Towards More Robust and Accurate Sequential Recommendation with Cascade-guided Adversarial Training

Juntao Tan*† Shelby Heinecke ‡ Zhiwei Liu ‡ Yongjun Chen ‡ Yongfeng Zhang* Huan Wang‡

Abstract

Sequential recommendation models, models that learn from chronological user-item interactions, outperform traditional recommendation models in many settings. Despite the success of sequential recommendation, their robustness has recently come into question. Two properties unique to the nature of sequential recommendation models may impair their robustness - the cascade effects induced during training and the model's tendency to rely too heavily on temporal information. To address these vulnerabilities, we propose Cascade-guided Adversarial training, a new adversarial training procedure that is specifically designed for sequential recommendation models. Our approach harnesses the intrinsic cascade effects present in sequential modeling to produce strategic adversarial perturbations to item embeddings during training. Experiments on training state-of-the-art sequential models on four public datasets from different domains show that our training approach produces superior model ranking accuracy and superior model robustness to real item replacement perturbations when compared to both standard model training and generic adversarial training.

Key words. Robust Recommendation; Adversarial Training; Sequential Recommendation

1 Introduction

Sequential recommender models learn dynamic user preferences and recommend next items by modeling past user behaviors in sequential order. Recently, deep learning based models have gained attention. These models learn user preferences by feeding the user histories into deep neural networks, such as RNN [12, 30, 11], CNN [28, 32], or transformer [13, 26, 6, 16].

Despite their outstanding performance, recent works reveal their unique robustness issues. A small change at the end of the user sequence, sometimes even on only one item, can dramatically change model behavior [34, 33]. In real-world applications, this kind of instability can lead to serious user dissatisfaction. For example, if a user accidentally clicks on an unwanted video and then returns to the previous page, they might find that all the recommended videos have changed due to that single misclick. In general, robustness is crucial for recommender systems as it affects user trust and acceptance [22, 1, 17]. Meanwhile, robustness and accuracy are sometimes correlated [22], and training a more stable recommender system may also lead to more accurate predictions. Previous work in robustness and stability in recommender systems focus on model families such as matrix factorization [10, 24]. Enhancing robustness and accuracy in complex models like sequential recommendation remains relatively untouched.

Sequential recommendation models possess two unique properties that may hinder their robustness: (1) Though they use temporal information to learn user preferences, they tend to **overemphasize** recent interactions, making the model's recommendations highly sensitive to them [34, 33]. (2) Due to their time-aware training, these models inherently exhibit *cascade effects*, which refers to the fact that changes in a user's behavior can indirectly affect recommendations for others through collaborative filtering [21, 4]. In sequential recommender models, item interactions in early timestamps generally induce larger cascade effects, since they affect the predictions on every subsequent interaction in each training epoch [21]. During training, their gradients are computed more frequently and are more diverse. As a result, while trained models might be robust against perturbations on the interactions with higher cascade values (i.e., the "early interactions"), they are less robust to perturbations on later ones. Paradoxically, even though the last few user interactions greatly influence predictions, they are underrepresented during training. The combination of these properties exacerbates the vulnerability to perturbations at the end of the user history sequences.

In this study, we enhance robustness by modifying the training process. Noting that interactions with lower cascade effects often play a pivotal role in predictions, we boost stability for these interactions through adding more noise on them during training via adversarial training. Adversarial training has been shown to improve the robustness of a variety of deep learning models by adding adversarial perturbation on the input data [29]. Originally proposed for image classification, adversarial training produces image classification

*Rutgers University.
{juntao.tan, yongfeng.zhang}@rutgers.edu
†This project was completed while participating in an internship at Salesforce Research.
‡Salesforce Research. {shelby.heinecke, zhiweiliu, yongjun.chen, huan.wang}@salesforce.com

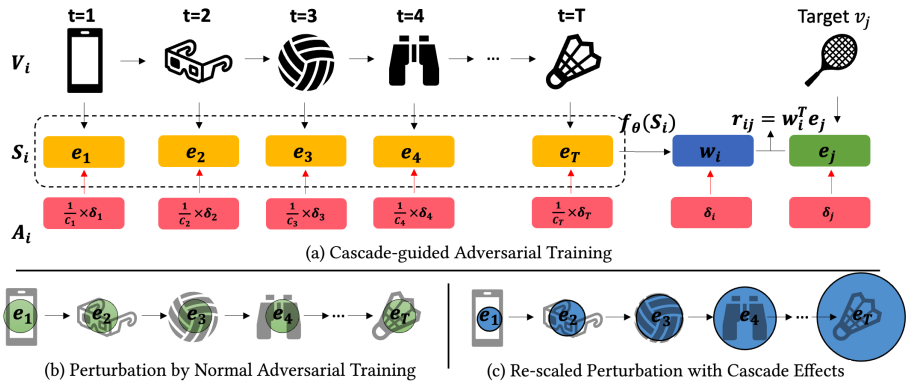


Figure 1: A toy example of applying adversarial training on sequential recommendation. (a) How adversarial perturbations are applied on the learned item and user embeddings. (b) Generic adversarial training applies adversarial perturbations of the same magnitude (green circles) to all item embeddings. (c) The Cascade-guided adversarial training method dynamically choose the magnitude of the perturbations (blue circles) according to the different cascade effects of each interaction in the user history.

models that are more robust to adversarial attack on the input pixels [18]. In the sequential recommendation setting, we expect that by adding greater adversarial perturbations to the interactions with lower cascade effects, the trained model will be more robust to perturbations occurring among the most recent items in the user sequences when making predictions. Based on this, we propose a Cascade-guided Adversarial Training strategy, which is a re-normalized adversarial training method that is specifically designed for deep sequential recommendation by harnessing the intrinsic cascade effects for each interaction.

