://www.yelp.com/dataset
[4]https://cseweb.ucsd.edu/~jmcauley/datasets.html#amazon_reviews
[5]https://cseweb.ucsd.edu/~jmcauley/datasets.html#steam_data

Thus, we only use the reverse SASRec as the augmentor and do not adopt extra training tricks. The BiCAT [13] uses the same augmentation method, so we do not discuss about it further.

*4.1.3 **Backbone Recommendation Models**.* Generality is a vital characteristic of the data augmentation method. To illustrate whether the proposed DiffuASR possesses such characteristic, we combine competing augmentation methods and our DiffuASR with several sequential recommendation models:

- **Bert4Rec [37]**. It is a sequential recommendation model equipped with bi-directional self-attention layers. Besides, it adopts the cloze task to train the model, which is different from general sequential recommendation models.
- **SASRec [14]**. Different from Bert4Rec, SASRec proposes a causal self-attention layer to capture the transitional patterns of the user's interaction sequence.
- **S3Rec [55]**. S3Rec designs four self-supervised tasks to embed the information of item attributes and sequence correlation into the model. Then, it finetunes the model using the next item prediction task. Since the referred augmentation methods do not consider the item attribute, we only implement masked item prediction and segment prediction tasks for pre-train.

*4.1.4 **Implementation Details**.* We choose the best hyper-parameters by the HR@10 metric on the validation set. For DiffuASR, we tune the augment number M from $\{4, 6, 8, 10, 12, 14, 16\}$ and the guidance scale γ from $\{0.1, 1, 10, 100\}$. The number of diffusion steps is fixed at 1,000. We set the dimension of item embedding as 64. After augmentation, $\mathcal{D}_A$ is used to train the sequential recommendation model directly. As for the sequential recommendation model, we fix the maximum sequence length as 200 and the dropout rate as 0.6 for all three datasets. Other hyper-parameters of the three SRS models follow the original paper. For Bert4Rec, the cloze task is used for training. For SASRec and finetune stage of S3Rec, sequence-to-one training scheme is adopted. To optimize the DiffuASR and sequential recommendation models, we apply Adam optimizer [16], and set the batch size and learning rate as 512 and 0.001, respectively. It is worth noting that, during the train and augment process of DiffuASR and competing methods, we follow [15] to exclude the test item of each sequence to avoid data leakage. For the stability of the results, all reported performances are averaged on three runs of SRS models with different random seeds. The implementation code is available online to ease reproducibility[6][7].

*4.1.5 **Evaluation Metrics**.* In this paper, we adopt *Top-10 Hit Rate* (**HR@10**) and *Top-10 Normalized Discounted Cumulative Gain* (**NDCG@10**) as the evaluation metrics, which have been used in related works [14, 37, 55]. HR evaluates the capacity to recommend correct items that appear in the Top-K list. NDCG can reflect the ranking ability of the recommendation models. Following [14, 55], we adopt the negative sampling strategy, *i.e.,* the ground-truth item paired with 100 randomly sampled negatives. All the reported results are averaged over all test users.

[6]https://github.com/liuqidong07/DiffuASR
[7]https://gitee.com/mindspore/models/tree/master/research/recommend/DiffuASR

Table 2: The overall results of competing augmentation methods and DiffuASR on three datasets. Boldface refers to the highest score and underline indicates the best result of the baselines. "*" indicates the statistically significant improvements (*i.e.,* two-sided t-test with $p < 0.05$) over the best baseline.

