Extrapolation-Based Data Augmentation for Sequential Recommendation

Vinicius Gabriel Machado
viniciusgabrielmachado@gmail.com
Federal University of Paraná (UFPR)
Curitiba, Paraná, Brazil

Murilo F. L. Schmitt
mschmitt@unicentro.br
State University of the Midwest of Paraná (UNICENTRO)
Guarapuava, Paraná, Brazil

Eduardo J. Spinosa
spinosa@inf.ufpr.br
Federal University of Paraná (UFPR)
Curitiba, Paraná, Brazil

ABSTRACT

With the advent of deep learning in the recommendation field, a lot of work has been and still is being done to bring deep learning-based models to their full potential. One line of work, Self-Supervised Learning (SSL), focuses on extracting the maximum potential of datasets to improve recommendation performance, and at the same time, attempts to diminish data-related problems, such as data sparsity, that are commonly seen in machine learning techniques. One branch of SSL for recommender systems uses predictive strategies to create new labels and examples for training by, for instance, adding new interactions to the user's history of interactions. However, the existing models that explore this idea are somewhat limited. They focus on adding new interactions at the start of the sequences, ignoring the performance improvements that could be achieved by adding interactions in the middle and end of the sequences. We propose Extrapolation-based Sequence Augmentation for Sequential Recommendation (ESA4SRec), a model that uses the sequence reconstruction capabilities of BERT4Rec to generate new data at any position of a sequence by extrapolating the existing knowledge to unknown, novel interactions. The resulting augmented dataset is then used as input to a model-agnostic sequential recommender system. We compare our approach to related models and demonstrate the performance improvements when compared with the original datasets and the overall best performance of our method. ESA4SRec's code available at https://github.com/viniciusgm000/ESA4SRec.

KEYWORDS

Recommender Systems, Self-Supervised Learning, Sequential Recommendation, Data Augmentation

1 INTRODUCTION

In the last decade, deep learning has revolutionized countless fields [13, 16, 29]. Recent developments in Natural Language Processing (NLP), specifically with Transformers [39] and their novel approach to sequence processing used in machine translation, have shown incredible advancements in accuracy and computational cost, paving the way to many possibilities and improvements in numerous areas. Recommender systems, being one of them, has seen a plethora of studies in the last eight years that make use of the power of attention mechanisms and Transformers in general to improve recommendation accuracy [37], computational cost [17], explainability [22], and to perform data imputation [6], graph augmentation [42], and many other possibilities that span different types of recommender systems [20, 24, 30, 49].

From all the recommender systems types, those categorized as Sequence-aware [33] have received much attention from the research community because of their similarities with the NLP problem. Both try to capture the importance of different items or tokens in a user's sequence of interactions to predict the next item or token with the highest accuracy possible. SASRec [17] and BERT4Rec [37] have shown the relationship between sequential recommendation and machine translation by adapting the original unidirectional Transformer and the bidirectional counterpart [5] to the recommendation problem with a few modifications, and demonstrated their models' superiority compared to other deep learning methods [10, 11, 38]. Despite their effectiveness and promising results, they have shortcomings and cannot avoid some of the most difficult problems in deep learning, such as a sufficient data volume [7, 46].

Since then, many studies have been conducted, and some have focused on the Self-Supervised Learning (SSL) aspect to approach problems derived from the lack of sufficient information to train a complex model such as a Transformer-based recommender system [48]. These approaches rely on designing self-supervised tasks to generate new supervision signals that partially overcome the high data volume requirement problem. The models proposed by these studies make use of a different number of ideas. The ones categorized as predictive, especially ASReP [26] and BARec [15] that perform sample prediction, are the most relevant to us, since they inspired our approach to data augmentation. These models perform data augmentation in the most common sense, increasing the training samples by adding new data directly to the original dataset. They do that by training with the reversed sequences of interactions, aiming to learn to add new items iteratively at the start of a user's history. The augmented dataset is then used to train the same Transformer architecture again, this time on the correctly ordered dataset. BARec (formerly BiCAT) differs from ASReP by using the reversed and forward sequences at the same time, aiming to capture reversed correlations that align with the forward correlations to generate better augmentations.

Although these ideas are inspiring, the original methodology employed has been criticized for fundamental problems, concerning possible data leakage, and, after modifications aimed at addressing these issues, has difficulties overcoming other approaches [18]. They also limit data augmentation to the start of a sequence, not

In: Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia'2025). Rio de Janeiro, Brazil. Porto Alegre: Brazilian Computer Society, 2025.

ISSN 2966-2753

addressing the fact that the most important interactions are usually the most recent ones [33].

With this in mind, we propose a new model called Extrapolation-based Sequence Augmentation for Sequential Recommendation, or ESA4SRec. The main idea of our method is to use the power of bidirectional learning of models like BERT4Rec to create augmentations that are not limited to the start of a user's history sequence, extrapolating the knowledge gained from learning to predict items independent of the order, at any position, to introduce new interactions in the original dataset, bridging gaps that may exist on the system understanding of a user while making it easier to provide better recommendations by artificially expanding users' history of interactions. The position of these augmentations is sampled from a stochastic distribution. The model then treats these positions as prediction targets and creates augmented interactions. Lastly, the augmented dataset is used to train a model-agnostic sequential recommender that will make improved recommendations with the new dataset.

2 BACKGROUND

Following recent developments in Transformers and SSL for recommendation, we present a general view of recommender systems, followed by a discussion on sequential recommendation, SSL, and related works.

2.1 General Recommendation

In recommender systems, the main objective is to filter item catalogs through recommendation, aiming to provide a more concise user experience. With that in mind, we can design simple, non-personalized recommendation algorithms, for example, by recommending the most popular items in a database. However, this approach does not consider the target user's preferences, so it has a high chance of not satisfying a sizeable proportion of the user base.

Various approaches have been proposed to integrate the user's preferences into the recommendation algorithm. Some earlier ones are based on Matrix Factorization techniques [3]. These methods create latent representations (embeddings) of users and items and model a user's interest in an item as the inner product between a user and a candidate item. For this purpose, different sources of data can be used. One can create an embedding for a user using the available metadata, which describes personal aspects, or, if privacy is of major concern, the history of interactions with items can be another option by approaching the sequence of interactions as a representation of a user's preference. On the items' side, their embedding can be achieved by using the same principles and modeling the metadata or items seen together in the users' interaction history. These interactions can be implicit or explicit [12]. However, most available datasets are implicit-based, so most research is done by considering only implicit feedback, and so is ours.