Our method distinguishes itself from previous research in two main aspects: (1) While prior studies focus on adversarial training for early recommendation models like matrix factorization and collaborative filtering [5], we explore its application specifically to deep sequential recommendation models, leveraging their unique characteristics; (2) Unless previous works focus on improving generalization or robustness to latent embedding perturbations, instead, we emphasize robustness against adversarial perturbations on real inputs, namely, item interaction sequences, which are more likely perturbations in practical scenarios.

Our work makes the following contributions:

- We propose a novel adversarial training algorithm that uses the cascade effects to construct strategic adversarial examples during training.
- Through rigorous experiments on four datasets using two widely used sequential recommendation models, we demonstrate our approach outperforms standard and generic adversarial training in terms of accuracy, generalization, and robustness against realistic item replacement perturbations.

2 Related Works

2.1 Adversarial Training. Adversarial training framework emerged from the seminal paper [27]. It demonstrates that images with human-imperceptible perturbations, *i.e.* adversarial examples, could be misclassified by well-trained computer vision classifiers. Thus, those perturbations also need to be tackled during model training phase, which aims to enhance the robustness of models against such perturbations [29]. Key advancements encompass the Fast Gradient Sign Attack (FGSA) [8] and the min-max optimization formulation [18].

Adversarial training has recently extended to fields like NLP, using word embedding perturbations [20], and recommendation, with user and item embedding perturbations [10]. However, in recommendation systems, users typically access only item interactions, not model internals, such as the embeddings. We contend that adversarial threats might more authentically come from item misclick-induced interaction perturbations than from direct embedding tweaks. To the best of our knowledge, adversarial training has yet to be explored as a method to enhance robustness against perturbations on model externals.

Existing adversarial training methods for recommendation systems target model types like matrix factorization, collaborative filtering, and others, as reviewed in [5]. While one study has attempted adversarial training for sequential recommendation [19], none have crafted a general approach that capitalizes on the unique characteristics of these models as we do.

2.2 Robustness of Recommendation Robust recommender systems can be categorized into three primary definitions. First, the most prevalent definition

centers on the system's accuracy amidst noisy data [22, 7, 25], given that users may inadvertently provide inaccurate ratings. Here, robustness gauges the accuracy shift relative to data noise. Second, robustness hinges on the stability of the recommendations [1, 2, 3, 17]. Stability in recommendations is evaluated by checking model predictions on unknown items after integrating its own predictions into the training data. A robust model remains largely unchanged under these conditions. Thirdly, robustness is viewed through an attack-defense lens [5, 33]. Here, malicious parties might launch attacks on models, for instance, using fake user profiles. Such attacks aim to either push specific items or sabotage the system. Robustness thus signifies the system's ability to fend off these threats. For insights on attack-defense tactics and further robustness aspects in recommendations, refer to [5] and [23] respectively.

The key goal of our work is to train robust sequential recommendation models with respect to the first definition of robustness, where our noise model is an adversarial item replacement scheme. We motivate the adversarial item replacement by first showing that replacing later items in interaction histories has a stronger negative impact to ranking accuracy than replacing items earlier in interaction histories. We then develop our novel adversarial training algorithm and show that it enhances resilience to this type of noise and it enhances generalization on unperturbed data.

3 Preliminary

In this section, we briefly introduce the basic ideas of adversarial training and sequential recommendation.

3.1 Adversarial Training Adversarial training is defined as a robust optimization problem with saddle point (min-max) formulation [18]. Consider a general classification problem with d-dimensional input data $x \in \mathbb{R}^d$ and corresponding label $y \in \mathbb{Z}$ under data distribution $\mathcal{D}$. For a classification model f_θ, the training goal is to minimize the risk $\mathbb{E}_{x,y\sim\mathcal{D}}[L(f_\theta(x), y)]$. Adversarial training aims to find a perturbation δ with bounded norm $||\delta|| < \epsilon$ that maximizes the minimum risk with respect to θ. This can be summarized as the following min-max equation:

$$\min_\theta \left(\max_{\delta, ||\delta|| < \epsilon} \mathbb{E}_{x,y\sim\mathcal{D}}[L(f_\theta(x+\delta), y)] \right) \tag{3.1}$$

Fast gradient sign method (FGSM) [8] is the most commonly used method that solves for δ. FGSM simply generates adversarial perturbations by multiplying the sign of the gradient of the loss function by the maximal perturbation magnitude, ϵ:

$$\delta = \epsilon \cdot sign\big(\nabla L(f_\theta(x+\delta), y)\big) \tag{3.2}$$