Backbone	Augment Method	Yelp		Beauty		Steam	
		NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10
Bert4Rec	- None	0.4899	0.7702	0.2581	0.4302	0.6104	0.8561
	- Random	0.4559	0.7464	0.2634	0.4297	0.6022	0.8505
	- Random-Seq	0.4703	0.7269	0.2509	0.4257	0.5771	0.8306
	- ASReP	0.4714	0.7563	0.2513	0.4102	0.5765	0.8289
	- DiffuASR (CG)	**0.5004***	**0.7820***	**0.2818***	**0.4462***	**0.6173***	**0.8615***
	- DiffuASR (CF)	**0.5066***	**0.7828***	0.2752*	0.4385*	0.6156*	0.8609*
SASRec	- None	0.4813	0.7590	0.2844	0.4343	0.5829	0.8323
	- Random	0.4817	0.7590	0.2808	0.4293	0.5837	0.8314
	- Random-Seq	0.4546	0.6862	0.2746	0.4204	0.5510	0.8055
	- ASReP	0.4758	0.7604	0.2882	0.4364	0.5866	0.8348
	- DiffuASR (CG)	0.4872*	**0.7623***	**0.2987***	**0.4424***	**0.5899***	**0.8371***
	- DiffuASR (CF)	**0.4885***	0.7617	0.2952*	0.4380	0.5891*	0.8362
S3Rec	- None	0.4832	0.7632	0.2908	0.4372	0.5755	0.8245
	- Random	0.4862	0.7666	0.2951	0.4367	0.5720	0.8218
	- Random-Seq	0.4518	0.6983	0.2560	0.4042	0.5326	0.7833
	- ASReP	0.4864	0.7681	0.2655	0.4055	0.5709	0.8220
	- DiffuASR (CG)	0.4994*	0.7710	**0.3022***	0.4415	0.5752	0.8251
	- DiffuASR (CF)	**0.4996***	**0.7727***	**0.3022***	**0.4454***	**0.5802***	**0.8294***

4.2 Overall Performance Comparison (RQ1)

To prove effectiveness and generality, we show the performance comparison between DiffuASR and competing baselines on three SRS models in Table 2. - *DiffuASR (CG)* and - *DiffuASR (CF)* represents the DiffuASR equipped with classifier guide and classifier free strategy, respectively. We analyze results in the following parts.

Overall, we find that DiffuASR (CG) and DiffuASR (CF) can consistently outperform competing augmentation methods on all three SRS models, illustrating that our DiffuASR can generate higher-quality augmentation. Then, observing the two baselines of random sampling, they underperform in most situations. Random-Seq samples items from the raw sequence, making the augmentation more similar to the original sequence. However, for long-tail users, this method causes a large amount of repetition of a few items, which makes the SRS models inclined to recommend these items repeatedly and thus degrades the performance. By comparison, Random augmentation generates noisy items as augmentation, but can elevate the performance slightly under some conditions, especially on more sparse dataset. We think the reason is that it can alleviate the problems of sparsity and long-tail users. Compared with DiffuASR, ASReP cannot bring more gains to SRS models, because it is limited by the inferior augmentation quality.

By analysis of the different datasets, we find that SRS models can get more promotion via augmentation methods on the Beauty dataset while few gains on the Steam dataset. The reason is that Beauty owns larger sparsity and shorter interaction sequences than Steam. This phenomenon illustrates that data sparsity and long-tail user problems degrade the SRS model exactly. Comparing the results among different SRS models, we find that Bert4Rec can only benefit from our DiffuASR, which indicates that Bert4Rec is more

prone to noisy augmentations. As for SASRec, ASReP can perform better than other baselines. We think the reason may be that the generation model has the same architecture as the recommendation model, *i.e.,* SASRec, which shares the same transitional patterns. In contrast, DiffuASR can outperform among various SRS models, which proves the generality of the proposed DiffuASR.

In summary, we can conclude that our DiffuASR is effective and general in promoting SRS models, which can be regarded as the response to research question **RQ1**.