Since these earlier developments, much has been proposed [19], with some works incorporating information like the order of the interactions [11], the moment that an interaction occurred, and the time between interactions [21], contextual information (metadata) [34], and even entirely new representation approaches, like knowledge-based graphs [40]. One line of work that stood out the most is sequence-aware recommender systems.

2.2 Sequential Recommendation

Sequence-aware or Sequential Recommender Systems (SRS) are recommendation algorithms that model the relationships between items in the user's sequence of interactions. By doing so, they aim to identify and capture sequential patterns, making it easier to predict future interactions with better accuracy.

Some earlier works on SRS focused on modeling the relationship between the current interaction and the previous one [35]. However, as much as the last interaction is usually the most important one [33] when predicting future interactions, the other interactions can still hold important relationships and patterns that could be decisive to make a good recommendation.

Different models have incorporated the last few or even the entire sequence of interactions. Some have considered each item's embedding in a sequence as an image, using convolutional techniques from computer vision to capture higher-order transitions between items [38]. Similarly, other models have applied different knowledge from fields beyond recommender systems, incorporating ideas from NLP, such as Recurrent Neural Networks (RNN) [11]. These approaches model the sequential aspect on an architectural level and create more refined representations considering short- and long-term dependencies. By doing so, they aim to capture a better understanding of the relationships between items.

Although RNN-based models are interesting, they have a major problem: training time requirements. Since these techniques model the sequential aspect at an architectural level, they are usually bound to processing one interaction at a time. New advancements in the NLP field have gained much attention on this aspect. Studies have shown the power of self-attention mechanisms and, more importantly, the Transformer [39] and its entirely self-attention-based structure, which enables sequence processing without limiting the input to a single item (or token in NLP) at a time. These techniques interpret each token as the sum of the representation of all relatable tokens (tokens that have been present in the same sequence). The idea is that tokens usually seen in the same sequences tend to have a more important relationship and are more informative when creating a good representation. The order of the tokens is still considered by a module called positional encoding. It provides a unique representation for each token, considering its position in the original sequence, and enables the model to process as many tokens simultaneously as possible.

Many approaches that use Transformers for SRS have been and are being proposed. Some of the earlier ones and most important are SASRec [17] and BERT4Rec [37]. The former adapts the original unidirectional Transformer for SRS, while the latter is an adaptation of BERT [5], a bidirectional Transformer. SASRec works with a next-item prediction task, so its training is modeled after the same principle by giving the model one interaction at a time, cumulatively, and trying to predict the next one. BERT4Rec, on the other hand, has access to all the training interactions and uses a Cloze task, which aims to train the model to reconstruct randomly corrupted portions of the original sequence.

2.3 Self-Supervised Learning

Despite the performance and computational cost achieved by SRS based on Transformers, they are still prone to the usual deep learning problems, with a sufficient data volume being one of the most important [7]. SSL emerged as a possible solution to this problem. Techniques based on this paradigm design self-supervised tasks to extract knowledge from unlabeled data, generating new supervision signals [4, 48] that can be used to alleviate the lack of sufficient data. By [48], SSL methods can be divided into four categories: contrastive, generative, predictive, and hybrid.

Contrastive methods usually approach each user/item/sequence as a class and try to approximate the representation of examples in the same class, while increasing the distance between the representations of examples in different classes. They often use data augmentation procedures, such as cropping/masking/reordering, to create these examples. For instance, in [45], the authors approached entire sequences of interactions as examples. In contrast, [25] expanded on the idea with novel augmentation procedures, [32] used an entirely different approach by making model-level augmentations instead of data-level to alleviate the representation degeneration problem.

Generative methods, on the other hand, work with the same principle as the Cloze task [5], they learn to reconstruct (predict) a portion of the original data. BERT4Rec [37] is one example in this category. [36] makes use of the idea to create a universal user representation by creating a general-purpose transfer learning model with nine different tasks, while [27] explores subsequence prediction instead of item prediction, learning to reconstruct entire sub-embeddings.

Predictive methods are similar to generative but differ in how the predictions are used. Generative methods reconstruct corrupted data, while predictive methods generate new data. ASReP [26] is one example of predictive SSL. The authors pre-trained the SASRec [17] model with the reversed sequences to predict new items at the start of each sequence until a threshold was reached. The augmented dataset was then used to train the model again, this time with the correctly-ordered sequence, and lastly, the model would predict the next interaction as testing. This article inspired us, but, as already stated, a data leakage problem has been pointed out in a recent study [18], concerning the use of the test interaction as input in the augmentation phase. BARec [15] expands on ASReP's idea by considering both reversed and correctly-ordered sequences.

Lastly, hybrid SSL methods aim to combine contrastive, generative, and predictive ideas. [1] pre-trains an attribute generator with a generative approach to create new examples to train a contrastive curriculum learning-based model later. [44] is similar to ASReP, but the objective is to create better user representations, not better recommendations. Initially, the model is pre-trained with a generative method, then it makes predictions at the end of a user's sequence.

Our approach differs from these methods by: (1) using BERT4Rec to augment each sequence, a model that learns to reconstruct a sequence from a corrupted version, which we believe to be more appropriate for sequence augmentation; (2) making augmentations in a non-iterative way, which we suppose can lead to error propagation, decreasing the quality of each consecutive augmentation; (3) being able to add new interactions in a distributed manner, not restricted to the start or end of sequences; (4) providing a model-agnostic recommendation module where the augmentation module is completely detached from the recommendation module.

3 PROPOSED MODEL

Inspired by works as ASReP, SASRec, and BERT4Rec, we present a hybrid SSL approach to data augmentation. It combines the power of sequence reconstruction learned by BERT4Rec to extrapolate its knowledge to unknown, randomly placed missing items on a given user's sequence of interactions. We hypothesize that there are gaps in a model's understanding of a user, experiences not represented in the current interaction history, that could be useful to make better recommendations by bridging the current interactions and providing smoother transitions between items. The following subsections present our framework, Extrapolation-based Sequence Augmentation for Sequential Recommendation (ESA4SRec). First, we discuss the model in general, followed by exploring the three main phases in detail.

