Learning Attribute as Explicit Relation for Sequential Recommendation

Gang Liu
University of Notre Dame
Notre Dame, United States
gliu7@nd.edu

Fan Yang
Amazon
Seattle, United States
fnam@amazon.com

Yang Jiao
Amazon
Seattle, United States
jaoyan@amazon.com

Alireza Bagheri Garakani
Amazon
Seattle, United States
alirezg@amazon.com

Tian Tong
Amazon
Seattle, United States
tongtn@amazon.com

Yan Gao
Amazon
Seattle, United States
yanngao@amazon.com

Meng Jiang
University of Notre Dame
Notre Dame, United States
mjiang2@nd.edu

ABSTRACT

The data on user behaviors is sparse given the vast array of user-item combinations. Attributes related to users (e.g., age), items (e.g., brand), and behaviors (e.g., co-purchase) serve as crucial input sources for item-item transitions of user's behavior prediction. While recent Transformer-based sequential recommender systems learn the attention matrix for each attribute to update item representations, the attention of a specific attribute is optimized by gradients from all input sources, leading to potential information mixture. Besides, Transformers mainly focus on intra-sequence attention for item attributes, neglecting cross-sequence relations and user attributes. Addressing these challenges, we propose the **Attr**ibute Trans**former** (AttrFormer) to learn attributes as explicit relations. This model transforms each type of attribute into an explicit relation defined in the feature space, and it ensures no information mixing among different input sources. Explicit relations introduce cross-sequence and intra-sequence relations. AttrFormer has novel relation-augmented heads to handle them at both the item and behavioral levels, seamlessly integrating the augmented heads into the multi-head attention mechanism. Furthermore, we employ position-to-position aggregation to refine behavior representation for users with similar patterns at the sequence level. To capture the subjective nature of user preferences, AttrFormer is trained using posterior targets where upcoming user behaviors follow a multinomial distribution with a Dirichlet prior. Our evaluations on four popular datasets, including Amazon (Toys & Games and Beauty) and MovieLens (1M and 25M versions), reveal that AttrFormer outperforms leading Transformer baselines, achieving around 20% improvement in NDCG@20 scores. Extensive ablation studies also demonstrate the efficiency of AttrFormer in managing long behavior sequences and inter-sequence relations.

KDD '25, August 3–7, 2025, Toronto, ON, Canada

ACM ISBN 979-8-4007-1245-6/25/08…$15.00
https://doi.org/10.1145/3690624.3709267

CCS CONCEPTS

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

KEYWORDS

User Modeling, Sequential Recommendation, Transformer, Attributes

ACM Reference Format:
Gang Liu, Fan Yang, Yang Jiao, Alireza Bagheri Garakani, Tian Tong, Yan Gao, and Meng Jiang. 2025. Learning Attribute as Explicit Relation for Sequential Recommendation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1 (KDD '25), August 3–7, 2025, Toronto, ON, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3690624.3709267

1 INTRODUCTION

Sequential recommendation aims to predict future user actions by analyzing their historical behaviors [22, 39]. Deep learning has advanced sequential recommender systems to capture extended patterns in user behavior sequences, moving beyond the Markov chain assumption [11] across various contexts, such as e-commerce product suggestions [15], music and movie ratings [12, 38]. Recently, Transformers have sparked significant research interest in sequential recommendation [6] and have been found to outperform convolutional neural networks (CNNs) and recurrent neural networks (RNNs)-based recommender models [20, 40].

User-item interactions occur with contextual information where factors from items, users, and interaction-specific contexts can affect user behaviors. We term these factors as *attributes*. It includes item information (price and brand), user demographics (age and occupation), and the specific context of the interaction, including timing and co-occurring items. While the conventional attention mechanism predominantly focuses on item IDs, the integration of these diverse attributes with behaviors presents a unique set of challenges [10, 25, 46]. We may blend all embeddings of item IDs

with attributes [6, 50] to obtain a single attention matrix, but this often results in lower-rank attention matrices, causing Transformers to fixate on irrelevant details [3, 46]. Therefore, recent studies have proposed learning a decoupled attention matrix. They attempt to achieve this by learning the embeddings for each attribute and item ID, calculating the attention matrices individually, and then combining these attention matrices [10, 46].

However, these methods [10, 46] may not effectively decouple information, leaving the low-rank attention issue unresolved. This arises because joint learning of attribute and item ID embeddings causes the optimization of attribute embeddings to be inadvertently influenced by gradients from other inputs. It leads to attention matrices for specific attributes mixing information from others.

Besides the attribute decoupling, it has been observed that the scope of self-attention is limited to a user's behavioral sequence, whereas attributes could act as connectors linking items across various sequences [10, 37]. This limitation in Transformers poses challenges for collaborative user modeling, as they often fail to capture the item-level cross-sequence relations reflected by attributes.

The third challenge is that the conventional Transformer learning paradigm struggles to accommodate the intricacies of diverse attributes, long user behavior sequences, and the dynamic nature of user preferences. Earlier research has focused on a specific attribute like timestamps [25] and item categories [46], and cannot address attributes from a broader spectrum of sources (e.g., users, items, behaviors) and types (e.g., categorical, numerical). Besides, certain approaches [10, 20] truncate long behavior sequences to predetermined lengths, potentially overlooking crucial user interactions. Furthermore, given the subjective nature of user preferences, influenced by factors like personality and mood shifts [4], the current Transformer is trained based on categorical probability and cross-entropy, fall short in representing uncertainties in user preferences, leading to potential overconfidence in predictions [19].

In this work, we propose the **Attr**ibute Trans**former** (AttrFormer), designed to learn from attributes as *explicit relations*, thereby addressing the aforementioned challenges. The explicit relation between pairwise behavior tokens decouples attribute semantics in the feature space, unlike jointly trained attribute embeddings that mix with other information and lack clear semantic meanings. In Figure 1, we train one of the latest Transformers for sequential recommendation [46] on different subsets of the Amazon Toys & Games dataset [20] with the item *category* attribute. We define the attribute's explicit relation as the dot product of two multi-hot encoding features, ensuring complete decoupled information from other input sources. We vary the sequence length and calculate the KL divergence between the learned attention and the explicit relation matrices. It is observed that the learned attention matrix is not aligned with the explicit relation, indicating the gradient descent algorithm cannot distinguish between the gradients for the *category* attribute and the item ID in backpropagation. The misalignment is more obvious with sparser user behavior data. It reveals the difficulty of learning an information-decoupled attribute embedding for the attention matrix in highly sparse behavior data. In contrast, our explicit relations are calculated by human understandable similarity functions without influence from other input sources and are also robust to data sparsity.

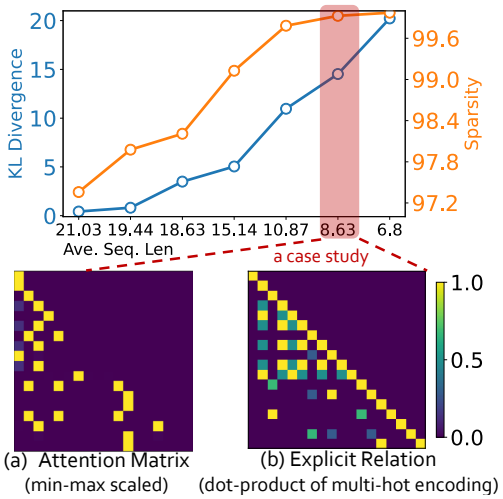


Figure 1: Comparing (a) learned attention [46] vs. (b) explicit relation (Proposed) under different levels of data sparsity sampled from Amazon Toys & Games for the *category* attribute. The average KL divergence between (a) and (b) is computed to analyze the matrix differences. Higher sparsity leads to larger divergence. Results indicate that learned *category* attention mixes with other information before final information fusion, potentially limiting the effectiveness of information fusion [46]. See appendix B.1 for setup details.

Explicit relations incorporate both intra- and cross-sequence relations, leading us to design relation-augmented multi-head attention within a batch for enhanced collaborative user modeling with Transformers. Item- and behavior-related explicit relations act as diverse queries to the keys in the attention mechanism, enriching the multi-head self-attention. *Multi-head attention effectively balances different attribute and self-attention subspaces to avoid conflicts in attribute relations* [51] when using multiple attributes. We conducted ablation studies in Figure 3 to examine the impact of different attributes. Moreover, we improve the efficiency of the relation-augmented heads with top-K relation selection and sparse matrix multiplication. For user attributes defined at the sequence level, we introduce a position-to-position aggregation module to smooth behavior representation across similar users.