3.2 Sequential Recommendation Our work focuses on adversarial training for sequential recommendation systems. Sequential recommendation systems learn from the ordering of historical user-item interactions to predict the next user-item interaction. We formalize the key components as follows. Let $i \in [1, m]$ denote the index of a user and and $\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$ denote the set of all possible items. Suppose user i has a sequentially ordered interaction history of length T, $\mathcal{V}_i = \{v_i^t \mid t = 1, \cdots, T\}$. In practice, each user has different length of history. T is a hyper parameter that decides the maximum length of user history, such that only the last T interactions is considered when making predictions. A sequential recommendation model learns the embedding for each item v_i denoted as $\boldsymbol{e_i}$. Together, these embeddings form the item embedding matrix, $\boldsymbol{E} \in \mathbb{R}^{n\times d}$. For each user i, we concatenate the sequence of embeddings of items in $\mathcal{V}_i$, denoted as $\boldsymbol{S_i} = [\boldsymbol{e}_i^t \mid t = 1, \cdots, T]$. The sequential models learn the user embedding, $\boldsymbol{w}_i$, as a function of the sequence embeddings,

$$\boldsymbol{w_i} = f(\boldsymbol{S_i}; \theta), \tag{3.3}$$

where f denotes a sequence embedding model, such as Transformer, and θ denotes the model parameters. After learning $\boldsymbol{w}_i$, for a target item v_j, the ranking score $r_{i,j}$ is predicted by

$$r_{i,j} = \boldsymbol{w_i}^T \boldsymbol{e_j}. \tag{3.4}$$

During model training, for each user i with target item v_j and a set of negative samples $\mathcal{N}^- \subset \mathcal{V} \setminus v_j$, we minimize the binary cross entropy (BCE) loss defined as:

$$L_B(i, j, \mathcal{N}^-; \theta, \boldsymbol{E}) = -\big(\log(\sigma(r_{i,j})) + \sum_{v_n \in \mathcal{N}^-} \log(1 - \sigma(r_{i,n}))\big) \tag{3.5}$$

During training, following the same setting as in [13], we truncate the user sequence $\mathcal{V}_i$ according to each timestamp t. For each sub-sequence, item v_i^t is treated as the target item and the ranking score is predicted by taking previous items as dynamic user history. Meanwhile, only one negative item v_n is sampled for each v_i^t in each sub-sequence. For simplicity, in the rest of the paper, we denote the ranking loss for item v_j to user i with negative sample v_n as $L_{\mathrm{B}}(i, j, n)$.

4 Method

We first introduce the proposed adversarial training algorithm that considers the cascade effects in sequential recommendation. Then, we propose an algorithm to effectively compute the cascade effects for each interaction in the user histories.

4.1 Cascade-guided Adversarial Training Since recommender systems take discrete input data (i.e., user/item IDs), adversarial perturbations are commonly applied on the latent embeddings. As shown in Figure 1 (a), a deep sequential recommendation model is a hierarchy consisting of two levels: (1) a non-linear deep neural network that takes the sequence embeddings of the user's history $\boldsymbol{S_i}$ as input and outputs the user embedding $\boldsymbol{w_i}$; (2) a linear model that predicts the final ranking score for user i on target item v_j by linearly multiplying their embeddings as $\boldsymbol{w_i^T e_j}$. We apply adversarial training on both levels separately.

For the first level, the adversarial training objective is described as follows. When perturbing the item embeddings in the history sequence within a certain small magnitude, even in the worst case, the learned user embedding shouldn't be too different from the original prediction. Suppose for user i with sequence embeddings $\boldsymbol{S_i}$, we denote the adversarial perturbations on the sequence embeddings as $\boldsymbol{A_i} = [\boldsymbol{\delta_i^t} \mid t = 1, \cdots, T]$, where $\boldsymbol{\delta_i^t}$ is the perturbation applied on the corresponding item embedding $\boldsymbol{e_i^t}$. We mathematically formulate this objective by an adversarial loss function:

$$
\begin{aligned}
&L_{\text{adv-1}}(i, \boldsymbol{A_i}) = ||\hat{\boldsymbol{w}}_{\boldsymbol{i}} - \boldsymbol{w_i}||_2 \\
&\text{where} \quad \boldsymbol{w_i} = f(\boldsymbol{S_i}; \theta), \quad \hat{\boldsymbol{w}}_{\boldsymbol{i}} = f\big(\boldsymbol{S_i} + \frac{1}{\boldsymbol{C_i}} \odot \boldsymbol{A_i}; \theta\big)
\end{aligned} \tag{4.6}
$$

In Eq (4.6), the vector $\boldsymbol{C_i} \in [1, +\infty)^T$ denotes the cascade effects of each interaction in user i's history. Higher values of $\boldsymbol{C_i}$ denote higher cascade effects. The way to calculate these cascade effects will be introduced in the next section. The factor $\frac{1}{\boldsymbol{C_i}}$ plays a crucial rule in our method: it re-scales the adversarial perturbations so that interactions with smaller cascade effects will receive a larger adversarial perturbation. During sequential recommendation model training, interactions with larger cascade effects are used more often in training than interactions with smaller cascade effects [21], hence, the latter can be more vulnerable and unstable (justified later in Section 5.6). By applying larger perturbations on the interactions with lower cascade effects, we obtain a model that is more equally robust across all sequence embeddings.