4.3 Analysis of Different User Groups (RQ2)

As mentioned above, the long-tail user problem is one challenge for SRS models. To answer the **RQ2**, we show the recommending performance on different groups of users with various numbers of historical interactions in Figure 5. We divide the users into three groups based on their interaction number: *short* ($3 \leq n_u \leq 5$), *medium* ($5 < n_u \leq 20$) and *long* ($n_u > 20$). The experimental results show that Bert4Rec has better recommending performance in the short group than the other two SRS models, indicating that Bert4Rec is less prone to the long-tail user problem. However, we can find that Random and ASReP degrade Bert4Rec on all three groups, because it is more sensitive to noisy augmentation. By comparison, DiffuASR elevates the performance due to the high-quality augmentation. As for SASRec and S3Rec, all augmentation methods promote short groups, which shows that they alleviate the long-tail user problem. Nevertheless, they show a performance decrease in long user group for noise reason. In contrast, the proposed DiffuASR degrades long user group less and even brings a little increase. In conclusion, the DiffuASR can better alleviate the long-tail user problem consistently and avoid too much damage for experienced users.
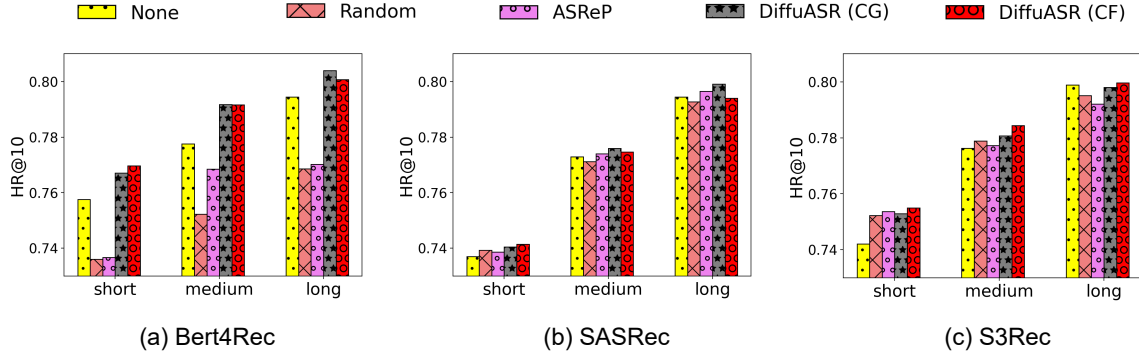
Figure 5: The performance on Yelp dataset with grouped users. "Short" denotes the length of user's interaction in the range of [3, 5], "Medium" in the range of (5, 20], "Long" in the range of (20, ∞).
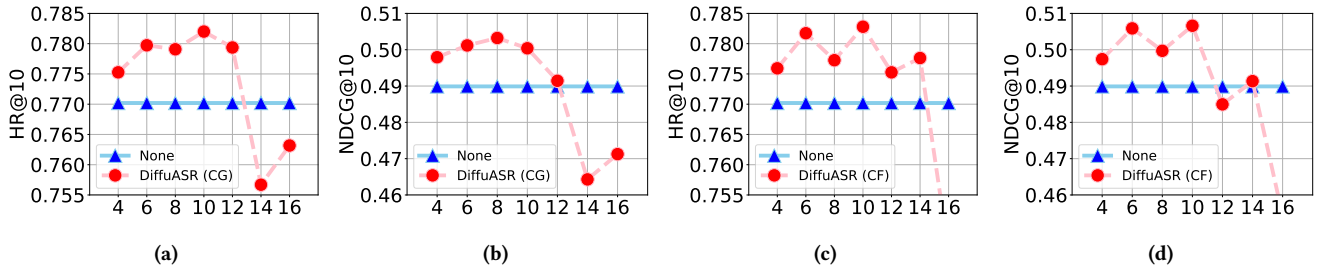