To address the third challenge, we consider explicit relations from various attribute sources, including categorical and numerical values. We implement attribute-specific similarity/relation functions to preserve relation semantics for each attribute type. On the input side, we chunk the long user's behavior sequence instead of truncating to alleviate information loss for users. On the output side, we represent the uncertainty in user preferences by modeling the categorical probability of training objectives as a random variable and constructing the posterior training targets for model optimization. Specifically, we gather all training user behaviors to establish a prior Dirichlet distribution on item popularity and model the likelihood of users' future behaviors as following a Multinomial distribution. The contribution of Attrformer is summarized as:

- A novel architecture to handle various explicit and decoupled attribute relations from users, items, and behaviors.
- Improved collaborative Transformer learning through intra- and inter-sequence relation modeling in batches.
- Learning with a novel strategy that captures relations from categorical and numerical attributes, chunks arbitrarily long user behavior sequences with minimal information loss, and alleviates overconfidence in prediction by optimizing for new posterior optimization targets.
- Extensive experiments demonstrate that AttrFormer could achieve up to +20% NDCG@20 improvement in Amazon and Movielens recommendation scenarios, surpassing recent Transformers in sequential recommendation tasks.

2 RELATED WORK

2.1 Sequential Recommendation

The task predicts user future behaviors based on historical patterns [18, 36, 43, 53]. Early approaches used matrix factorization with first-order Markov chain [34, 35] to capture sequential patterns. Although higher-order Markov chains [14] were proposed to model sequential smoothness, they often overlooked longer dependencies among behaviors. Subsequent advances in deep learning explored CNNs [41, 48] and RNNs [17]. However, CNNs may overemphasize local features, while RNNs face challenges such as vanishing and exploding gradients. Self-attention in recommendation [20, 40] could automate relation learning in user sequences.

Recent advances in Transformers with self-attention mechanism showcased their effectiveness in the sequential recommendation, spanning different scenarios like self-supervision [45], multi-behavior [47], knowledge tracing [32], uncertainty attribution [2] and attribute-aware [10, 46, 50]. Transformers utilize item IDs as tokens, yet other modalities also play a crucial role in providing complementary information [49]. Combining inputs from multiple sources is beneficial, but existing solutions often introduce new low-rank issues in the attention matrix [3, 10, 46, 49].

2.2 Transformer for Attribute-rich Sequential Recommendation Data

Language models have greatly improved Transformers for language pre-training tasks [7, 28, 29, 42]. However, adapting Transformers from language models to recommendations is challenging due to modality differences. Earlier works like SASRec [20] and BERT4Rec [40] were proposed with single-direction and bidirectional self-attention, respectively, but they cannot address attributes. Attributes could improve sequential recommendation by mitigating data sparsity issues [6].The following works have designed Transformers for specific attributes: TiSASRec [25] incorporated the timestamp attribute, while SSE-PT [44] concatenated the user ID embedding with the item ID embedding. SASRecF [52] extended SASRec [20] and integrated ID and attribute embeddings in the input space [6, 50]. NOVA [27] separately fused the attribute embeddings and computed their attention matrix. Recently, DIF-SR [46] and MT4SR [10] decoupled the fusion of attributes in the attention matrix space from the query-key multiplication results. While intersample attention is introduced in [37] for sequence-level attention, it cannot effectively handle attributes and more fine-grained token-level information. In summary, existing works still have several key limitations in attribute decoupling, cross-sequence relation learning, and addressing broader attributes.

2.3 Learning Strategy of Transformer for Sequential Recommendation

Popular recommendation library RecBole [52] used behavior or interaction count-based sequence processing. However, extending this approach to larger datasets is not efficient due to the extensive computational complexity per epoch, which scales with the number of interactions. Alternatives like SASRec [20] and recent MT4SR [10] truncated user sequences to fixed lengths, efficiently reducing the complexity to the number of users but introducing challenges with missing previous behaviors.

Transformers could be trained in different ways. BERT4Rec [40] used masking prediction tasks for positive examples. Recently, next item prediction, where the subsequent item is treated as a positive target, has become more common [10, 20, 25, 46]. While early works [20, 25, 44] like SASRec used a single negative example to construct binary cross-entropy loss, recent research [10, 24] preferred cross-entropy, utilizing all non-next items in the training data as negative examples. However, in real interaction scenarios, users have hesitation and their final decisions are subjective and influenced by many random factors such as personality, mood, and change in behavior over time [4]. Therefore, a single behavior observation (i.e. label) may not reflect a correct user pattern and easily lead to over-confidence issues [19]. To represent the uncertainty in the label, we construct the training targets as a random variable following the Multinomial distribution in conjunction with a prior following the Dirichlet distribution.

3 THE PROPOSED ATTRFORMER

Given N_{user} users $\mathcal{U}$ and N_{item} items $\mathcal{X}$, the behavior sequence $\mathbf{x}_{1:T} = (x_1, x_2, \ldots, x_T)$ is formed as the user interactions with the items in chronological order. At any position $t \in [1, T-1]$, the next-item prediction task in sequential recommendation is $p(x_{t+1}|\mathbf{x}_{1:t})$.

The interaction matrix, with dimensions $N_{\text{user}} \times N_{\text{item}}$, often exhibits a high level of data sparsity, with nonzero entries typically less than 1% [20, 46]. Fortunately, user-item interactions provide valuable attribute data from users, items, and behavior itself. Suppose we have M attributes sequence $\mathbf{z}^{(m)}_{1:T} = (z^{(m)}_1, z^{(m)}_2, \ldots, z^{(m)}_T)$, where $m \in [1, M]$ is a specific attribute such as brand or price. We note that $M = M_u + M_i + M_b$, which indicates that the total number of attributes M comes from the source of users M_u, items M_i, and behaviors M_b. With all the items and attributes as input, we could build our model $f_\theta\left(\mathbf{x}_{1:t}; \{\mathbf{z}^{(m)}_{1:t}\}^M_{m=1}\right)$ to infer $p(x_{t+1}|\mathbf{x}_{1:t})$.

Our model f_θ has an input layer that explicitly converts M attribute inputs $\{\mathbf{z}^{(m)}_{1:t}\}^M_{m=1}$ to relations between items or users. It then stacks multiple encoder layers. Each encoder consists of a relation-augmented multi-head attention module, a feed-forward network module, and a position-to-position aggregation module (when user-related attributes are available). We present details about the architecture in Section 3.1 and its training method in Section 3.2, respectively. An overview is presented in Figure 2.

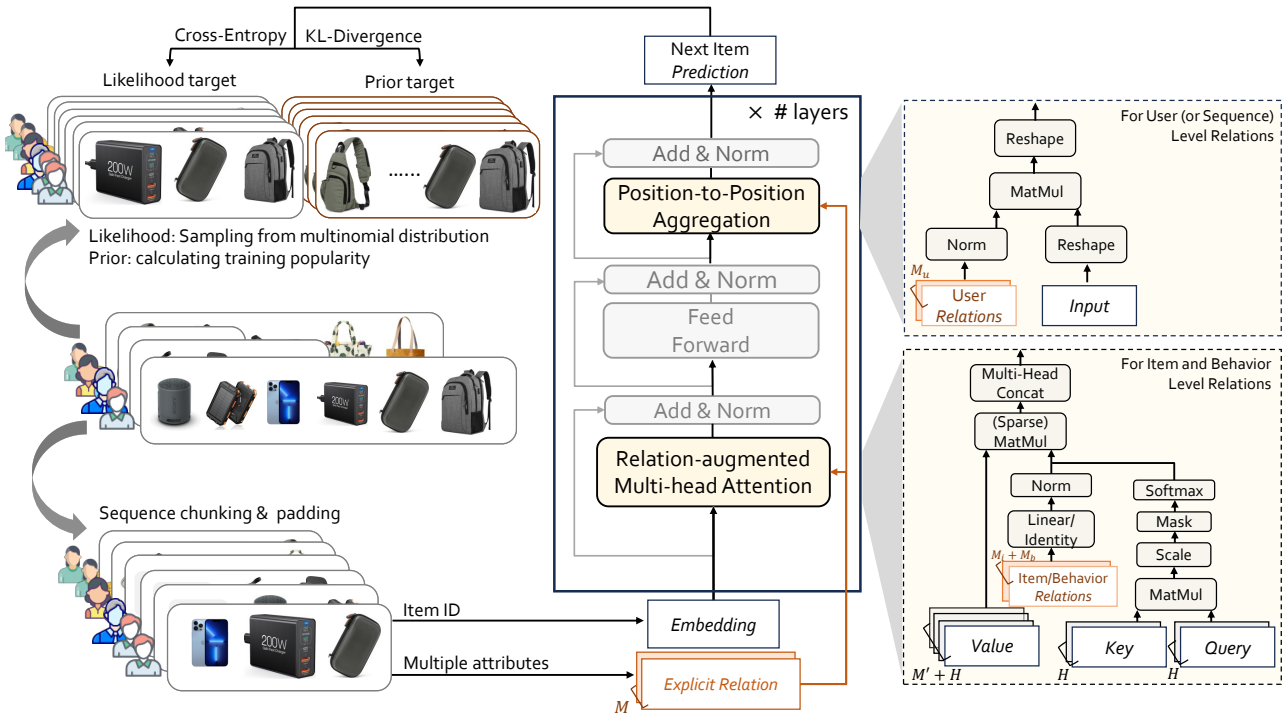


Figure 2: The training strategy and model architecture of AttrFormer. Bottom-left: AttrFormer uses sequence chunking and padding for varying-length behavior sequences of users. Item IDs transform into embeddings, while attributes are transformed into explicit relations. Right side: AttrFormer has multiple encoder layers. Each layer has a relation-augmented multi-head attention module to handle item- and behavior-level relations and an aggregation module to handle user-level relations. Top-left side: AttrFormer has novel posterior targets for model optimization, consisting of likelihood and prior components.