To approximate the worst case adversarial perturbation $\boldsymbol{A_i}$, we apply the FGSM attack introduced in Eq.(3.2):

$$
\mathcal{A}_i = \epsilon \frac{\boldsymbol{g}}{||\boldsymbol{g}||_2} \text{ where } \boldsymbol{g} = \frac{\partial L_{\text{adv-1}}(i, \boldsymbol{A_i})}{\partial \boldsymbol{S}_i} \tag{4.7}
$$

We note that $\frac{\boldsymbol{g}}{||\boldsymbol{g}||_2}$ is the sign of the direction of the applied perturbation and ϵ is a human defined parameter to determine the general magnitude of the adversarial attack. The specific magnitude of the perturbation on each interaction will be re-scaled by their cascade effects as one of the key features of our method.

The adversarial training on the first level guarantees the aggregated user embedding $\boldsymbol{w_i}$ is robust to small perturbations on the user history. On the second level, we apply adversarial training on the learned user embedding and the target item embeddings. This assures that when small changes are applied to the user and target item embeddings, the model is still able to generate highly accurate recommendation results. We use BCE loss as the second adversarial training loss, which is exactly the same loss function used to train the base recommendation model. Suppose $\boldsymbol{\delta_i}$, $\boldsymbol{\delta_j}$, $\boldsymbol{\delta_n}$ are the adversarial perturbations applied on the embeddings of the user i, target item v_j, and negative sampled item v_n, respectively. The second adversarial training loss is defined as:

$$
\begin{aligned}
&L_{\text{adv-2}}(i, j, n, \boldsymbol{\delta_i}, \boldsymbol{\delta_j}, \boldsymbol{\delta_n}) = -\big(\log(\sigma(\hat{r}_{i,j})) + \log(1 - \sigma(\hat{r}_{i,n}))\big) \\
&\text{where} \quad \hat{r}_{i,j} = (\boldsymbol{w_i^T} + \boldsymbol{\delta_i})(\boldsymbol{e}_j + \boldsymbol{\delta_j}), \\
&\qquad\quad \hat{r}_{i,n} = (\boldsymbol{w_i^T} + \boldsymbol{\delta_i})(\boldsymbol{e}_n + \boldsymbol{\delta_n})
\end{aligned} \tag{4.8}
$$

Here $\hat{r}_{i,k}$ is the predicted ranking score for any user i and item k after perturbation. Similarly, we approximately generate the worst case $\boldsymbol{\delta_i}$, $\boldsymbol{\delta_j}$, and $\boldsymbol{\delta_n}$ within maximum magnitude ϵ by:

$$
\begin{aligned}
\boldsymbol{\delta_i} &= \epsilon \frac{\boldsymbol{h_i}}{||\boldsymbol{h_i}||_2} \text{ where } \boldsymbol{h_i} = \frac{\partial L_{\text{adv-2}}(i, j, n, \boldsymbol{\delta_i}, \boldsymbol{\delta_j}, \boldsymbol{\delta_n})}{\partial \boldsymbol{e_i}} \\
\boldsymbol{\delta_j} &= \epsilon \frac{\boldsymbol{h_j}}{||\boldsymbol{h_j}||_2} \text{ where } \boldsymbol{h_j} = \frac{\partial L_{\text{adv-2}}(i, j, n, \boldsymbol{\delta_i}, \boldsymbol{\delta_j}, \boldsymbol{\delta_n})}{\partial \boldsymbol{e_j}} \\
\boldsymbol{\delta_n} &= \epsilon \frac{\boldsymbol{h_n}}{||\boldsymbol{h_n}||_2} \text{ where } \boldsymbol{h_n} = \frac{\partial L_{\text{adv-2}}(i, j, n, \boldsymbol{\delta_i}, \boldsymbol{\delta_j}, \boldsymbol{\delta_n})}{\partial \boldsymbol{e_n}}
\end{aligned} \tag{4.9}
$$

The two adversarial training objectives synergistically improve the robustness of the trained model with respect to all the components. Finally, we optimize the model parameters by minimizing the sum of original BCE loss and the two adversarial training losses:

$$
\begin{aligned}
\min_{\theta, \boldsymbol{E}} L = &L_{\text{B}}(i, j, n) + \lambda_1 L_{\text{adv-1}}(i, \boldsymbol{A_i}) \\
&+ \lambda_2 L_{\text{adv-2}}(i, j, n, \boldsymbol{\delta_i}, \boldsymbol{\delta_j}, \boldsymbol{\delta_n})
\end{aligned} \tag{4.10}
$$

It's worth noting that since the cascade effects only affect the model after the general training process, the adversarial training should be applied after the training of the base model. In the adversarial training phase, by minimizing Eq. (4.10), we expect to learn better item embeddings $\boldsymbol{E}$ and the model parameters θ such that the model is more accurate and robust.

4.2 Cascade Effects Calculation In this section, we will introduce how we calculate the cascade effect

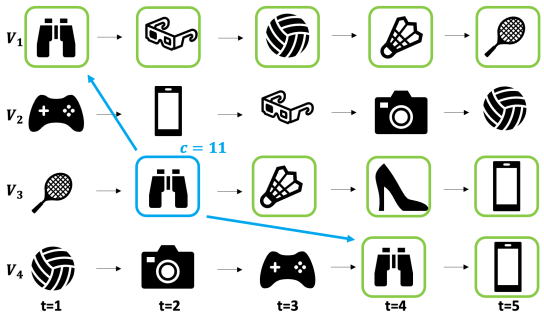


Figure 2: An example of calculating cascade effects. The item with blue bounding box has cascade effects on the 10 items with green bounding boxes plus itself. Its cascade value is 11.

matrix $\boldsymbol{C} \in [0,1]^{m\times T}$ for all the interactions in the user histories, in a time efficient manner.