Figure 6: The results of experiments for the number of augmented items M for each user. All the results are conducted on Yelp dataset and for Bert4Rec model.
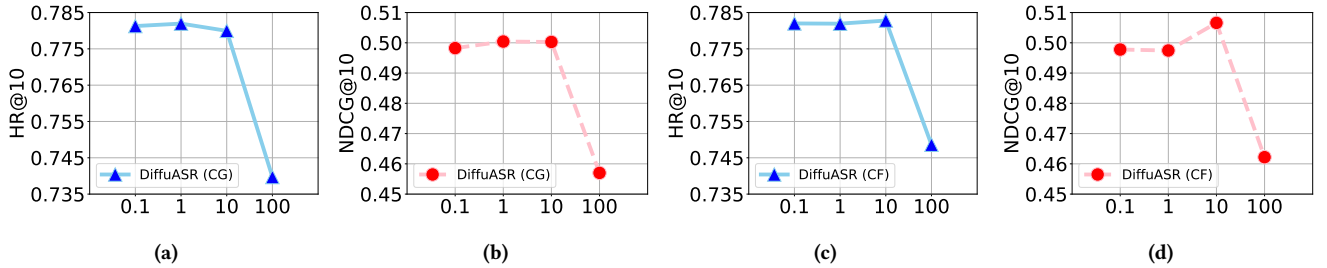

Figure 7: The results of experiments for hyper-parameter γ. All the results are conducted on Yelp dataset and for Bert4Rec.

4.4 Analysis of Augmented Number (RQ3)

The augmentation can enrich the interactions for the whole dataset and each user, so the number of augmented items indicates to what extent the DiffuASR alleviates sparsity and long-tail user problem. For the response to the **RQ3**, we show the performance changes of Bert4Rec according to the augment number M in Figure 6. The experimental results show that the recommending performance gets increased when the augment number M grows from 4 to 10, because more augment items can prolong the user's interaction sequence and reduce the sparsity of the whole dataset. However, we also find that if M continuously rises from 10 to a larger number, the recommending performance drops and is even worse than no augmentation. The reason is that with the growth of M, the data that can be used for training DiffuASR becomes less. $\mathcal{D}_T$ only contains the interaction sequence with length $n_u < M$, because the first M items are used for diffusion process in training process. With the decrease of training data for DiffuASR, the diffusion model easily falls into the underfitting problem, which leads to poor generation

quality. In conclusion, the augmented number M should take an appropriate value to maximize the benefits of data augmentation.

4.5 Hyper-parameter Analysis (RQ4)

As the response to **RQ4**, we explore the effects of two important hyper-parameters in DiffuASR, *i.e.,* schedule of β and guidance scale γ. As illustrated in [28], different schedules of β fit for various data distributions, so we aim to explore the most suitable schedule for our DiffuASR. In this experiment, we test the linear [11] (- *linear*), sqrt-root [20] (- *sqrt*), cosine [28] (- *cosine*) and sigmoid (- *sigmoid*) schedules, whose results are shown in Table 3. The experimental results indicate that the linear schedule performs the best for both classifier guide and classifier free variants. Therefore, we choose the linear schedule for β across all the experiments in this paper. Besides, Dhariwa *et al.* [5] and Ho *et al.* [12] illustrate that the guidance scale controls the trade-off between fidelity and diversity of generation. To explore the effects of γ, we show the performance trends with the change of γ in Figure 7. The experimental results

Table 3: The results on Yelp dataset and Bert4Rec model with different scheduler of β. The best results are in boldface.

Scheduler	DiffuASR (CG)		DiffuASR (CF)	
	NDCG@10	HR@10	NDCG@10	HR@10
- linear	**0.5004**	**0.7820**	**0.5066**	**0.7828**
- sqrt	0.4440	0.7304	0.4495	0.7345
- cosine	0.4510	0.7386	0.4779	0.7579
- sigmoid	0.4512	0.7410	0.4533	0.7391

show that when γ increases in small values, the recommending performance of DiffuASR gets elevation, because the generated items are more corresponding to the intent contained in the raw sequence. However, when the γ grows continuously, the performance of the SRS model drops drastically. We think it is because larger γ leads to augmented items more similar to items in raw sequence, which causes a similar problem as the Random-Seq baseline.