3.1 Model Architecture

3.1.1 Input Layer with Explicit Relation. Given T-length sequence, we initialize a learnable embedding table $\mathbf{E} \in \mathbb{R}^{N_{\text{item}} \times D}$, where D is the hidden dimension. We retrieve the items from $\mathbf{x}_{1:T}$ to obtain the embedding matrix $\mathbf{X} \in \mathbb{R}^{T \times D}$, where $\mathbf{X}_i = \mathbf{E}_{x_i}$ for the sequence. Unlike prior work [20, 46], we do not initialize embedding tables for attributes. To explicitly keep the attribute relations decoupled, we directly apply a similarity function to the feature vectors of attributes. Concretely, given any attribute $\mathbf{z}_{1:T}^{(m)}$, we have its feature matrix $\mathbf{Z}^{(m)} \in \mathbb{R}^{(B \cdot T) \times d_m}$, where B denotes the batch size and d_m is the feature dimension, we define the explicit relation between item i and item j from the attribute m as follows:

$$r_{ij}^{(m)} = r_{ji}^{(m)} = S^{(m)}(\mathbf{Z}_i^{(m)}, \mathbf{Z}_j^{(m)}) \geq 0. \tag{1}$$

S represents a specific similarity function for an attribute m. Therefore, we have explicit relations $\{\mathbf{R}^{(i)}\}_{i=1}^{M_i+M_b+M_u}$

3.1.2 Relation-augmented Multi-head Attention. For each encoder layer, we assume that the hidden representation of a sequence of items is $\mathbf{H} \in \mathbb{R}^{T \times D}$, where $\mathbf{H}$ is initialized by item embedding $\mathbf{X}$. Self-attention (SA) is defined on the query, key, value $(\mathbf{Q}, \mathbf{K}, \mathbf{V}) \in \mathbb{R}^{T \times d}$ matrices with dimension d, which are transformed from $\mathbf{H}$ with different parameters $(\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V) \in \mathbb{R}^{D \times d}$:

$$\text{SA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d}}\right)\mathbf{V}, \tag{2}$$

$$\text{where } \mathbf{Q} = \mathbf{H}\mathbf{W}^Q, \mathbf{K} = \mathbf{H}\mathbf{W}^K, \mathbf{V} = \mathbf{H}\mathbf{W}^V. \tag{3}$$

Here, $\mathbf{Q}\mathbf{K}^T$ defines the learnable relation. Following the previous work [20], we apply the causal (or the left-to-right) attention mask on self-attention. For the output of multi-head self-attention (MSA), we concatenate H attention heads as follows:

$$\text{MSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}^{(1)}, \text{head}^{(2)}, \ldots, \text{head}^{(H)}), \tag{4}$$

$$\text{where head}^{(i)} = \text{SA}(\mathbf{Q}^{(i)}, \mathbf{K}^{(i)}, \mathbf{V}^{(i)}). \tag{5}$$

Sparse data make it difficult for multi-head attention to learn meaningful relations, and the limited receptive field of self-attention within a single sequence restricts user collaborative modeling. We overcome these limitations with the relations $\{\mathbf{R}^{(i)} \in \mathbb{R}^{BT \times BT}\}_{i=1}^{M_i+M_b}$ within the batch with size B. Since the matrices define inter- and intra-sequence relations, we first reshape the value $\mathbf{V}$ within batch into a two-dimensional matrix $\mathbf{V}' \in \mathbb{R}^{BT \times d}$. Attention can be described as the mapping from a query to a key-value pair [42], where the relations in $\mathbf{R}^{(i)}$ naturally depict the query's results to the key according to the attribute. Hence, a relation-augmented attention

head for an attribute is:

$$\text{RAH}(\mathbf{R}^{(i)}, \mathbf{V}) = \text{Norm}\left(\mathbf{R}^{(i)}\right)\mathbf{V}'. \tag{6}$$

Explicit relations do not depend on the parameterized linear layer of the query and the key. It eliminates the need for the scaling term $\frac{1}{\sqrt{d}}$. As the relation may be symmetric and non-negative, we prefer more appropriate normalization methods to preserve the symmetric property of $\mathbf{R}$, as suggested by [30]. A choice of symmetric normalization is $\tilde{\mathbf{D}}^{-1/2}\mathbf{R}^{(i)}\tilde{\mathbf{D}}^{-1/2}$, where $\tilde{\mathbf{D}}$ is the diagonal matrix with the same shape as the $\mathbf{R}^{(i)}$. Each row in $\tilde{\mathbf{D}}$ is the number of relations an interacted item has. With RAH, we could calculate:

$$\mathbf{H}^{RAH} = \text{RAH}\left(\mathbf{R}^{(1:M_i+M_b)}, \mathbf{V}\right), \tag{7}$$

$$\text{where } \mathbf{R}^{(1:M_i+M_b)} = \text{Concat}(\mathbf{R}^{(1)}, \ldots, \mathbf{R}^{(M_i+M_b)})\mathbf{W}^R, \tag{8}$$

where $\mathbf{R}^{(i:M_i+M_b)} \in \mathbb{R}^{(M' \times BT \times BT)}$ is obtained by concatenating all $M_i + M_b$ relations along the first dimension, followed by applying $\mathbf{W}^R \in \mathbb{R}^{(M_i+M_b)\times M'}$ to regulate the number of augmented heads M' in the standard self-attention mechanism. Note that $\mathbf{H}^{RAH} \in \mathbb{R}^{B\times T\times (M'd)}$ is obtained after applying permutation and reshaping operations to the output of RAH. To integrate with standard multi-head attention, $\mathbf{H}^{MSA} \in \mathbb{R}^{B\times T\times (Hd)}$ is calculated for each sequence in the batch using Equation 4, subsequently obtaining the new hidden representation of items $\mathbf{H}'$ in the batch:

$$\mathbf{H}' = \text{Concat}(\mathbf{H}^{MSA}, \mathbf{H}^{RAH})\mathbf{W}^O, \tag{9}$$

where $\mathbf{W}^O \in \mathbb{R}^{(Hd+M'd)\times D}$ is a linear layer to balance the attention between self-attention and relation-augmented heads.

A direct implementation of Eq. (7) may not be efficient and effective given the potential order of magnitude of total relations (B^2T^2) that stem from multiple attributes. This could lead to the inclusion of redundant and noisy relations. Therefore, we sparsify the redundant relations to remove noise and make the implementation of Eq. (7) to be efficient. For each type of relation (attribute), we select the top-K relations based on the number calculated in Eq. (1) for each item, thereby retaining only highly correlated items. Suppose that the total number relations after top-K filtering is E, we efficiently reduce the computational complexity from $O(B^2T^2)$ to $O(E)$ ($E \ll B^2T^2$) with sparse matrix operation. We empirically study the efficiency in Section 4.4 and appendix B.6.

3.1.3 Position-to-Position Aggregation. Recommendation data often include user attributes such as *age* and *gender*, leading to user-level relations in $\{\mathbf{R}^{(i)}\}_{i=M_i+M_b+1}^{M_i+M_b+M_u}$. To address this, we implement a position-to-position aggregation operation, smoothing the representation of items interacted with by similar users. We first apply average pooling to aggregate multiple relations for M_u attributes into $\mathbf{R}^U \in \mathbb{R}^{B\times B}$. Next, $\mathbf{R}^U$ is normalized as per Eq. (6) and multiplied with the flattened hidden state $\text{Flatten}(\mathbf{H}) \in \mathbb{R}^{B\times TD}$, yielding the updated representation $\mathbf{H}' = \text{Norm}(\mathbf{R}^U) \cdot \text{Flatten}(\mathbf{H})$. Finally, we reshape $\mathbf{H}'$ to $\mathbb{R}^{B\times T\times D}$.