Figure 1 presents the framework and its three main phases. In the first phase, *Pre-training* (Figure 1b), the BERT4Rec model is trained as normal, aiming to learn to reconstruct sequences from corrupted versions. Second phase, *Extrapolation augmentation* (Figure 1c), approaches all original sequences as if they were already corrupted by adding masks in random positions that are replaced with item predictions made by the model trained in the earlier phase. The augmented or extrapolated dataset is then passed to the third phase, *Training and recommendation* (Figure 1d), where a model-agnostic SRS is used to make the next-item prediction. For example, we use SASRec and BERT4Rec in this phase.

3.1 ESA4SRec

ESA4SRec assumes that every sequence of interactions is already corrupted and uses the Cloze task of models like BERT4Rec to reconstruct the "original" version. To simulate this condition, masks are inserted on sampled positions, and the model is tasked with predicting these masks. The idea is that in the training phase, the model would learn patterns in the entire dataset, which could be used to bridge these gaps created in the sequence. In the augmentation phase, it would generalize and extrapolate the knowledge acquired. The items inserted into the gaps would not harm an SRS's performance since they would only connect existing items, making it easier for another SRS to detect and predict patterns in a user's history of interactions.

The idea is similar to the sequence extension category present in [4]. However, instead of making iterative cumulative insertions at the end or start of a sequence, as in most models pointed out by the survey, our approach can generate new items in any position, all simultaneously, without the risk of propagating errors with each insertion. Our idea can also be compared with data imputation techniques [6, 14], where known missing values are filled with different methods. Although, in our case, we are purposefully creating supposed missing values.

3.2 Pre-Training

The pre-training phase consists of training the bidirectional Transformer for SRS, BERT4Rec, as designed. BERT4Rec learns to recreate

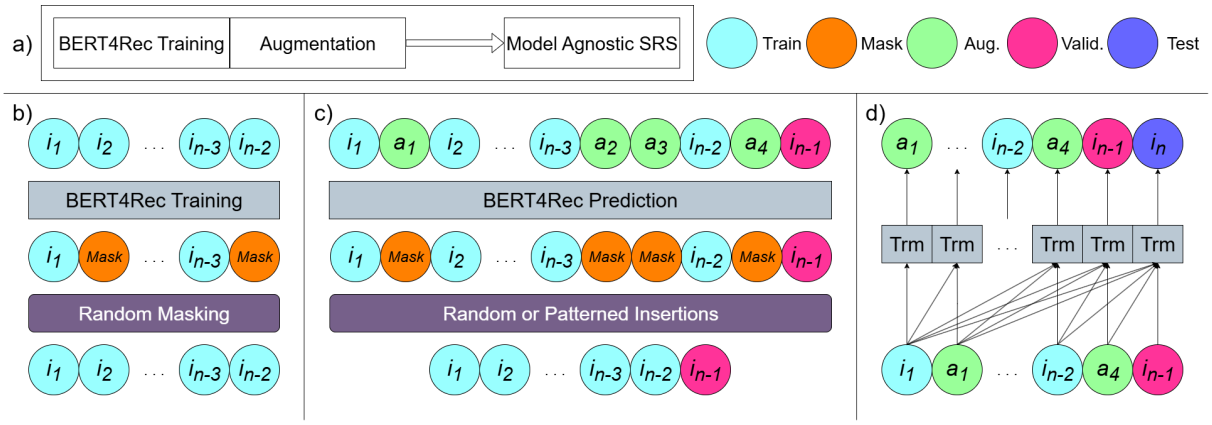

Figure 1: Our proposed framework, ESA4SRec. (a) A general view of the model. (b) The pre-training phase, where BERT4Rec is trained to reconstruct corrupted sequences. (c) The sequence extrapolation augmentation phase, where the model trained earlier is used to predict new interactions in randomly selected positions. (d) The training and recommendation phase, where the data-extrapolated dataset is used by a model-agnostic SRS (here we use SASRec as an example) for the next-item prediction. Interactions i are the original interactions, and a represent the augmented interactions. Trm represents the transformer model.

the original sequence from the corrupted version by randomly sampling a percentage of the interactions, masking these selected items, and asking to predict these items. The resulting model can extrapolate a limited view of the original data to a, hopefully, complete view. The training data consists of all items, except each user's validation and test interactions (second-to-last and last interaction, respectively). The original prediction phase is removed, and the new augmentation phase follows.

3.3 Extrapolation Augmentation

The sequence extrapolation augmentation phase starts with sampling positions to insert new unknown items (mask items) into the existing sequences. This sampling can be random or pre-determined with patterns. In our experiments, the random sampling usually performs better, but the pre-determined sampling has advantages in some datasets. A hyperparameter specifies the percentage of items to be added to each sequence to control both sampling methods. This percentage is proportional to each sequence length, and new insertions are made until this limit or the model's maximum length is reached.

The random sampling can be done with different distributions, such as normal, linear, or exponential, as exemplified in Figure 2). However, we discovered that the exponential distribution tail-focused is the best in almost every case, making it the default distribution.

The pre-determined sampling creates windows of augmentation between every original item (interval), and makes an insertion in each window from end to start until the maximum number of insertions is reached, as illustrated in Figure 3. Different values for the interval have been tested, and one item between every window performed the best in our tests.

After insertion, the model trained in the last phase predicts the inserted masks for every sequence. Every item, excluding items already present, is ranked for each position simultaneously, and the algorithm selects the one that ranked first for every position from right to left, avoiding duplicate items.

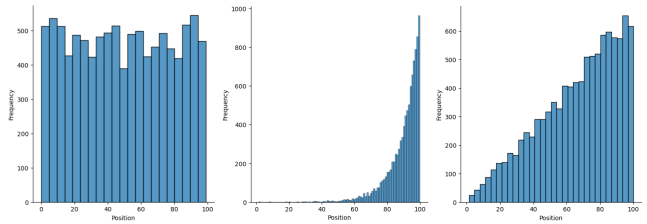

Figure 2: An example of normal, exponential, and linear distribution used in the random sampling, respectively. Here, 100 positions were sampled 10000 times.