5 RELATED WORKS

Sequential Recommendation. There are two main branches in the recommendation research field. One line aims to make full use of side information of users and items, so these works focus on feature interaction [8, 30] and feature selection [24, 44, 45]. The other branch pays attention to modeling the user's historical records, *i.e.,* sequential recommendation, which predicts the next item based on the user's historical interactions. At the early stage, some traditional deep neural networks are adopted. For example, Hidasi *et al.* [10] and Tang *et al.* [38] propose to use RNN and CNN to extract preference from user's records, respectively. Recently, self-attention has shown the powerful ability for sequential modeling, which inspires researchers to adapt it to sequential recommendation, such as SASRec [14] and Bert4Rec [37]. Besides, some works [18, 19, 23, 55] begin to merge the two branches mentioned above, *i.e.,* combining side information with the sequential recommendation. Some other works explore to alleviate the problem of popularity bias [25, 48], efficiency [54] and noise [49, 50] for SRS. Despite the brilliant progress of SRS models, the data sparsity problem hinders further improvement. Two lines of works try to face this challenge, *i.e.,* data augmentation [3, 13, 26, 40, 46, 53] and contrastive learning [4, 29, 47]. We categorize data augmentation methods into two types: sequence-level and item-level. Sequence-level methods [3, 40, 46, 53] often enrich the data by generating new sequences, such as L2Aug [40] and CauseRec [53]. Though sequence-level methods can promote sequential recommendation models, they often need fabricated training procedures. Item-level methods [13, 26] augment pre-order items for each sequence. AS-ReP [26] firstly proposes to train a SASRec reversely as an augmentor and generate the pre-order items iteratively. Based on the augmentation of ASReP, BiCAT [13] further adds a forward constraint and self-knowledge distillation to enhance the robustness of the model. ASReP and BiCAT pay more attention to how to use augmented datasets, but ignore poor augment quality, which may hinder the benefits of augmentation. Therefore, to further improve the augment quality, we propose a diffusion augmentation method.

Diffusion Model. The diffusion model has attracted much attention recently, because of its superior performance on image generation. DPM [35] and DDPM [11] propose the general process of diffusion model. Since the reverse process is not parallel, the diffusion model suffers from low sampling speed. To accelerate the generation process, DDIM [36] proposes to transform the forward process to a non-markovian process and thus can reduce the steps for the reverse process. Besides, control of generation is another vital issue. Dhariwal *et al.* [5] and Ho *et al.* [12] devise a classifier guide and classifier free diffusion model to control the generation corresponding to the desired class. ControlNet [52] proposes a novel finetuning strategy to add an additional control signal to a well-trained diffusion model. However, there are still few works to explore the applications of diffusion model in recommendation. Three recent works [6, 22, 43] take the first step. Wang *et al.* [43] adapts the diffusion model to generative recommender system. DiffuRec [22] and DiffRec [6] utilize the diffusion model for the sequential recommendation, but they generate the target item directly, which differs from the usage of DiffuASR. We use the generative ability of the diffusion model for data augmentation, which explores a novel way to adopt the diffusion model in the recommendation field.

6 CONCLUSION

In this paper, we propose a diffusion augmentation for sequential recommendation (DiffuASR). To adapt the diffusion model to the item sequence generation, we first derive the framework of DiffuASR. Then, we design a Sequential U-Net to capture the sequence information while predicting the added noise. Besides, two guide strategies are designed to control the DiffuASR to generate the items more corresponding to the preference contained in the raw sequence. We conduct extensive experiments on three real-world datasets combined with three sequential recommendation models. The experimental results show the effectiveness and generality of our DiffuASR. Besides, we find that the number of augmented items affects performance largely in the experiments.

REFERENCES

[1] 2020. MindSpore. https://www.mindspore.cn/

[2] Huiyuan Chen, Yusan Lin, Menghai Pan, Lan Wang, Chin-Chia Michael Yeh, Xiaoting Li, Yan Zheng, Fei Wang, and Hao Yang. 2022. Denoising self-attentive sequential recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*. 92–101.