3.2 Learning Strategy of AttrFormer

3.2.1 Handling various attributes for explicit relations. AttrFormer takes into account both categorical and numerical attributes. Moreover, AttrFormer can readily incorporate a wider range of relations from images and texts using pre-trained vision and language models [7, 13]. The categorical attribute indicates that $\mathbf{Z}_i$ is one-hot or multi-hot encoding with $d_m > 0$ and the numerical attribute indicates that $\mathbf{Z}_i \in \mathbb{R}$. Their similarity function $S^{(m)}$ defined by Eq. (1) may not be the same and should align with real-world assumptions about user behavior patterns. Specifically, we use the dot-product as $S^{(m)}$ for categorical attribute and the exponential of negative absolute difference as $S^{(m)}$ for numerical attributes.

$$\begin{cases} S^{(m)} & = \mathbf{Z}_i^{(m)} \cdot \left(\mathbf{Z}_j^{(m)}\right)^T, \text{ if } \mathbf{Z}^{(m)} \text{ is categorical,} \\ S^{(m)} & = \exp(-\frac{|\mathbf{Z}_i^{(m)} - \mathbf{Z}_j^{(m)}|}{r_{\text{norm}}}), \text{ if } \mathbf{Z}^{(m)} \text{ is numerical.} \end{cases} \tag{10}$$

r_{norm} represents a scaling factor for the relative distance between numerical attribute features. Details can be found in Section A.2.

3.2.2 Sequence chunking. The computational complexity of self-attention is $O(T^2)$, which significantly limits its ability to model very long behavior sequences. Although recent advances in low-ranking estimation have reduced the complexity to $O(N_{\text{interest}}T)$ [9], where N_{interest} represents the number of latent user interests, this approach still lacks flexibility to handle arbitrary long sequences, which is common in practical scenarios (see Table 1). As a result, recommendation Transformers like SASRec [20] and MT4SR [10] risk losing useful information through sequence truncation, where many previous user behaviors are omitted due to fixed-length constraints. In this work, we chunk long sequences into multiple fixed-length subsequences with cross-sequence relations. While standard attention captures short-term user behavior patterns within a sequence, it struggles to learn the relation across different sequences. AttrFormer incorporates cross-sequence relations, enabling self-attention to learn from similar behaviors in different chunked behavior sequences and addressing the concern.

3.2.3 Optimization with posterior target. The standard choice of the next item x_{t+1} prediction uses cross-entropy (CE) to maximize the likelihood for $q_\theta(x_{t+1}|x_{1:t})$ with the item embedding $\mathbf{E}$.

$$\text{CE}\left(p(x_{t+1}), p_\theta(x_{t+1}|\mathbf{x}_{1:t})\right) = -\log q_\theta(x_{t+1}|\mathbf{x}_{1:t}), \tag{11}$$

$$\text{where, } q_\theta(x_{t+1}|\mathbf{x}_{1:t}) = \frac{\exp\left(f_\theta(\mathbf{x}_{1:t})\mathbf{E}_{x_{t+1}}\right)}{\sum_{x'\in\mathcal{X}}\exp\left(f_\theta(\mathbf{x}_{1:t})\mathbf{E}_{x'}\right)}. \tag{12}$$

However, a single next-item observation (i.e., x_{t+1}) may not fully uncover the underlying factors of user preferences. From a probabilistic perspective, most existing work [1, 31] considers posterior estimation with the Gaussian prior, which is unimodal and cannot reflect the diversity of user preference in different recommendation scenarios. In this work, we sample the next item x_{t+1} from a multinomial distribution based on the user's future preference $\mathbf{x}_{>t}$ and use the Dirichlet distribution as the prior. The Dirichlet distribution, a multivariate generalization of the Beta distribution, serves as the conjugate prior of the multinomial distribution. It effectively models prior knowledge about user preferences, such as item popularity, particularly when user behavior data are sparse. Based on it, the posterior learning target is obtained by balancing the prior and likelihood. Specifically, we assume that the user preference on the next item x_{t+1} follows a multinomial distribution with parameters $\mathbf{p} = (p_1, p_2, \ldots, p_{N_{\text{item}}})$:

$$x_{t+1} \sim \text{Multinomial}(\mathbf{p}), \tag{13}$$

Table 1: Statistics of four datasets with high sparsity.

	Amazon		MovieLens	
	Beauty	Toys & Games	1M	25M
# User	22,364	19,413	6,041	162,542
# Item	12,102	11,925	3,417	32,721
# Interaction	198,502	167,597	999,611	24,945,870
Sparsity (%)	99.93	99.92	95.16	99.53
Ave. Seq. Len.	8.87	8.63	165.50	153.47
Max. Seq. Len.	204	550	2,277	22,348
# Item Attr.	3	3	1	1
# Behavior Attr.	5	5	1	1
# User Attr.	0	0	3	3

It becomes the categorical distribution [20, 46] in many cases. Nonetheless, we opt for the Multinomial distribution to accommodate generalized scenarios. Then, we formulate the prior distribution of user preference with the Dirichlet distribution $\mathrm{Dir}(N_{\text{item}}, \boldsymbol{\alpha} = \boldsymbol{c}+1)$, where $\boldsymbol{c}$ is the count of items preferred by all users. Therefore, the prior discrete probability distribution over all items is

$$\mathbf{p} = \{p_1, p_2, \ldots, p_{N_{\text{item}}}\} \sim \mathrm{Dir}(N_{\text{item}}, \boldsymbol{\alpha}). \tag{14}$$

When user's future behaviors are available, we have the posterior $\mathrm{Prob}(\boldsymbol{\alpha}|\mathbf{x}_{>t}) \propto \mathrm{Prob}(\boldsymbol{\alpha})\,\mathrm{Prob}(\mathbf{x}_{>t}|\boldsymbol{\alpha})$. The log of the posterior distribution could be formulated by prior and likelihood as follows.

$$\log \mathrm{Prob}(\boldsymbol{\alpha}|\mathbf{x}_{>t}) \propto \log \mathrm{Prob}(\boldsymbol{\alpha}) + \log \mathrm{Prob}(\mathbf{x}_{>t}|\boldsymbol{\alpha}) \tag{15}$$

$$\propto \sum_{j=1}^{N_{\text{item}}} (\alpha_j - 1) \log p_j + \sum_{x_i \in \mathbf{x}_{>t}} \log p_{x_i}. \tag{16}$$

Details of the derivation can be found in appendix A.1. Here, the term $\alpha_j - 1$ could be interpreted as the regularization weight of the prior for item j when making the prediction. We may adjust p_{x_i} to balance between prior and posterior. Note that the setting $p_{x_i} = 1$ cannot maximize the posterior for our task. Because we focus on next-item prediction, rather than enumerating predictions for all possible future behaviors of the user. To represent our belief in the relatedness of future items to the next item prediction task, we use a monotonically decreasing function $p_{x_i} = \beta(i)$, where $i > t$, to control the posterior update. Specifically, $p_{x_{t+1}} = \beta(t+1) = 1$ because x_{t+1} is the ground-truth observation for the next item. When we make $\beta(i) = 0$ for $\forall i > t+1$. We get the special case of Eq. (11). To implement the posterior target, we find that separating the prior with KL-divergence (KL) and likelihood with cross-entropy achieves the best empirical performance.

$$\mathcal{L} = \mathrm{KL}(\mathbf{p}_{\text{prior}}|\mathbf{q}_\theta(\mathbf{x}_{1:t})) + \sum_{i \in \mathbf{x}_{>t}} \beta(i)\,\mathrm{CE}(p(x_i)|p_\theta(x_i|\mathbf{x}_{1:t})) \tag{17}$$

where $\mathbf{q}_\theta(\mathbf{x}_{1:t}) = \mathrm{Softmax}(f_\theta(\mathbf{x}_{1:t}) \cdot \mathbf{E}^\top)$ is the predicted distribution. We use the expectation of the prior $\mathrm{Dir}(N_{\text{item}}, \boldsymbol{\alpha})$ for $\mathbf{p}_{\text{prior}}$.

4 EXPERIMENTS

We conduct experiments to demonstrate the effectiveness of AttrFormer with a few research questions. **RQ1:** How it performs on sequential recommendation tasks compared to the state-of-the-art recommender Transformer? **RQ2:** Where does the effectiveness come from? **RQ3:** Is the model robust to different sequence lengths with sequence chunking and cross-sequence relation?