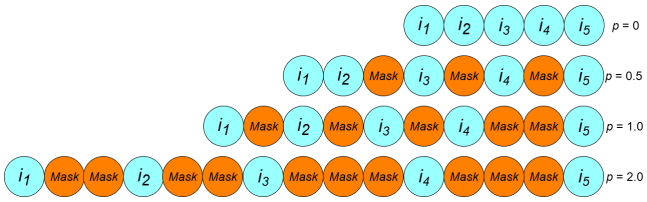

Figure 3: An example of patterned sampling with windows of augmentation for every original item. The insertions are made from end to start until the maximum number of insertions, determined by $ceiling(p * sequencelength)$, is reached. Where p is a hyperparameter that specifies the number of augmentations that will be done as a percentage relative to the sequence length.

3.4 Training and Recommendation

After augmenting the sequences, the resulting dataset is passed to another SRS. In the Figure 1, we used SASRec to show the model-agnostic characteristic of our solution. However, we also tested the idea with BERT4Rec to further demonstrate the capabilities of our approach. Any SRS could be used if the data format is recognized. If not, it can be converted easily. We call the version that uses SASRec as the recommendation module **ESA4SRec-S**, and the version that uses BERT4Rec, **ESA4SRec-B**.

SASRec's training was not altered. It uses the same next-item prediction task specified on [17]. At the end of the training phase, the model is tasked with predicting the future item, the test interaction, given the training and validation interactions. The same is true for BERT4Rec, it is the same model that we compare with in our experiments, we just adjusted the dataset input.

4 EXPERIMENTS

In this section, we present our experimental setup, results, and analyses, aiming to answer the following research questions (RQs):

- RQ1: (Recommendation test) Is ESA4SRec capable of improving sequential recommendation performance?
- RQ2: (Sequence length test) How does ESA4SRec perform over different sequence length intervals?
- RQ3: (Model-agnostic test) Are performance improvements provided by ESA4SRec independent of the model used for recommendation?

4.1 Datasets and Metrics

To validate our model, we used the following datasets:

- Amazon Beauty and Amazon Video: Datasets from the Amazon marketplace platform, here filtered to the datasets from the "Beauty" and "Video Games" categories [9, 28];
- MovieLens 1M and MovieLens 20M: Datasets of varying amounts of interactions from the MovieLens website [8], a platform where users can find details about their favorite movies, manage the ones that they have already watched, and get recommendations;
- Steam: A dataset introduced in [17] obtained from crawling the largest distribution platform of video games.

To facilitate performance comparisons between different methods, all datasets were obtained from SASRec's GitHub page[1] except for the MovieLens 20M, which was acquired from the GroupLens website[2].

All datasets were pre-processed (again, in the case of SASRec's datasets) to guarantee the quality of the original data by removing users and items with less than five interactions. Originally, SASRec's pre-processing removed users and items with this condition. However, new users and items can fall below this threshold by doing so, which is not usually considered in other works. To avoid this, we created a recursive procedure to remove users and items as long as instances with fewer than five interactions exist in the dataset. In addition to this procedure, we limited the sequence length to the 50 most recent interactions for all datasets, except for the MovieLens 1M and 20M, which were limited to the 200 most recent interactions. We present the resulting statistics for each dataset in Table 1. In addition to the usual number of users/items/interactions, sparsity, and average length, we added the 25% and 75% quantiles, which will be important to evaluate the model's performance in different intervals of sequences' length.

[1]https://github.com/kang205/SASRec
[2]https://grouplens.org/datasets/movielens/20m/

Table 1: Datasets Statistics.

Dataset	Beauty	ML 1M	Steam	Video	ML 20M
Source	[17]	[17]	[17]	[17]	[8]
# Users	22011	6040	281137	23933	138493
# Items	11660	3328	11630	10072	15428
# Inter.	190003	660552	3143617	216873	12877320
Sparsity	0.99926	0.96714	0.99904	0.99910	0.99397
Avg. Len.	8.632	109.363	11.182	9.062	92.982
25% Q.	5	44	6	5	35
75% Q.	9	199	12	10	154

Following [17], all ratings in the MovieLens 20M dataset were treated as interactions, independent of their given rating.

The sequences were split into test, validation, and training: the last interaction, the second-to-last interaction, and all the remaining interactions, respectively. For the ASReP and ESA4SRec executions, the validation interaction was considered in the augmentation and test phases, with the principle that the models should be able to use all existing interactions, as long as they do not represent a future interaction (an interaction that has not yet occurred). Following this idea, the test interaction was not considered in any training or augmentation phases.

Following the most common metrics for evaluation found in the literature of sequential recommenders, we measured the Normalized Discounted Cumulative Gain (NDCG) and Hit-Ratio (HR).

4.2 Compared Methods

To validate our model's effectiveness, we compare it with the models that served as the basis for our implementation, SASRec [17] and BERT4Rec [37]. In addition to these models, we compare our approach to ASReP [26], the work that inspired our model and works similarly to ESA4SRec, by extending the original sequences.

In our research, we considered making comparisons with other models. However, to the best of our knowledge, ASReP and BARec are the only models that extend the sequences directly while aiming to achieve better recommendation performance with the augmented sequences, and so, are the closest to our approach. Models like L2Aug [41], STEAM [23], and DR4SR [47], pointed by [4], were considered as candidates, but we did not find them suitable for a fair comparison: L2Aug works by creating entirely new sequences; STEAM tries to remove noise in a sequence and uses the resulting sequence, together with the original, as input to the next-item prediction training; DR4SR learns to extract patterns that can be used to personalize the dataset to different target models.

The implementation of ASReP was modified by removing the test interaction from the augmentation phase, addressing the concerns presented in [18].

BARec [15] was a possible candidate to compare our approach to, considering that it extends ASReP, the main differences being (1) that the model considers both the reversed sequence and the original sequence as input to train the model, and (2) the model's backbone is now model-agnostic. The data leakage problem was not addressed, so we decided not to use it.

While developing ESA4SRec, we found different implementations of BERT4Rec. We decided to use a version made on PyTorch, considering the framework's ease-of-use library compared to the original implementation on Tensorflow version 1.x. BERT4RecVAE [2] is a GitHub repository that implements BERT4Rec and a variational auto-encoder on PyTorch. It has been tested in a study [31] of BERT4Rec's overall performance, where the authors compared different implementations to understand why the model showed an inconsistent performance between various articles, and it performed relatively well. We use the BERT4Rec implementation of BERT4RecVAE as the backbone for our model and will use it as a comparison instead of the original version in Tensorflow.