[3] Xu Chen, Zhenlei Wang, Hongteng Xu, Jingsen Zhang, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Data Augmented Sequential Recommendation based on Counterfactual Thinking. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[4] Yongjun Chen, Zhiwei Liu, Jia Li, Julian McAuley, and Caiming Xiong. 2022. Intent contrastive learning for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*. 2172–2182.

[5] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 8780–8794.

[6] Hanwen Du, Huanhuan Yuan, Zhen Huang, Pengpeng Zhao, and Xiaofang Zhou. 2023. Sequential Recommendation with Diffusion Models. *arXiv preprint arXiv:2304.04541* (2023).

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

[8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[9] Zhengfu He, Tianxiang Sun, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. 2022. DiffusionBERT: Improving Generative Masked Language Models with Diffusion Models. *arXiv preprint arXiv:2211.15029* (2022).

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[11] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems* 33 (2020), 6840–6851.

[12] Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).

[13] Juyong Jiang, Yingtao Luo, Jae Boum Kim, Kai Zhang, and Sunghun Kim. 2021. Sequential recommendation with bidirectional chronological augmentation of transformer. *arXiv preprint arXiv:2112.06460* (2021).

[14] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[15] Kibum Kim, Dongmin Hyun, Sukwon Yun, and Chanyoung Park. 2023. MELT: Mutual Enhancement of Long-Tailed User and Item for Sequential Recommendation. *arXiv preprint arXiv:2304.08382* (2023).

[16] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[17] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[18] Muyang Li, Zijian Zhang, Xiangyu Zhao, Wanyu Wang, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2023. AutoMLP: Automated MLP for Sequential Recommendations. In *Proceedings of the ACM Web Conference 2023*. 1190–1198.

[19] Muyang Li, Xiangyu Zhao, Chuan Lyu, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2022. MLP4Rec: A Pure MLP Architecture for Sequential Recommendations. In *31st International Joint Conference on Artificial Intelligence and the 25th European Conference on Artificial Intelligence (IJCAI-ECAI 2022)*. International Joint Conferences on Artificial Intelligence, 2138–2144.

[20] Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems* 35 (2022), 4328–4343.

[21] Yifan Li, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Diffusion Models for Non-autoregressive Text Generation: A Survey. *arXiv preprint arXiv:2303.06574* (2023).

[22] Zihao Li, Aixin Sun, and Chenliang Li. 2023. DiffuRec: A Diffusion Model for Sequential Recommendation. *arXiv preprint arXiv:2304.00686* (2023).

[23] Jiahao Liang, Xiangyu Zhao, Muyang Li, Zijian Zhang, Wanyu Wang, Haochen Liu, and Zitao Liu. 2023. MMMLP: Multi-modal Multilayer Perceptron for Sequential Recommendations. In *Proceedings of the ACM Web Conference 2023*. 1109–1117.

[24] Weilin Lin, Xiangyu Zhao, Yejing Wang, Tong Xu, and Xian Wu. 2022. AdaFS: Adaptive feature selection in deep recommender system. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 3309–3317.

[25] Qidong Liu, Feng Tian, Qinghua Zheng, and Qianying Wang. 2023. Disentangling interest and conformity for eliminating popularity bias in session-based recommendation. *Knowledge and Information Systems* 65, 6 (2023), 2645–2664.

[26] Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S Yu. 2021. Augmenting sequential recommendation with pseudo-prior items via reversely pre-training transformer. In *Proceedings of the 44th international ACM SIGIR conference on Research and development in information retrieval*. 1608–1612.

[27] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 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.

[28] Alexander Quinn Nichol and Prafulla Dhariwal. 2021. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*. PMLR, 8162–8171.

[29] Xiuyuan Qin, Huanhuan Yuan, Pengpeng Zhao, Junhua Fang, Fuzhen Zhuang, Guanfeng Liu, Yanchi Liu, and Victor Sheng. 2023. Meta-optimized Contrastive Learning for Sequential Recommendation. *arXiv preprint arXiv:2304.07763* (2023).