4.1 Experiment Setup

4.1.1 Datasets. Data statistics are in Table 1.

- **Amazon**: Following previous work [20, 46], we consider two top-level product categories collected from *Amazon.com* due to their significantly sparse user behavior data (over 99%): Beauty dataset and Toys & Games dataset.Amazon datasets have attributes: *price*, *timestamp*, *brand*, *categories*, *also_viewed*, *also_bought*, *bought_together*, *buy_after_viewing*;
- **Movie**: We include two larger-scale datasets from the well-known MovieLens datasets [12]. The first version is MovieLens-1M, which includes rich user categorical user attributes: *age*, *gender*, *occupation*. The second version has around 25M user-item interactions: MovieLens-25M, which is used to evaluate methods on large-scale datasets. Both datasets have attributes: *genre*, *timestamp*.

We consider the *position* as a special attribute [46]. We filter out users and items that have less than 5 interactions following [10, 46].

4.1.2 Baselines. Baseline methods broadly include: (1) GRU4Rec [16]; (2) the contrastive model DuoRec [33]; (3) Transformers tailored for sequential recommendation: SASRec [20], BERT4Rec [40], CL4SRec [45], LightSANs [9], FEARec [8]; and (4) Transformers that process attributes: TiSASRec [25], SASRecF [50, 52], MT4SR [10], DIF-SR [46]. Baseline details are in appendix B.2.

4.1.3 Data Splitting and Model Evaluation. We split the sequential data following [20, 46]: we reserve the last two interacted item per user as validation and test data. The remaining behaviors in the sequence are used for model training. Evaluation is conducted using Recall and Normalized Discounted Cumulative Gain (NDCG) scores for top-5 and top-20 predicted items [46]. To avoid sampling bias, negative samples are not used for evaluation; instead, we rank all items and select the top prediction, as suggested by [5, 23].

4.1.4 Implementation. Our experiments are conducted five times on two Amazon datasets and MovieLens-1M, reporting only the mean values due to negligible standard deviation. We also assess model performance on the larger MovieLens-25M dataset with a single run, using a fixed sequence length of 50. To manage computational resources, we compare only efficient models selected from the Amazon Toys & Games dataset (Table 4). The $\beta(\cdot)$ function in Eq. (17) is defined as $\beta(t+1) = 1$, $\beta(t+2) = 0.25$, and $\beta(i) = 0$ for $i > t+2$. Early stopping is based on the sum of recall and mean reciprocal rank. AttrFormer and most baselines are implemented and evaluated using a unified pipeline [52] for fair and comprehensive evaluations. Details are in appendix B.3.

4.2 RQ1: Performance Analysis

From Table 2, AttrFormer consistently outperforms other sequential recommender models across four high-sparsity datasets. A significant improvement of over 15% in NDCG@20 is observed for the Amazon Toys & Games dataset, along with greater than 15% improvement across all metrics for the MovieLens-1M dataset. Specifically, we have the following observations,

(1) Transformers outperform recurrent neural network (RNN) based GRU4Rec. An exception is BERT4Rec, where our findings

Table 2: Average value over five runs. Standard deviations < 0.001, not reported. MT4SR is the same as SASRec on MovieLens (as there is no behavior attribute). AttrFormer (Ours) in bold. Best mean bolded, second-best underlined.

Dataset	Metric			Transformer: N.A. for Attribute					Transformer: With Attribute Input					Improve (+%)
		GRU4Rec	DuoRec	SASRec	BERT4Rec	CL4SRec	LightSANs	FEARec	TiSASRec	SASRecF	MT4SR	DIF-SR	**AttrFormer**	
Amazon Beauty	Recall@5	0.0349	0.0642	0.0556	0.0382	0.0392	0.0561	0.0594	0.0576	0.0587	0.0559	0.0578	**0.0657**	9.4
	Recall@20	0.0817	0.1132	0.1107	0.0783	0.0742	0.1222	0.1239	0.1244	0.1231	0.1169	0.1273	**0.1324**	4.0
	NDCG@5	0.0231	0.0330	0.0343	0.0265	0.0217	0.0342	0.0337	0.0344	0.0413	0.0360	0.0337	**0.0446**	8.0
	NDCG@20	0.0362	0.0447	0.0540	0.0378	0.0296	0.0528	0.0520	0.0534	0.0594	0.0533	0.0535	**0.0639**	7.6
Amazon Toys & Games	Recall@5	0.0271	0.0651	0.0600	0.0364	0.0324	0.0632	0.0674	0.0666	0.0585	0.0607	0.0675	**0.0720**	6.7
	Recall@20	0.0654	0.086	0.1073	0.0691	0.0595	0.1273	0.1297	0.1325	0.1217	0.1148	0.1342	**0.1357**	1.1
	NDCG@5	0.0175	0.0339	0.0435	0.0265	0.0183	0.0370	0.0379	0.0379	0.0393	0.0410	0.0380	**0.0501**	15.2
	NDCG@20	0.0368	0.0392	0.0570	0.0356	0.0244	0.0552	0.0557	0.0566	0.0571	0.0563	0.0569	**0.0681**	19.3
MovieLens 1M	Recall@5	0.1752	0.1477	0.1854	0.1341	0.1395	0.1840	0.1372	0.1816	0.1829	-	0.1518	**0.2258**	21.8
	Recall@20	0.3579	0.2538	0.3483	0.2728	0.2284	0.3590	0.3097	0.3558	0.3553	-	0.3195	**0.4128**	16.0
	NDCG@5	0.1172	0.0947	0.1285	0.1120	0.0535	0.1226	0.8285	0.1216	0.1239	-	0.0964	**0.1554**	20.9
	NDCG@20	0.1687	0.1638	0.1745	0.1311	0.0990	0.1725	0.1320	0.1711	0.1726	-	0.1440	**0.2088**	19.7

Table 3: Model performance on the large scale MovieLens dataset. AttrFormer in bold. Best mean bolded, second-best underlined. The seven most efficient Transformer-based models (from Table 4) are compared, while others may use excessive time/resources.

Dataset	Metric		Transformer: N.A. for Attribute			Transformer: With Attribute Input				Improve (+%)
		GRU4Rec	SASRec	BERT4Rec	LightSANs	TiSASRec	SASRecF	DIF-SR	**AttrFormer**	
MovieLens 25M	Recall@5	0.1713	0.1848	0.1651	0.1723	0.1802	0.1830	0.1443	**0.1976**	6.9
	Recall@20	0.3272	0.3484	0.3132	0.3295	0.3440	0.3318	0.2927	**0.3641**	4.5
	NDCG@5	0.1213	0.1287	0.1170	0.1193	0.1240	0.1310	0.0973	**0.1394**	6.4
	NDCG@20	0.1654	0.1751	0.1589	01639	0.1706	0.1733	0.1393	**0.1867**	6.6

Table 4: Comparison of training time per epoch on Amazon Toys & Games shows that AttrFormer efficiently handles up to eight attributes in just a few seconds. Models above the dashed line are efficient, and their effectiveness is further compared on MovieLens-25M in Table 3.

Method	Avg. Time (s)	# Processed Relation	Hidden Size	# Layer
SASRec	1.4	0	256	3
SASRecF	2.1	2	256	3
DIF-SR	5.0	2	256	4
AttrFormer	6.5	8	256	4
LightSANs	8.0	0	256	3
TiSASRec	9.3	1	256	4
BERT4Rec	15.4	0	256	3
FeaRec	19.4	0	256	2
MT4SR	79.3	4	128	1
DuoRec	171.6	0	64	2
CL4SRec	265.0	0	64	2

align with prior studies [26, 46] under unsampled evaluation settings. An important reason is that the randomly masked item prediction tasks used in BERT4Rec are not effective in modeling users' future behaviors as they may not follow the distribution in Eq. (13).

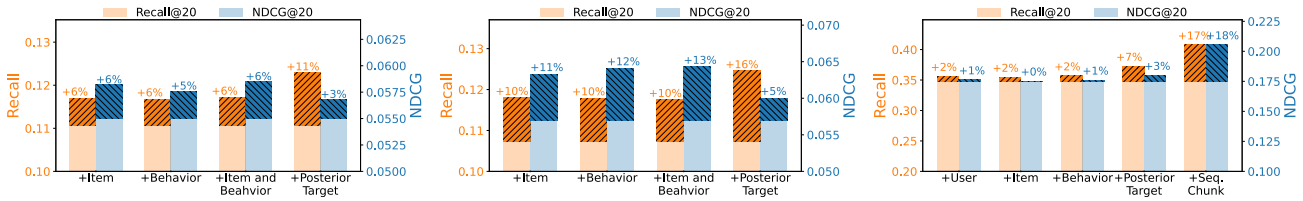
(2) In nine out of sixteen rows in Table 2, attribute-aware Transformers outperform those that cannot handle attributes. This trend is notable in the two Amazon datasets, characterized by data sparsity exceeding 99.9%. Rich attributes are crucial in enhancing the training of Transformers in this situation. However, previous baseline methods [10, 46, 50] often focus solely on either item-related [46] or behavior-related [10] aspects. AttrFormer is capable of harnessing multiple attributes as intra- and cross-sequence relations, thereby contributing to its remarkable performance.