4.3 Implementation Details and Setup

To evaluate all models, we follow the common, recent methodology adopted in most SRS works. We adapted all models to rank every item (full-sampling) instead of making a sampled ranking like [17] originally did, avoiding potential bias problems. We also added interval evaluations to every model. In addition to evaluating all sequences, the models can report the performance in sequences that originally fall in different length intervals. The intervals considered are: 0% to 25% (inclusive) quantile, 25% (not inclusive) to 75% (inclusive) quantile, and 75% (not inclusive) to 100% (inclusive) quantile.

The following model-specific adaptations were made. We adjusted the test in BERT4RecVAE, which did not consider the validation interaction, making the model predict two interactions in advance instead of one, giving it a small disadvantage. We added dumping to BERT4RecVAE's implementation as seen in the Tensorflow BERT4Rec. Originally, the Cloze task creates multiple corrupted variants from a single sequence, a behavior controlled by a hyperparameter called "dump". BERT4RecVAE did not implement this, and was creating a single variant from each sequence, so we added this option and chose 10 as the value for this hyperparameter, as in the original code. Finally, to get BERT4RecVAE as close as possible to the original implementation, in its current form, it does not use the 80-10-10 corruption method [43], where 80% of the time a corruption is replaced with the mask token, 10% with a random item, and 10% the corruption does not occur. All corruptions are replaced by the mask token 100% of the time.

For the SASRec and ASReP settings, we use the original hyperparameters found in the articles and their respective GitHub pages. For the datasets not present in their analysis, we conducted the grid search as pointed out in the articles. To make a fair comparison, all models use 64 dimensions as the embedding size, and the maximum sequence length was set as 200 interactions for the MovieLens datasets and 50 for the remaining.

To properly configure BERT4RecVAE and ESA4SRec, we made a grouped grid search (grid searching a few parameters at a time) with a small subset (10% for the MovieLens 20M and Steam datasets, 30% for the remaining) of each dataset. The learning rate was chosen between {0., 0.01, 0.001}, dropout from {0.2, 0.4, 0.6}, number of transformer blocks and heads from {1, 2, 4}, and mask probability between {0.1, 0.2, 0.4, 0.6, 0.8}. We also used early stopping for these two methods, finishing the training early if the validation metric did not improve by 2% or more. And specifically for ESA4SRec, the hyperparameter p that controls the amount of augmentations was chosen between {1, 2, 3, 4, 5}. Recall that the number of interactions added is determined by $ceiling(p{*}sequencelength)$ or the maximum length being reached.

The best thresholds for data augmentation in ASReP were found by grid searching between the values {10, 20, 50} for the datasets with maximum length equal to 50 and from {50, 100, 200} for the ones with maximum length equal to 200.

4.4 Performance Analysis

Table 2 presents the performance achieved with the best setting for the SASRec, BERT4RecVAE, ASReP, and ESA4SRec methods. We present the models' performance for NDCG@10 and HR@10. We also show the relative performance gain of our approach (in bold) over the best result from the compared models (underlined).

SASRec is the worst performer in our tests, followed by ASReP in the non-MovieLens datasets, and BERT4RecVAE in the MovieLens datasets. The modified version of ASReP has difficulties improving the recommendation performance in datasets that have shorter sequences compared to MovieLens. We believe that this is caused by the iterative augmentation procedure and mostly by the fact that the augmentation process is done at the start of sequences, since the recent interactions are usually the most important ones. In preliminary studies (absent given the lack of space), we have compared our approach when making augmentations focused on the first half of a sequence and the second half. We found that the latter performed significantly better.

In general, we can see that ESA4SRec performs the best, with NDCG improvements between 11.9 and 65.6% and HR between 2.8 and 40.4%. The exception is the MovieLens 20M dataset, where ESA4SRec-B performed slightly worse than ASReP in the NDCG metric. Table 3, which presents the performance achieved for different quantiles of original length in the MovieLens 20M dataset, sheds some light on the root of the problem. The difference was caused by the sequences that fall within the first quantile, sequences that go up to 35 interactions in length (Table 1), where ASReP performed significantly better in the NDCG metric. At the moment, we are not able to ascertain the cause of this, given that this was the only occurrence.

Another exception happens in the Steam dataset, ESA4SRec-S achieved the best results, while ESA4SRec-B performed significantly worse than its SASRec counterpart. Table 4 shows the interval metric evaluation for the Steam dataset. We can see that, strangely, ESA4SRec-B had a worse performance in every interval. In this case, we theorize that it was caused by overfitting. Comparing the results between the default BERT4Rec and our ESA4SRec-B, we can see that they are similar in the first and second intervals. In our experiments, we simplified the grid-search procedure of our approach's recommendation module (SASRec or BERT4Rec) by using the same parameters of the respective default models, found

Table 2: Recommendation performance comparison. We report the results for NDCG@10 and HR@10.

	Beauty		Steam		Video		MovieLens1M		MovieLens20M	
	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR	NDCG	HR
SASRec	0.024	0.055	0.058	0.112	0.040	0.091	0.106	0.226	0.076	0.162
BERT4RecVAE	0.037	0.070	0.065	0.123	0.055	0.105	0.141	0.257	0.125	0.225
ASReP	0.041	0.069	0.059	0.108	0.050	0.097	0.180	0.299	0.202	0.288
ESA4SRec-S	0.043	0.070	**0.101**	**0.163**	0.084	0.145	0.202	**0.372**	0.187	**0.332**
ESA4SRec-B	**0.046**	**0.072**	0.065	0.124	**0.091**	**0.147**	**0.206**	0.316	**0.197**	0.285
ESA4SRec-S / SASRec (%)	75.6	27.1	75.5	45.8	111.7	59.5	91.1	64.7	145.2	105.4
ESA4SRec-B / BERT4RecVAE (%)	25.8	2.8	-0.6	0.7	65.6	40.4	45.8	22.9	58.1	26.7
Bold / Underlined (%)	11.9	2.8	54.8	32.4	65.6	40.4	14.5	24.1	-2.5	15.6

Table 3: Interval performance comparison - MovieLens20M.