[30] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[31] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684–10695.

[32] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 234–241.

[33] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems* 35 (2022), 36479–36494.

[34] Uriel Singer, Haggai Roitman, Yotam Eshel, Alexander Nus, Ido Guy, Or Levi, Idan Hasson, and Eliyahu Kiperwasser. 2022. Sequential modeling with multiple attributes for watchlist recommendation in e-commerce. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 937–946.

[35] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*. PMLR, 2256–2265.

[36] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

[37] 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 *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

[38] Jiaxi Tang and Ke Wang. 2018. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 565–573.

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[40] Jianling Wang, Ya Le, Bo Chang, Yuyan Wang, Ed H Chi, and Minmin Chen. 2022. Learning to Augment for Casual User Recommendation. In *Proceedings of the ACM Web Conference 2022*. 2183–2194.

[41] Jianling Wang, Raphael Louca, Diane Hu, Caitlin Cellier, James Caverlee, and Liangjie Hong. 2020. Time to Shop for Valentine's Day: Shopping Occasions and Sequential Recommendation in E-commerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 645–653.

[42] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z Sheng, and Mehmet Orgun. 2019. Sequential recommender systems: challenges, progress and prospects. *arXiv preprint arXiv:2001.04830* (2019).

[43] Wenjie Wang, Yiyan Xu, Fuli Feng, Xinyu Lin, Xiangnan He, and Tat-Seng Chua. 2023. Diffusion Recommender Model. *arXiv preprint arXiv:2304.04971* (2023).

[44] Yejing Wang, Zhaocheng Du, Xiangyu Zhao, Bo Chen, Huifeng Guo, Ruiming Tang, and Zhenhua Dong. 2023. Single-shot Feature Selection for Multi-task Recommendations. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 341–351.

[45] Yejing Wang, Xiangyu Zhao, Tong Xu, and Xian Wu. 2022. Autofield: Automating feature selection in deep recommender systems. In *Proceedings of the ACM Web Conference 2022*. 1977–1986.

[46] Zhenlei Wang, Jingsen Zhang, Hongteng Xu, Xu Chen, Yongfeng Zhang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Counterfactual data-augmented sequential recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 347–356.

[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 *2022 IEEE 38th international conference on data engineering (ICDE)*. IEEE, 1259–1273.

[48] Yuhao Yang, Chao Huang, Lianghao Xia, Chunzhen Huang, Da Luo, and Kangyi Lin. 2023. Debiased Contrastive Learning for Sequential Recommendation. In *Proceedings of the ACM Web Conference 2023*. 1063–1073.

[49] Chi Zhang, Rui Chen, Xiangyu Zhao, Qilong Han, and Li Li. 2023. Denoising and Prompt-Tuning for Multi-Behavior Recommendation. In *Proceedings of the ACM Web Conference 2023*. 1355–1363.

[50] Chi Zhang, Yantong Du, Xiangyu Zhao, Qilong Han, Rui Chen, and Li Li. 2022. Hierarchical Item Inconsistency Signal Learning for Sequence Denoising in Sequential Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2508–2518.

[51] Jing Zhang, Bowen Hao, Bo Chen, Cuiping Li, Hong Chen, and Jimeng Sun. 2019. Hierarchical reinforcement learning for course recommendation in MOOCs. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 435–442.

[52] Lvmin Zhang and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. *arXiv preprint arXiv:2302.05543* (2023).

[53] Shengyu Zhang, Dong Yao, Zhou Zhao, Tat-Seng Chua, and Fei Wu. 2021. Causerec: Counterfactual user sequence synthesis for sequential recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research*

and Development in Information Retrieval. 367–377.

[54] Kesen Zhao, Xiangyu Zhao, Zijian Zhang, and Muyang Li. 2022. MAE4Rec: Storage-saving Transformer for Sequential Recommendations. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 2681–2690.

[55] 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 *Proceedings of the 29th ACM international conference on information & knowledge management*. 1893–1902.