When training a sequential recommendation model, each user-item interaction produces cascade effects. For a given user-item interaction, two types of interactions receive its cascade effects: (1) all interactions following the given interaction, within the same user history sequence; (2) all interactions with the same item occurring in different user history sequences within the same training batch. This is illustrated in Figure 2.

Based on the above observation, for an item v_t^i for which user i interacted at timestamp t, we define the cascade effect $C(i,t)$ as:

$$C(i,t) = 1+T-t+\frac{b}{m}\sum_{k\in\mathcal{U},k\neq i}\sum_{l\leq T}(1+T-l)\mathbb{I}(v_i^t, v_k^l) \tag{4.11}$$

$$\mathbb{I}(v_i^t, v_k^l) = \begin{cases} 1, & \text{if } v_i^t = v_k^l \\ 0, & \text{otherwise} \end{cases} \tag{4.12}$$

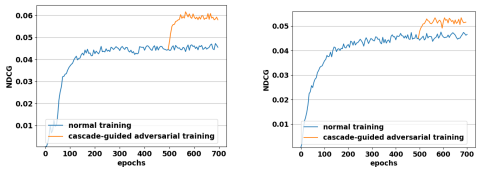
Here, b is the batch size during training and $\frac{b}{m}$ approximates the probability of two user sequences appearing in the same training batch. After calculating $\boldsymbol{C}$, its inverse will be a real number in (0, 1], and this will be used to re-normalize the magnitudes of adversarial perturbations.

We note that in Eq. (4.11), $1+T-t$ calculates the cascade effects that directly come from the temporal information in the **same** user history, i.e., the inverse of timestamp. The accumulative term calculates the cascade effects that comes from the same item in **different** user sequences. In Section 5.5, we do ablation study to show how much of each type of cascade effects contributes to the final performance.

See supplementary material for a more detailed algorithm for calculating cascade scores.

Table 1: Statistics of the datasets.

Dataset	#User	#Item	#Interaction	Density
ML-1M	6,040	3,416	987,540	4.786%
Beauty	22,363	10,121	198,502	0.088%
Video	24,303	10,672	231,780	0.089%
Clothing	39,387	23,033	278,677	0.031%



(a) Train SASRec on Video. (b) Train GRU4Rec on Video.

Figure 3: NDCG vs. the number of training epochs.

5 Experiments

We conduct experiments to evaluate the generalization and robustness of deep sequential models trained under our proposed method. Specifically, we consider the following questions:

- **EXP1:** Does the proposed method improve the ranking accuracy of deep sequential recommendation models? How does it compare to normal adversarial training?
- **EXP2:** Does the proposed method improve the robustness of the trained models? If so, which aspect of robustness does it improve, and how much is the improvement?

We first introduce the datasets, base recommendation models, implementation details, and evaluation metrics used in the experiments. Then, in Section 5.5, we discuss the experiments on ranking accuracy. In Section 5.6, we address the experiments on robustness. The time efficiency of the proposed adversarial training algorithm is analyzed in the supplementary materials.

5.1 Datasets We conduct experiments on four diverse public datasets spanning various domains and densities, which are MovieLens-1M * [9] and Video, Beauty, Clothing from Amazon review datasets. They are widely used datasets in recommendation research. Dataset statistics are shown in Table 1.

5.2 Base Models Since RNN and transformer are the most common structures been used in deep sequential models [14], we choose two of the most representative recommendation models from each of the above categories as the base models:

*https://files.grouplens.org/datasets/movielens

Table 2: Improvement of the accuracy by applying different adversarial training methods on the base models.