	[0%-25%]		]25%-75%]		]75%-100%]	
	NDCG	HR	NDCG	HR	NDCG	HR
SASRec	0.102	0.219	0.076	0.162	0.049	0.102
BERT4R	0.180	0.315	0.122	0.223	0.072	0.135
ASReP	0.346	0.441	0.200	0.296	0.057	0.110
E-S	0.261	**0.455**	**0.211**	**0.373**	0.062	0.125
E-B	**0.291**	0.399	0.209	0.302	**0.075**	**0.134**
E-S / S	155.5	107.7	177.3	130.7	25.7	21.7
E-B / B	61.9	26.6	71.3	35.1	4.4	-0.1
B / U	-15.7	3.3	5.2	25.7	4.4	-0.1

Table 4: Interval performance comparison - Steam.

	[0%-25%]		]25%-75%]		]75%-100%]	
	NDCG	HR	NDCG	HR	NDCG	HR
SASRec	0.059	0.116	0.058	0.113	0.055	0.105
BERT4R	0.068	0.128	0.066	0.124	0.059	0.113
ASReP	0.063	0.112	0.060	0.110	0.052	0.099
E-S	**0.095**	**0.156**	**0.106**	**0.169**	**0.104**	**0.165**
E-B	0.060	0.115	0.067	0.128	0.072	0.132
E-S / S	60.8	34.9	82.3	50.3	89.6	57.7
E-B / B	-12.2	-9.7	1.3	2.6	21.9	17.2
B / U	39.1	22.0	60.5	36.0	76.5	46.1

in the original papers, and through grid-search for the new datasets. We hypothesize that the original BERT4Rec already struggled with the amount of data in the original Steam dataset, and that the BERT4Rec recommendation module in ESA4SRec-B was not able to handle the augmented dataset with the same parameters, requiring adjustments.

Our approach achieved a similar relative performance gain in all datasets, with the exception of the MovieLens and Amazon Beauty datasets. The former was probably caused by ASReP being unusually good in the MovieLens datasets. For the latter, we are not sure what has caused this. For now, we hypothesize that this could be a consequence of the dataset characteristics, making it a difficult dataset.

Compared to other datasets, the MovieLens datasets are relatively easy for SRS models to achieve high performance values, even when using full-sampling for ranking. We believe that this is caused by the high average length of the sequences and not the number of items or sparsity, considering that the models achieved high values even with the MovieLens 20M dataset, which has a similar number of items and sparsity to the other datasets.

Answering RQ1 - Is ESA4SRec capable of improving sequential recommendation performance?: Compared to BERT4Rec and SASRec, we can see that ESA4SRec-B and ESA4SRec-S, respectively, increased the metrics of the original models significantly. Considering that the hyperparameters were shared between the original models and the recommendation module of ESA4SRec, this improvement was caused by the data augmentation process.

Answering RQ2 - How does ESA4SRec perform over different sequence length intervals?: With the peculiarity of the MovieLens 20M dataset (Table 3) and the Steam dataset (Table 4). The other datasets (not reported here, considering the limited space) achieved higher metric values independently of the sequence length. The sequences in the latter intervals, the longest, were able to reach increasingly higher metrics, except for the MovieLens datasets, which are already close to the maximum sequence length in the last interval. This indicates the already known fact that users with longer interaction sequences are usually easier to recommend to and, with the same principle, to augment.

Table 5 shows the difference between patterned and random exponential augmentation. The stochastic sampling method yielded better results in all datasets, except for the MovieLens 1M and 20M datasets, where the patterned method performed significantly better. We theorize that these datasets could be overfitting in the most recent interactions, opening opportunities for performance gains at the start of the sequences. This could also explain why ASReP performs the best in these datasets, even if it only makes augmentations at the start of a sequence. All metrics were reported with these results in mind. MovieLens executions used patterned sampling, while the remaining datasets used random exponential sampling.

Table 5: Performance comparison between random exponential and patterned augmentation for ESA4SRec-S. Random is the best overall, except in the MovieLens datasets.

		Patterned	Random
Beauty	NDCG	0.039	**0.043**
	HR	0.066	**0.070**
Steam	NDCG	0.086	**0.101**
	HR	0.150	**0.163**
Video	NDCG	0.077	**0.084**
	HR	0.140	**0.145**
MovieLens1M	NDCG	**0.203**	0.172
	HR	**0.372**	0.313
MovieLens20M	NDCG	**0.187**	0.180
	HR	**0.332**	0.311

Answering RQ3 - Are performance improvements provided by ESA4SRec independent of the model used for recommendation?: As we could see in Tables 2, 3, and 4. It is possible to improve recommendation accuracy in different models with our approach by simply changing the model that receives the augmented dataset, and we were able to achieve this in most datasets without adjusting the hyperparameters. However, as we believe to be the case for the Steam dataset (Table 4), it can be a necessary step for some datasets.

In summary, ESA4SRec was able to improve the original models significantly, especially ESA4SRec-S. ESA4SRec-B had some peculiarities, which we believe to be related to the fact that the model's parameters were not adjusted to the augmented dataset. Relatively to the best model from the compared methods, our approach was able to improve NDCG and HR with a significant margin, primarily on datasets with shorter sequences, something not achievable by ASReP. On datasets with longer sequences, as MovieLens 1M and 20M, we were able to provide a competitive performance, improving mainly HR. Our results showed a performance gain even in the interval evaluation, improving recommendation independently of the original sequence length, with exceptions mainly due to the MovieLens datasets.