(3) Existing Transformers do not consistently benefit from attributes for several reasons. First, movie genres often contain redundant noise. For example, a movie may incorporate conflicting elements like comedy and thriller, but audiences might only lean towards one of them. We alleviate this by selecting the top-k genres, but the existing learning-based baselines [10, 46, 50] learn attention from attributes and hardly denoise when the user behavior data are sparse. Second, attributes like *category* are more fine-grained with over 200 dimensions or subcategories on Amazon datasets, while movie genres only encompass around 20 dimensions.

(4) AttrFormer outperforms existing Transformers across different scales of datasets from thousands to millions. Its strengths stem from multi-aspects: various explicit relations in AttrFormer correspond to distinct subspaces in the multi-head attention, while cross-sequence relations in sequence chunking adapt to variable sequence lengths, enabling attention across related sequences.

4.3 RQ2: Ablation Studies

We perform comprehensive ablation studies on attributes, sequence length, and posterior targets. For Amazon datasets, we classify



(a) Amazon Beauty (b) Amazon Toys & Games (c) MovieLens-1M

Figure 3: For ablation studies, SASRec (shallow bar) serves as the starting point. We add various attributes (deep slashed bar) for ablation. Amazon datasets (up to 8 attributes) are split into item and behavior categories. We assess their combination and posterior target loss. Sequence chunking's impact on Amazon datasets is minimal (Avg. Seq. Len. < 10 as in Table 1). MovieLens-1M is used to examine user attributes and sequence chunking effects.

attributes as follows: item-related categorical attributes (*brand*, *categories*), item-related numerical attributes (*price*), behavior-related categorical attributes (*also_viewed*, *also_bought*, *bought_together*, *buy_after_viewing*), and behavior-related numerical attributes (*timestamp*). For MovieLens-1M, we classify attributes as follows: user-related attributes (*age*, *gender*, *occupation*), item-related attribute (*genre*), behavior-related attribute (*timestamp*). We conduct ablation studies for different attributes, the posterior training targets, and the sequence chunking. Results are presented in Figure 3. We note that removing the effect of attributes, which means eliminating corresponding modules in AttrFormer, also reflects the impact of different model components.

In Amazon datasets, Figure 3 reveals that item- or behavior-related attributes improve NDCG scores, while posterior targets with training popularity prior contribute to higher recall in AttrFormer. In particular, leveraging either item-related or behavior-related attributes results in a relative performance boost of +5% for Amazon Beauty and +11.5% for Amazon Toys & Games. Their combined performance improvement is not linearly scaled. This suggests that different attributes might capture overlapping relations. Hence, further research into a deeper understanding of the interplay between attribute relations holds promise. As we compute item popularity as the prior optimization target, AttrFormer is inclined to suggest popular items to users when their behavior data are sparse. This improves recall performance for the model. More fine-grained results are in appendix B.5. In Figure 3(c), user, item, and behavior attributes improve performance by 1-2%, with sequence chunking having a greater impact. User attributes in MovieLens are limited to age, gender, and occupation, but more extensive data in proprietary e-commerce can amplify the impact of position-to-position aggregation.

4.4 RQ3: Sensitivity and Efficiency Analysis

We study the impact of behavior sequence length, top-K selection in the relation-augmented heads, and the batch size in this subsection.

We study the impact of sequence length on the MovieLens-1M dataset. We vary the sequence lengths in a wide range: {10, 25, 50, 75, 100, 125, 150}. The test performance NDCG@20 and Recall@20, are shown in Figure 4. Usually, training with longer behavior sequences incurs higher computational memory costs and time costs. Baselines often require longer sequences to achieve their best performance, which occurs around 125 to 150 in our study. Compared to baselines, AttrFormer has robust performance when varying sequence lengths, particularly when the sequence length is set to 10. These findings demonstrate the advantages of AttrFormer with sequence chunking.

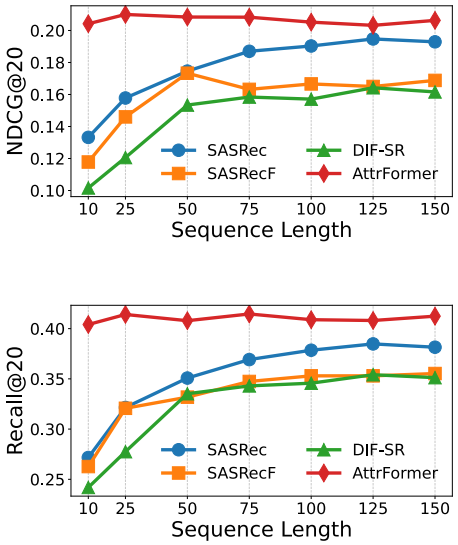


Figure 4: Sensitivity analysis for sequence length on the MovieLens-1M dataset. Training of AttrFormer is robust to diverse sequence lengths.

Table 5: Efficiency analysis for the ratio between the number of sparse relation E and the number of full relations, which has the magnitude of B^2T^2.

	Amazon Beauty			Amazon Toys & Games		
top-K	1	10	100	1	10	100
$E/(B^2T^2)$	7e-4	7.5e-3	5.7e-2	5.2e-4	5.0e-3	3.8e-2

We vary top-K and report results on model performance in Figure 5. More relations do not necessarily yield better performance due to noise; AttrFormer proves robust across a wide range of top-K

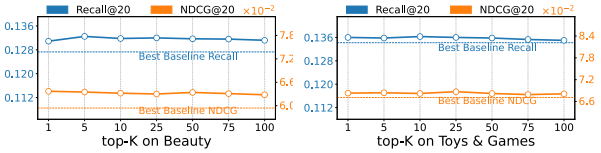


Figure 5: Sensitivity analysis of top-K vs. test performance on Amazon datasets.

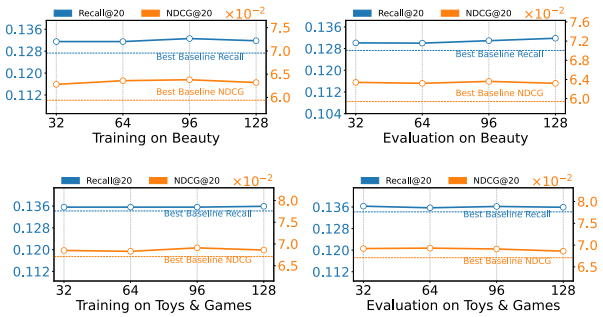


Figure 6: Sensitivity analysis of batch size (training or evaluation) vs. test performance for cross-sequence relations.

choices. Efficiency analysis was carried out on the number of sparse relations E with a ratio $E/(B^2T^2)$, where B represents the batch size and T denotes the sequence length. The results are in Table 5, revealing significant computational time savings through top relation selection, as $E \ll B^2T^2$. A lower ratio indicates that we zero-out relatively noiser relations, thereby accelerating the computation speed and reducing the computational memory required to obtain the hidden state $\mathbf{H}^{RAH}$ from Eq. (7).

We compare the training time per epoch of different models in Table 4. This comparison demonstrates the efficiency of AttrFormer in handling up to eight attributes, considering that all methods use comparable model sizes. Among the models, SASRec has the shortest training time as it doesn't consider any attributes. AttrFormer takes approximately 6.53 seconds for eight relations, while MT4SR requires about 79.29 seconds for four relations. Note that MT4SR needs to convert these attributes into relations during each training epoch [10, 25], which reduces efficiency. In contrast, explicit relations come from the attribute feature space and allow preprocessing before training, thereby improving training model efficiency. For more details, refer to appendix B.6.

We investigate the impact of training and evaluation batch sizes on cross-sequence relations using Amazon datasets. Results are in Figure 6. It indicates that the explicit relations remain robust across various batch sizes during training and inference. We explore an extreme case in which explicit relations are solely used during training, but not during inference. On the Beauty dataset, the test performance is 0.1256 (0.0613) for Recall@20 (NDCG@20). Even without relations, AttrFormer achieves competitive performance compared to the best baseline methods in Table 2.