		MovieLens-1M				Beauty			
		NDCG@10	vs. Base	HT@10	vs. Base	NDCG@10	vs. Base	HT@10	vs. Base
SASRec	base	0.1359	-	0.2528	-	0.0264	-	0.0537	-
	adv_linear[10]	0.1508	10.96%	0.2795	10.56%	0.0315	19.32%	0.0626	16.57%
	adv_seq[20, 19]	0.1440	5.96%	0.2680	6.01%	0.0304	15.15%	0.0597	11.17%
	adv_global	0.1519	11.77%	0.2808	11.08%	0.0313	18.56%	0.0622	15.83%
	adv_cas	**0.1546**	**13.76%**	**0.2831**	**11.99%**	**0.0320**	**21.21%**	**0.0630**	**17.32%**
GRU4Rec	base	0.1255	-	0.2419	-	0.0228	-	0.0460	-
	adv_linear[10]	0.1351	7.65%	0.2581	6.70%	0.0288	26.32%	0.0560	21.74%
	adv_seq[20, 19]	0.1308	4.22%	0.2520	4.18%	0.0252	10.53%	0.0493	7.17%
	adv_global	0.1345	7.17%	0.2543	5.13%	0.0285	25.00%	0.0550	19.57%
	adv_cas	**0.1360**	**8.37%**	**0.2588**	**6.99%**	**0.0298**	**30.70%**	**0.0566**	**23.04%**
		Video				Clothing			
		NDCG@10	vs. Base	HT@10	vs. Base	NDCG@10	vs. Base	HT@10	vs. Base
SASRec	base	0.0441	-	0.0875	-	0.0088	-	0.0184	-
	adv_linear[10]	0.0553	25.40%	0.1059	21.03%	0.0074	-15.91%	0.0154	-16.30%
	adv_seq[20, 19]	0.0519	17.69%	0.1001	14.40%	0.0106	20.45%	0.0213	15.76%
	adv_global	0.0557	26.30%	0.1068	22.06%	0.0103	17.05%	0.0216	17.39%
	adv_cas	**0.0606**	**37.41%**	**0.1162**	**32.80%**	**0.0107**	**21.59%**	**0.0219**	**19.02%**
GRU4Rec	base	0.0442	-	0.0872	-	0.0072	-	0.0150	-
	adv_linear[10]	0.0500	13.12%	0.0962	10.32%	0.0070	-2.78%	0.0154	2.67%
	adv_seq[20, 19]	0.0456	3.17%	0.0888	1.49%	0.0074	2.78%	0.0152	1.33%
	adv_global	0.0496	12.22%	0.0966	10.78%	0.0071	-1.39%	0.0146	-2.67%
	adv_cas	**0.0520**	**17.65%**	**0.0993**	**13.88%**	**0.0083**	**15.28%**	**0.0168**	**12.00%**

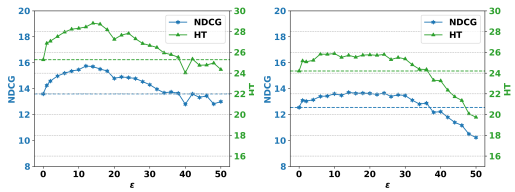
- GRU4Rec[12]: GRU4Rec utilizes RNNs to learn user preferences based on their history sequences.

- SASRec[13]: SASRec relies on attention mechanisms that can dynamically learn the attention weights on each interaction in the user sequences.

5.3 Implementation Details First, we implement exactly the same architectures for the two base models as described in their original papers. We use a single layer of GRU units in the GRU4Rec model and 2 self-attention blocks in the SASRec model. The hidden size is set to 100 for both models. When computing user embeddings, the maximum sequence length T is set to 200 for MovieLens-1M and 50 for Video, Beauty and Clothing datasets according to their different densities.

We use same training strategy for training all the models on all the datasets: We first train the base models for 500 epochs to ensure their convergence, then we apply Adversarial Training (generic or our method) on the trained models for further 100 epochs. For both of the training phases, we use Adam optimizer[15] with 0.001 learning rate. The batch size is 128. We use 0.2 dropout rate and 1×10^{-5} L_2 norm to prevent overfitting. We follow a leave-one-out strategy to split training and test data, which is commonly used in sequential recommendation.

For the hyper-parameters, the magnitude ϵ is always set to 10 for all the datasets. The ablation study on different ϵ values can be found in Section 5.5. For parameters λ_1 and λ_2 in Eq. (4.10) are always 1 such that the three loss functions equally contribute to the training. Since no hyper-parameter used to compute the cascade effects, our proposed adversarial training method can be easily applied on new datasets and models without additional tuning effort.



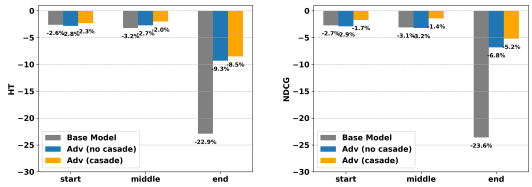
(a) SASRec on ML-1M (b) GRU4Rec on ML-1M

Figure 4: Influence of ϵ

5.4 Evaluating Metrics We use standard evaluation metrics, Normalized Discounted Cumulative Gain (NDCG) and Hit Ratio (HT), to evaluate the ranking performances of the recommendation models in this paper. Noted that negative sampling is not used in the evaluation. Instead, we rank over all items except the ones already in the user histories and retrieve the top-10 ranking list. This evaluation method is used in more and more state-of-the-arts [28, 31].

5.5 EXP1: Improvement on Ranking Accuracy In the first set of experiments, we evaluate the extent to which our proposed adversarial training method can improve the ranking accuracy in comparison to baseline adversarial training methods. We consider three baseline adversarial training approaches:

- **adv_linear[10]**: Adversarial training for MF-



(a) HT@10: SASRec on Video (b) NDCG@10: SASRec on Video

Figure 5: Drop of accuracy by attacking the the first, middle, and the last items in the user sequences respectively. The y-axis depicts negative values, with larger bars indicating larger decreases in accuracy of the recommendation models.

based recommenders, perturbing only user and item embeddings, i.e., $\boldsymbol{w_i}$ and $\boldsymbol{e_j}$.

- **adv_sequence[20, 19]**: Originally for LSTM text classification[20], later adapted for sequential recommendation[19], perturbing only sequence embeddings.

- **adv_global**: Our self-defined baseline, similar to our proposed method but without cascade-guided re-normalization. Serves as an ablation study on cascade value contributions.