5 CONCLUSION

In this paper, inspired by models such as ASReP and BARec, we proposed a novel approach to sequence augmentation that leverages the power of sequence reconstruction of the BERT4Rec model to augment sequences at any position. Our approach, ESA4SRec, trains the BERT4Rec model as usual and uses the trained model to extrapolate the original sequences by adding new interactions in distributed positions, which can be sampled with a patterned or random method. Extensive tests showed that ESA4SRec achieved the overall best performance in the majority of the datasets tested. We also proposed two different versions of our model, one that uses SASRec and the other BERT4Rec, ESA4SRec-S, and ESA4SRec-B, respectively, as a recommendation module. These variants showed that our approach is capable of improving recommendation performance for different models, demonstrating the model-agnostic characteristic. We also tested the performance in various intervals of sequence length, and showed that it is possible to improve the performance independently of the original length. For future works, we list some options: (1) Test ESA4SRec in session-based datasets, an even more extreme scenario with fewer interactions; (2) Explore the model's computational cost compared to other approaches. In our tests, the model showed to be relatively fast thanks to the early stopping used, which interrupted the training in less than five validations in most cases. However, it can be an interesting topic; (3) Test ESA4SRec in datasets similar to MovieLens 1M and 20M, such as the Gowalla dataset, to verify if the patterned augmentation superiority is an exception to the MovieLens datasets; (4) Study the model's performance in the augmented datasets with a dedicated hyperparameter grid search before training, specially for the Steam dataset, instead of using the same hyperparameters from the non-augmented datasets.

ACKNOWLEDGMENTS

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001 - Program of Academic Excellence (PROEX).

REFERENCES

[1] Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Jing Cai, Yancheng He, Cunxiang Yin, and Ji-Rong Wen. 2021. Contrastive Curriculum Learning for Sequential User Behavior Modeling via Data Augmentation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 3737–3746. doi:10.1145/3459637.3481905

[2] Jae-Won Chung and Sung Min Cho. 2019. BERT4Rec-VAE-Pytorch. https://github.com/jaywonchung/BERT4Rec-VAE-Pytorch.

[3] Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. 2010. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the Fourth ACM Conference on Recommender Systems* (Barcelona, Spain) *(RecSys '10)*. Association for Computing Machinery, New York, NY, USA, 39–46. doi:10.1145/1864708.1864721

[4] Yizhou Dang, Enneng Yang, Yuting Liu, Guibing Guo, Linying Jiang, Jianzhe Zhao, and Xingwei Wang. 2024. Data Augmentation for Sequential Recommendation: A Survey. arXiv:2409.13545 [cs.IR] https://arxiv.org/abs/2409.13545

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. doi:10.18653/v1/N19-1423

[6] Zhicheng Ding, Jiahao Tian, Zhenkai Wang, Jinman Zhao, and Siyang Li. 2024. Data Imputation using Large Language Model to Accelerate Recommendation System. arXiv:2407.10078 [cs.IR] https://arxiv.org/abs/2407.10078

[7] Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. 2020. Deep Learning for Sequential Recommendation: Algorithms, Influential Factors, and Evaluations. *ACM Trans. Inf. Syst.* 39, 1, Article 10 (Nov. 2020), 42 pages. doi:10.1145/3426723

[8] Maxwell F. Harper and Joseph A. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5, 4, Article 19 (Dec. 2015), 19 pages. doi:10.1145/2827872

[9] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web* (Montréal, Québec, Canada) *(WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 507–517. doi:10.1145/2872427.2883037

[10] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent Neural Networks with Top-k Gains for Session-based Recommendations. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 843–852. doi:10.1145/3269206.3271761

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. arXiv:1511.06939 [cs.LG] https://arxiv.org/abs/1511.06939

[12] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *2008 Eighth IEEE International Conference on Data Mining*. Institute of Electrical and Electronics Engineers, New York, NY, USA, 263–272. doi:10.1109/ICDM.2008.22

[13] Jian Huang, Junyi Chai, and Stella Cho. 2020. Deep learning in finance and banking: A literature review and classification. *Frontiers of Business Research in China* 14, 1 (2020), 13. doi:10.1186/s11782-020-00082-6

[14] Won-Seok Hwang, Shaoyu Li, Sang-Wook Kim, and Kichun Lee. 2018. Data imputation using a trust network for recommendation via matrix factorization. *Computer Science and Information Systems* 15, 2 (2018), 347–368.

[15] Juyong Jiang, Peiyan Zhang, Yingtao Luo, Chaozhuo Li, Jae Boum Kim, Kai Zhang, Senzhang Wang, Sunghun Kim, and Philip S. Yu. 2025. Improving Sequential Recommendations via Bidirectional Temporal Data Augmentation With Pre-Training. *IEEE Transactions on Knowledge and Data Engineering* 37, 5 (2025), 2652–2664. doi:10.1109/TKDE.2025.3546035

[16] Andreas Kamilaris and Francesc X. Prenafeta-Boldú. 2018. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture* 147 (2018), 70–90. doi:10.1016/j.compag.2018.02.016

[17] Wang-Cheng Kang and Julian McAuley. 2018. Self-Attentive Sequential Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. Institute of Electrical and Electronics Engineers, New York, NY, USA, 197–206. doi:10.1109/ICDM.2018.00035

[18] Kibum Kim, Dongmin Hyun, Sukwon Yun, and Chanyoung Park. 2023. MELT: Mutual Enhancement of Long-Tailed User and Item for Sequential Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 68–77. doi:10.1145/3539618.3591725

[19] Yehuda Koren, Steffen Rendle, and Robert Bell. 2022. *Advances in Collaborative Filtering*. Springer US, New York, NY, 91–142. doi:10.1007/978-1-0716-2197-4_3

[20] Chaoliu Li, Lianghao Xia, Xubin Ren, Yaowen Ye, Yong Xu, and Chao Huang. 2023. Graph Transformer for Recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Taipei, Taiwan) *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 1680–1689. doi:10.1145/3539618.3591723

[21] Jiacheng Li, Yujie Wang, and Julian McAuley. 2020. Time Interval Aware Self-Attention for Sequential Recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining* (Houston, TX, USA) *(WSDM '20)*. Association for Computing Machinery, New York, NY, USA, 322–330. doi:10.1145/3336191.3371786

[22] Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized Transformer for Explainable Recommendation. arXiv:2105.11601 [cs.IR] https://arxiv.org/abs/2105.11601

[23] Yujie Lin, Chenyang Wang, Zhumin Chen, Zhaochun Ren, Xin Xin, Qiang Yan, Maarten de Rijke, Xiuzhen Cheng, and Pengjie Ren. 2023. A Self-Correcting Sequential Recommender. In *Proceedings of the ACM Web Conference 2023* (Austin, TX, USA) *(WWW '23)*. Association for Computing Machinery, New York, NY, USA, 1283–1293. doi:10.1145/3543507.3583479