5 CONCLUSIONS

In this work, we improved the Transformer by explicit relations from attributes. We proposed AttrFormer along with a novel training strategy. First, AttrFormer was flexible in handling diverse attributes from users, items, and behaviors. Second, AttrFormer utilized token-level relation-augmented heads and a sequence-level aggregation module. It enabled attention to related sequences and overcame the limitation of traditional Transformers, which had attention only within a user sequence. Third, we proposed posterior targets to train AttrFormer, which treated users' future behavior as a random variable and modeled uncertainty in user preferences. Extensive experiments on Amazon and MovieLens datasets demonstrated the advantages of AttrFormer in both effectiveness and efficiency for sequential recommendation tasks.

ACKNOWLEDGMENTS

This work was supported by NSF IIS-2142827, IIS-2146761, IIS-2234058, CBET-2332270, and ONR N00014-22-1-2507.

REFERENCES

[1] Omer Deniz Akyildiz, Gerrit van den Burg, Theodoros Damoulas, and Mark Steel. 2021. Probabilistic sequential matrix factorization. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 3484–3492.

[2] Carles Balsells-Rodas, Fan Yang, Zhishen Huang, and Yan Gao. 2024. Explainable Uncertainty Attribution for Sequential Recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2401–2405.

[3] Pu-Chin Chen, Henry Tsai, Srinadh Bhojanapalli, Hyung Won Chung, Yin-Wen Chang, and Chun-Sung Ferng. 2021. A simple and effective positional encoding for transformers. *arXiv preprint arXiv:2104.08698* (2021).

[4] Victor Coscrato and Derek Bridge. 2023. Estimating and evaluating the uncertainty of rating predictions and top-n recommendations in recommender systems. *ACM Transactions on Recommender Systems* 1, 2 (2023), 1–34.

[5] Alexander Dallmann, Daniel Zoller, and Andreas Hotho. 2021. A case study on sampling strategies for evaluating neural sequential item recommendation models. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 505–514.

[6] Gabriel de Souza Pereira Moreira, Sara Rabhi, Jeong Min Lee, Ronay Ak, and Even Oldridge. 2021. Transformers4rec: Bridging the gap between nlp and sequential/session-based recommendation. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 143–153.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Xinyu Du, Huanhuan Yuan, Pengpeng Zhao, Jianfeng Qu, Fuzhen Zhuang, Guanfeng Liu, Yanchi Liu, and Victor S Sheng. 2023. Frequency enhanced hybrid attention network for sequential recommendation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 78–88.

[9] Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. 2021. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 1733–1737.

[10] Ziwei Fan, Zhiwei Liu, Chen Wang, Peijie Huang, Hao Peng, and S Yu Philip. 2022. Sequential Recommendation with Auxiliary Item Relationships via Multi-Relational Transformer. In *2022 IEEE International Conference on Big Data (Big Data)*. IEEE, 525–534.

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

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

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[14] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th international conference on data mining (ICDM)*. IEEE, 191–200.

[15] 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*. 507–517.

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

[17] Balázs Hidasi, Massimo Quadrana, Alexandros Karatzoglou, and Domonkos Tikk. 2016. Parallel recurrent neural network architectures for feature-rich session-based recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 241–248.

[18] Yang Jiao, Fan Yang, Yetian Chen, Yan Gao, Jia Liu, and Yi Sun. 2024. Rethinking sequential relationships: Improving sequential recommenders with inter-sequence data augmentation. In *Companion Proceedings of the ACM on Web Conference 2024*. 641–645.

[19] Taejong Joo, Uijung Chung, and Min-Gwan Seo. 2020. Being bayesian about categorical probability. In *International conference on machine learning*. PMLR, 4950–4961.

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

[21] Nicolas Keriven. 2022. Not too little, not too much: a theoretical analysis of graph (over) smoothing. *Advances in Neural Information Processing Systems* 35 (2022), 2268–2281.

[22] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[23] Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 1748–1757.

[24] Fangyu Li, Shenbao Yu, Feng Zeng, and Fang Yang. 2023. Improving Sequential Recommendation Models with an Enhanced Loss Function. arXiv:2301.00979 [cs.IR]

[25] 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*. 322–330.

[26] Yang Li, Tong Chen, Peng-Fei Zhang, and Hongzhi Yin. 2021. Lightweight self-attentive sequential recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 967–977.

[27] Chang Liu, Xiaoguang Li, Guohao Cai, Zhenhua Dong, Hong Zhu, and Lifeng Shang. 2021. Noninvasive self-attention for side information fusion in sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 4249–4256.

[28] Gang Liu, Michael Sun, Wojciech Matusik, Meng Jiang, and Jie Chen. 2024. Multimodal Large Language Models for Inverse Molecular Design with Retrosynthetic Planning. *arXiv preprint arXiv:2410.04223* (2024).

[29] Gang Liu, Jiaxin Xu, Tengfei Luo, and Meng Jiang. 2024. Graph Diffusion Transformers for Multi-Conditional Molecular Generation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[30] Jiachen Lu, Jinghan Yao, Junge Zhang, Xiatian Zhu, Hang Xu, Weiguo Gao, Chunjing Xu, Tao Xiang, and Li Zhang. 2021. Soft: Softmax-free transformer with linear complexity. *Advances in Neural Information Processing Systems* 34 (2021), 21297–21309.

[31] Andriy Mnih and Russ R Salakhutdinov. 2007. Probabilistic matrix factorization. *Advances in neural information processing systems* 20 (2007).

[32] Shalini Pandey and Jaideep Srivastava. 2020. RKT: relation-aware self-attention for knowledge tracing. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1205–1214.

[33] 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*. 813–823.

[34] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[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*. 811–820.

[36] Jin Shang, Yang Andrew Jiao, Chenghuan Guo, Allen Sun, Yan Gao, Jia Kevin Liu, Michinari Momma, Itetsu Taru, and Yi Sun. 2024. Transitivity-encoded graph attention networks for complementary item recommendations. (2024).

[37] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. 2021. Saint: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342* (2021).

[38] Yading Song, Simon Dixon, and Marcus Pearce. 2012. A survey of music recommendation systems and future perspectives. In *9th international symposium on computer music modeling and retrieval*, Vol. 4. Citeseer, 395–410.

[39] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).

[40] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1441–1450.

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

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

[43] Zhenlei Wang, Xu Chen, Rui Zhou, Quanyu Dai, Zhenhua Dong, and Ji-Rong Wen. 2023. Sequential Recommendation with User Causal Behavior Discovery. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 28–40.

[44] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential recommendation via personalized transformer. In *Proceedings of the 14th ACM Conference on Recommender Systems*. 328–337.

[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)*. IEEE, 1259–1273.

[46] Yueqi Xie, Peilin Zhou, and Sunghun Kim. 2022. Decoupled side information fusion for sequential recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1611–1621.

[47] Yuhao Yang, Chao Huang, Lianghao Xia, Yuxuan Liang, Yanwei Yu, and Chenliang Li. 2022. Multi-behavior hypergraph-enhanced transformer for sequential recommendation. In *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*. 2263–2274.

[48] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining*. 582–590.

[49] Zheng Yuan, Fajie Yuan, Yu Song, Youhua Li, Junchen Fu, Fei Yang, Yunzhu Pan, and Yongxin Ni. 2023. Where to go next for recommender systems? id-vs. modality-based recommender models revisited. *arXiv preprint arXiv:2303.13835* (2023).

[50] Tingting Zhang, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Deqing Wang, Guanfeng Liu, Xiaofang Zhou, et al. 2019. Feature-level Deeper Self-Attention Network for Sequential Recommendation. In *IJCAI*. 4320–4326.

[51] Wei Zhang, Zeyuan Chen, Hongyuan Zha, and Jianyong Wang. 2021. Learning from substitutable and complementary relations for graph-based sequential product recommendation. *ACM Transactions on Information Systems (TOIS)* 40, 2 (2021), 1–28.

[52] Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. 2021. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *proceedings of the 30th acm international conference on information & knowledge management*. 4653–4664.

[53] Rui Zhou, Xian Wu, Zhaopeng Qiu, Yefeng Zheng, and Xu Chen. 2023. Distributionally Robust Sequential Recommnedation. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 279–288.