For fair comparison, when implementing our method, we always re-normalize the average cascade values to 1 so that the total magnitudes of added perturbations are the same for all the baselines. The experiments results are shown in Table 2. First, we observe that compared to the pre-trained base models (no adversarial training), applying any of the adversarial training methods can generally further improve the ranking accuracy. This means that while adversarial training improves the models robustness on the adversarial examples, it also improve the models generalization on the clean data [20]. Second, our proposed method consistently outperforms all the other baselines on all the datasets. On average, it improves the ranking accuracy of the base model by 15.32% for SASRec and 15.99% for GRU4Rec. Compared to the second best baselines, it shows 18.16% **more improvement** of the base model for SASRec and 168.94% for GRU4Rec. Third, the improvements on the sparser datasets (i.e., Video, Beauty, and Clothing) are more significant than the other dataset. This is expected since when training complicated deep neural networks on sparse datasets, the trained models are more easily to overfit on the data and less stable. It's worth noting that all the other adversarial training methods fail to improve GRU4Rec on Clothing dataset possibly due to it's extreme sparsity, but the cascade-guided adversarial training method still performs well.

We illustrate the learning curves of the two training phases in Figure 3. As shown in the figures, the base models converge in the first 500 epochs of pre-training. Training for more epochs can hardly benefits the trained models. However, when applying the Cascade-guided Adversarial Training, the ranking accuracy can be quickly improved further, especially in the next 100 steps.

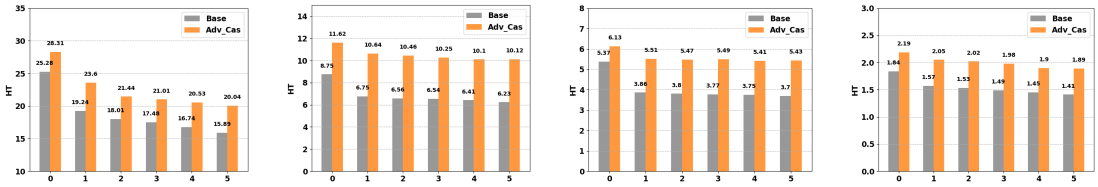
We also consider a related question: what should be the magnitude of the adversarial perturbations? The magnitude of adversarial perturbations, controlled by hyper-parameter ϵ in Eq. (4.7) and (4.9), impacts the model's robustness. Minor perturbations might not enhance robustness, while excessive perturbations can hinder learning. We perform ablation studies on ϵ, by setting its value from 0.1 to 50 as shown in Figure 4. We find our algorithm effective across a spectrum of ϵ. Even with small values like 1, can significantly improve the models' performance. The method shows the best performance when $\epsilon \in [10, 20]$ for the both base recommendation models. When ϵ is larger than 30, the models' accuracy start to drop, suggesting that the adversarial perturbations are too large, and useful information is lost during training.

As in Eq.(4.11), each interaction has two types of cascade effects on other interactions (i.e., cascade effects on direct later interactions and cascade effects on the interactions from other sequences). Please refer to supplementary materials for ablation study about how each component of cascade effects contribute to the final performance.

5.6 EXP2: Improvement on Robustness We evaluate the performance of the trained recommendation models in the presence of noise data. Notably, our evaluations perturb by substituting real items, not merely tweaking item embeddings, therefore, simply increasing the norm of the learned item embeddings can not lead to trivial solutions.

We ensure replacements are near the original in the embedding space for minimal perturbation. Since major changes in user history should eventually lead to different recommendations, hence, substituting a similar item better mimics "natural" data noise, like misclicks. Experimentally, we derive the most adverse minor replacements by first using a gradient-based attack on target item embeddings and then pairing with the nearest items in the gradient direction.

Since the proposed method is motivated based on the idea that recommendation models have varying robustness across a user's history, being more susceptible towards the end. We test this by replacing the first, middle, and last items in user sequences, then measuring



(a) SASRec on MovieLen-1M. (b) SASRec on Video. (c) SASRec on Beauty. (d) SASRec on Clothing.

Figure 6: Decreases of model accuracy by replacing K items at the end of user sequences according to HT@10.

accuracy drops. In Figure 5 using the Video dataset, replacing initial and middle items results in minimal accuracy reduction (under 3%). However, changing the last item sees a notable drop (23.6% in NDCG and 22.9% in Hit ratio). This underscores the models' vulnerability towards sequence ends. Our method significantly improves this end-sequence robustness, outperforming base models and standard adversarial training.

Next, We evaluate model robustness by replacing the last K items in user sequences. We choose K from 1 to 5 to show the trend of decreasing model accuracy. These tests are conducted on both base models across all four datasets. Figure 6 presents the results for the SASRec model while additional results are in the supplementary materials. Our algorithm-trained models consistently surpass the original models in overall performance, with a slower decline in accuracy at the same time. Notably, for the Video, Beauty, and Clothing datasets, modifying the last 5 items still results in a ranking accuracy higher than traditionally trained models on clean data, underscoring the efficacy of our method, especially on sparse datasets.

6 Conclusion

In this work, we introduce Cascade-guided Adversarial Training, a novel adversarial training algorithm designed for sequential recommender systems. This is the first adversarial training algorithm that considers the intrinsic properties of sequential recommendation models. Using our approach, we show that the trained sequential recommendaion models are more robust and accurate. Considering the small number of additional training epochs and the simplicity of the additional parameters involved in the proposed training process, the proposed method is very practical.