[24] Qidong Liu, Jiaxi Hu, Yutian Xiao, Xiangyu Zhao, Jingtong Gao, Wanyu Wang, Qing Li, and Jiliang Tang. 2024. Multimodal Recommender Systems: A Survey. *ACM Comput. Surv.* 57, 2, Article 26 (Oct. 2024), 17 pages. doi:10.1145/3695461

[25] Zhiwei Liu, Yongjun Chen, Jia Li, Philip S. Yu, Julian McAuley, and Caiming Xiong. 2021. Contrastive Self-supervised Sequential Recommendation with Robust Augmentation. arXiv:2108.06479 [cs.IR] https://arxiv.org/abs/2108.06479

[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* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1608–1612. doi:10.1145/3404835.3463036

[27] Jianxin Ma, Chang Zhou, Hongxia Yang, Peng Cui, Xin Wang, and Wenwu Zhu. 2020. Disentangled Self-Supervision in Sequential Recommenders. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Virtual Event, CA, USA) *(KDD '20)*. Association for Computing Machinery, New York, NY, USA, 483–491. doi:10.1145/3394486.3403091

[28] 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* (Santiago, Chile) *(SIGIR '15)*. Association for Computing Machinery, New York, NY, USA, 43–52. doi:10.1145/2766462.2767755

[29] Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. 2017. Deep learning for healthcare: review, opportunities and challenges. *Briefings in Bioinformatics* 19, 6 (05 2017), 1236–1246. arXiv:https://academic.oup.com/bib/article-pdf/19/6/1236/27119191/bbx044.pdf doi:10.1093/bib/bbx044

[30] Narjes Nikzad-Khasmakhi, Mohammad Ali Balafar, Mohammad Reza Feizi-Derakhshi, and Cina Motamed. 2021. BERTERS: Multimodal representation learning for expert recommendation system with transformers and graph embeddings. *Chaos, Solitons & Fractals* 151 (2021), 111260. doi:10.1016/j.chaos.2021.111260

[31] Aleksandr Petrov and Craig Macdonald. 2022. A Systematic Review and Replicability Study of BERT4Rec for Sequential Recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems* (Seattle, WA, USA) *(RecSys '22)*. Association for Computing Machinery, New York, NY, USA, 436–447. doi:10.1145/3523227.3548487

[32] Ruihong Qiu, Zi Huang, Hongzhi Yin, and Zijian Wang. 2022. Contrastive Learning for Representation Degeneration Problem in Sequential Recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining* (Virtual Event, AZ, USA) *(WSDM '22)*. Association for Computing Machinery, New York, NY, USA, 813–823. doi:10.1145/3488560.3498433

[33] Massimo Quadrana, Paolo Cremonesi, and Dietmar Jannach. 2018. Sequence-Aware Recommender Systems. *ACM Comput. Surv.* 51, 4, Article 66 (July 2018), 36 pages. doi:10.1145/3190616

[34] Shaina Raza and Chen Ding. 2019. Progress in context-aware recommender systems — An overview. *Computer Science Review* 31 (2019), 84–97. doi:10.1016/j.cosrev.2019.01.001

[35] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web* (Raleigh, North Carolina, USA) *(WWW '10)*. Association for Computing Machinery, New York, NY, USA, 811–820. doi:10.1145/1772690.1772773

[36] Kyuyong Shin, Hanock Kwak, Kyung-Min Kim, Minkyu Kim, Young-Jin Park, Jisu Jeong, and Seungjae Jung. 2021. One4all User Representation for Recommender Systems in E-commerce. arXiv:2106.00573 [cs.IR] https://arxiv.org/abs/2106.00573

[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* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1441–1450. doi:10.1145/3357384.3357895

[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* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 565–573. doi:10.1145/3159652.3159656

[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. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[40] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *The World Wide Web Conference* (San Francisco, CA, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 3307–3313. doi:10.1145/3308558.3313417

[41] 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* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2183–2194. doi:10.1145/3485447.3512147

[42] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. 2024. LLMRec: Large Language Models with Graph Augmentation for Recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining* (Merida, Mexico) *(WSDM '24)*. Association for Computing Machinery, New York, NY, USA, 806–815. doi:10.1145/3616855.3635853

[43] Alexander Wettig, Tianyu Gao, Zexuan Zhong, and Danqi Chen. 2023. Should You Mask 15% in Masked Language Modeling?. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, Andreas Vlachos and Isabelle Augenstein (Eds.). Association for Computational Linguistics, Dubrovnik, Croatia, 2985–3000. doi:10.18653/v1/2023.eacl-main.217

[44] Chuhan Wu, Fangzhao Wu, Tao Qi, Jianxun Lian, Yongfeng Huang, and Xing Xie. 2020. PTUM: Pre-training User Model from Unlabeled User Behaviors via Self-supervision. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, Trevor Cohn, Yulan He, and Yang Liu (Eds.). Association for Computational Linguistics, Online, 1939–1944. doi:10.18653/v1/2020.findings-emnlp.174

[45] 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)*. Institute of Electrical and Electronics Engineers, New York, NY, USA, 1259–1273. doi:10.1109/ICDE53745.2022.00099

[46] Joo yeong Song and Bongwon Suh. 2022. Data Augmentation Strategies for Improving Sequential Recommender Systems. arXiv:2203.14037 [cs.IR] doi:10.48550/ARXIV.2203.14037

[47] Mingjia Yin, Hao Wang, Wei Guo, Yong Liu, Suojuan Zhang, Sirui Zhao, Defu Lian, and Enhong Chen. 2024. Dataset Regeneration for Sequential Recommendation. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery,

New York, NY, USA, 3954–3965. doi:10.1145/3637528.3671841

[48] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Jundong Li, and Zi Huang. 2024. Self-Supervised Learning for Recommender Systems: A Survey. *IEEE Transactions on Knowledge and Data Engineering* 36, 1 (2024), 335–355. doi:10.1109/TKDE.2023.3282907

[49] Jie Zou, Evangelos Kanoulas, Pengjie Ren, Zhaochun Ren, Aixin Sun, and Cheng Long. 2022. Improving Conversational Recommender Systems via Transformer-based Sequential Modelling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval* (Madrid, Spain) *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 2319–2324. doi:10.1145/3477495.3531852