References

[1] Gediminas Adomavicius and Jingjing Zhang. On the stability of recommendation algorithms. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 47–54, 2010.

[2] Gediminas Adomavicius and Jingjing Zhang. Maximizing stability of recommendation algorithms: A collective inference approach. In *21st Workshop on Information Technologies and Systems (WITS)*, pages 151–56. Citeseer, 2011.

[3] Gediminas Adomavicius and Jingjing Zhang. Classification, ranking, and top-k stability of recommendation algorithms. *INFORMS Journal on Computing*, 28(1):129–147, 2016.

[4] Konstantina Christakopoulou and Arindam Banerjee. Adversarial attacks on an oblivious recommender. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*, pages 322–330. ACM, 2019.

[5] Yashar Deldjoo, Tommaso Di Noia, and Felice Antonio Merra. A survey on adversarial recommender systems: From attack/defense strategies to generative adversarial networks. *ACM Comput. Surv.*, 54(2):35:1–35:38, 2021.

[6] Ziwei Fan, Zhiwei Liu, Jiawei Zhang, Yun Xiong, Lei Zheng, and Philip S Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 433–442, 2021.

[7] Min Gao, Bin Ling, Quan Yuan, Qingyu Xiong, and Linda Yang. A robust collaborative filtering approach based on user relationships for recommendation systems. *Mathematical Problems in Engineering*, 2014, 2014.

[8] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[9] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[10] Xiangnan He, Zhankui He, Xiaoyu Du, and Tat-Seng Chua. Adversarial personalized ranking for recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 355–364, 2018.

[11] Balázs Hidasi and Alexandros Karatzoglou. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge*

management, pages 843–852, 2018.

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

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

[14] Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, and Wangyou Zhang. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 449–456, 2019.

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

[16] Zhiwei Liu, Ziwei Fan, Yu Wang, and Philip S Yu. 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*, pages 1608–1612, 2021.

[17] R Logesh, V Subramaniyaswamy, D Malathi, N Sivaramakrishnan, and Varadarajan Vijayakumar. Enhancing recommendation stability of collaborative filtering recommender system through bio-inspired clustering ensemble method. *Neural Computing and Applications*, 32(7):2141–2164, 2020.

[18] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[19] Jarana Manotumruksa and Emine Yilmaz. Sequential-based adversarial optimisation for personalised top-n item recommendation. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020, Virtual Event, China, July 25-30, 2020*, pages 2045–2048. ACM, 2020.

[20] Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*, 2016.

[21] Sejoon Oh, Berk Ustun, Julian McAuley, and Srijan Kumar. Rank list sensitivity of recommender systems to interaction perturbations. *31st ACM International Conference on Information and Knowledge Management (CIKM 2022)*, 2022.

[22] Michael O'Mahony, Neil Hurley, Nicholas Kushmerick, and Guénolé Silvestre. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technology (TOIT)*, 4(4):344–377, 2004.

[23] Zohreh Ovaisi, Shelby Heinecke, Jia Li, Yongfeng Zhang, Elena Zheleva, and Caiming Xiong. Rgrecsys: A toolkit for robustness evaluation of recommender systems. In K. Selcuk Candan, Huan Liu, Leman Akoglu, Xin Luna Dong, and Jiliang Tang, editors, *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 1597–1600. ACM, 2022.

[24] Dae Hoon Park and Yi Chang. Adversarial sampling and training for semi-supervised information retrieval. In *The World Wide Web Conference*, pages 1443–1453, 2019.

[25] David Shriver, Sebastian Elbaum, Matthew B Dwyer, and David S Rosenblum. Evaluating recommender system stability with influence-guided fuzzing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4934–4942, 2019.

[26] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 1441–1450, 2019.

[27] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

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

[29] Rey Reza Wiyatno, Anqi Xu, Ousmane Dia, and Archy de Berker. Adversarial examples in modern machine learning: A review. *CoRR*, 2019.

[30] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *Proceedings of the tenth ACM international conference on web search and data mining*, pages 495–503, 2017.

[31] Xu Xie, Fei Sun, Zhaoyang Liu, Shiwen Wu, Jinyang Gao, Jiandong Zhang, Bolin Ding, and Bin Cui. Contrastive learning for sequential recommendation. In *2022 IEEE 38th international conference on data engineering (ICDE)*, pages 1259–1273. IEEE, 2022.

[32] An Yan, Shuo Cheng, Wang-Cheng Kang, Mengting Wan, and Julian McAuley. Cosrec: 2d convolutional neural networks for sequential recommendation. In *Proceedings of the 28th ACM international conference on information and knowledge management*, pages 2173–2176, 2019.

[33] Zhenrui Yue, Zhankui He, Huimin Zeng, and Julian McAuley. Black-box attacks on sequential recommenders via data-free model extraction. In *Fifteenth ACM Conference on Recommender Systems*, pages 44–54, 2021.

[34] Hengtong Zhang, Yaliang Li, Bolin Ding, and Jing Gao. Practical data poisoning attack against next-item recommendation. In *Proceedings of The Web Conference 2020*, pages 2458–2464, 2020.