A METHOD DETAILS

A.1 Derivation of posterior targets

When user's future behaviors $\mathbf{x}_{>t}$ are available, we have the following posterior:

$$\begin{aligned}
\text{Prob}\,(\boldsymbol{\alpha}|\mathbf{x}_{>t}) &\propto \text{Prob}\,(\boldsymbol{\alpha})\,\text{Prob}\,(\mathbf{x}_{>t}|\boldsymbol{\alpha}) && (18)\\
&= \text{Prob}\,(\mathbf{p}|\boldsymbol{\alpha}) \cdot \prod_{x_i \in \mathbf{x}_{>t}} \text{Prob}\,(x_i|\mathbf{p}) && (19)\\
&= \text{Prob}\,(p_1, p_2, \ldots, p_{N_{\text{item}}}|\alpha_1, \alpha_2, \ldots, \alpha_{N_{\text{item}}}) && (20)\\
&\cdot \prod_{x_i \in \mathbf{x}_{>t}} \text{Prob}\,(x_i|p_1, p_2, \ldots, p_{N_{\text{item}}}) && (21)\\
&\propto \prod_{j=1}^{N_{\text{item}}} p_j^{\alpha_j - 1} \prod_{x_i \in \mathbf{x}_{>t}} \prod_{j=1}^{N_{\text{item}}} p_j^{\mathbb{1}\{x_i = j\}} && (22)\\
&= \prod_{j=1}^{N_{\text{item}}} p_j^{\alpha_j - 1 + \sum_{x_i \in \mathbf{x}_{>t}} \mathbb{1}\{x_i = j\}}, && (23)
\end{aligned}$$

where $\mathbb{1}\{x_i = j\} = 1$ if the item x_i from the user sequence has the index j in the distribution parameter vector. The log of the posterior distribution is as follows:

$$\begin{aligned}
\log \text{Prob}\,(\boldsymbol{\alpha}|\mathbf{x}_{>t}) &\propto \log \text{Prob}\,(\boldsymbol{\alpha}) + \log \text{Prob}\,(\mathbf{x}_{>t}|\boldsymbol{\alpha}) && (24)\\
&\propto \sum_{j=1}^{N_{\text{item}}} \log p_j^{\alpha_j - 1 + \sum_{x_i \in \mathbf{x}_{>t}} \mathbb{1}\{x_i = j\}} && (25)\\
&= \sum_{j=1}^{N_{\text{item}}} (\alpha_j - 1) \log p_j + \sum_{x_i \in \mathbf{x}_{>t}} \log p_j^{\sum_{j=1}^{N_{\text{item}}} \mathbb{1}\{x_i = j\}} && (26)\\
&= \underbrace{\sum_{j=1}^{N_{\text{item}}} (\alpha_j - 1) \log p_j}_{\text{prior}} + \underbrace{\sum_{x_i \in \mathbf{x}_{>t}} \log p_{x_i}}_{\text{likelihood}}. && (27)
\end{aligned}$$

A.2 Converting Numerical Attribute Features to Relations

We focus on up to two numerical attributes in this work: price and timestamp. We apply the price relation to items that have the same brand and set $r_{\text{norm}} = 1$. Regarding the time gap, we utilize two scaling factors: $r_{\text{norm}} = 60 \times 60$ for the hour level and $r_{\text{norm}} = 60 \times 60 \times 24$ for the day level. We calculate the hour-level and day-level relations separately and average them to get the final temporal relation. We limit the use of temporal relations to within the sequence, as a user's behavior at the hour or day level may not correlate with other users. For example, consider two users: a student and a company employee. Their behaviors may differ significantly at the same time point.

B MORE EXPERIMENTS

B.1 Setup for Motivation

In Figure 1, we train DIF-SR [46], one of the latest recommendation Transformers, on different subsets of the Amazon Toys & Games dataset [20]. We follow the original setting and use only the item categories attribute. We vary the threshold to filter out users and items with fewer than threshold interactions when sampling the subsets. We employ multi-hot encoding to obtain attribute input features and calculate explicit relations via dot products of multi-hot encoding features. For DIF-SR, we first embed the attribute features into the embedding space and then utilize key and query linear layers to generate key and query embeddings. The learned relation is computed by averaging the multiplication results of the key and query embeddings from different layers, followed by row-wise min-max normalization. We evaluate the KL divergence between the learned relation and explicit (decoupled) relations under varying data sparsity (or average user sequence length). We specifically choose a case from the subset with an average user sequence length of 8.63 and visualize a user example for both the learned relation and the explicit relation.

B.2 More Baseline Details

SASRecF [50, 52] is an extension of SASRec [20] by mixing attribute embedding and item embedding in the input space. MT4SR [10] and DIF-SR [46] leverage learnable parameters to embed attributes and implicitly align them with relations in the attention matrix space. TiSASRec [25] uses only *timestamp*. MT4SR is only applicable on behavior-related attribute like *also_viewed* and *bought_together* on Amazon datasets [10]. DIF-SR uses only the categorical item-related attributes [46]. In practice, we find FEARec [8] takes 460 seconds per epoch on MovieLens-1M, while DuoRec [33] and CL4SRec [45] have similar times, compared to 6 seconds per epoch for AttrFormer. These methods generally require much longer than AttrFormer to achieve comparable performance.

B.3 More Implementation Details

We follow the original implementations of LightSANs [10], MT4SR [10], and DIF-SR [46]. For other baselines and our model, we utilize the RecBole pipeline [52]. We note that our reported results may slightly differ from the original reports on the Amazon datasets for methods like MT4SR [10] and DIF-SR [46] due to multiple reasons. For example, we report performance by averaging multiple runs under different seeds, whereas they perform evaluation with a fixed seed and run the model only once. Particularly, for MT4SR [10], we do not exclude training items during evaluation to ensure a consistent and fair evaluation setup. For DIF-SR [46], the data processing code for attributes under RecBole [52] contains some delimiter bugs, making the differences in reported performance. On the MovieLens-1M dataset containing user attributes, to avoid over-smoothing [21] for AttrFormer, we enable the aggregation module in odd-numbered layers and disable it in even-numbered layers. For our hyperparameters, we set attention head dimension $d = 32$; standard attention heads to $H = 8$; training and evaluation batch size to 128; and attention-to-MLP dimension ratio to 4. We select top-10 relations by default for each interacted item. We also explore within a constrained range: relation heads $M' = \{2, 4\}$; relation dimension $d \in \{16, 32, 64\}$; encoder layers $\in \{3, 4\}$; and attention dropout ratio $\in \{0, 0.3\}$.

B.4 Comparison with Same Attributes

Table 6 compares AttrFormer with attribute-tailored recommendation Transformers using the same attributes as inputs. AttrFormer

Table 6: Comparing attribute-based recommendation Transformers with the same attributes as input. Best result is bolded.

	Amazon Beauty				Amazon Toys & Games			
	Recall@5	Recall@20	NDCG@5	NDCG@20	Recall@5	Recall@20	NDCG@5	NDCG@20
Using categorical item-related: *category* and *brand*								
DIF-SR	0.0578	0.1273	0.0337	0.0535	0.0675	0.1342	0.0380	0.0569
AttrFormer	0.0633	**0.1325**	0.0441	0.0636	0.0714	0.1351	0.0501	0.0682
Using categorical behavior-related: *also_viewed*, *also_bought*, *bought_together*, *buy_after_viewing*								
MT4SR	0.0559	0.1169	0.0360	0.0533	0.0607	0.1148	0.0410	0.0563
AttrFormer	0.0641	0.1324	0.0444	0.0635	0.0718	0.1342	**0.0509**	**0.0686**
Using numerical behavior-related: *timestamp*								
TiSASRec	0.0576	0.1244	0.0344	0.0534	0.0664	0.1322	0.0379	0.0566
AttrFormer	0.0636	0.1309	0.0444	0.0633	0.0709	0.1356	0.0500	0.0683
Using all numerical and categorical attributes								
AttrFormer	**0.0642**	0.1324	**0.0446**	**0.0639**	**0.0720**	**0.1357**	0.0501	0.0681

Table 7: Fine-grained attribute ablation on Amazon datasets. The eight attributes were categorized into item-related and behavior-related parts, encompassing both categorical and numerical attributes.

	Amazon Beauty		Amazon Toys & Games	
	Recall@20	NDCG@20	Recall@20	NDCG@20
SASRec	0.1107	0.0550	0.1073	0.0570
+ Item-related attributes				
w/ categorical	0.1164	0.0576	0.1167	0.0615
w/ numerical	0.1154	0.0571	0.1158	0.0631
w/ all	0.1170	0.0582	0.1180	0.0633
+ Behavior-related attributes				
w/ categorical	0.1152	0.0575	0.1184	0.0635
w/ numerical	0.1152	0.0567	0.1143	0.0626
w/ all	0.1168	0.0576	0.1179	0.0641

consistently outperforms by effectively utilizing attributes as cross-sequence relations.

B.5 More Ablation Studies

We decouple the study of different attributes in Table 7, which illustrates that the model improvements stem from both numerical and categorical attributes. It reveals that AttrFormer can take advantage of explicit relations from diverse attribute types. There is potential for future research to improve AttrFormer by incorporating multi-modal data, such as images and text.

B.6 More Efficiency Analysis

We provide the training times over epochs (average from five epochs) for Transformer-based sequential recommendation methods in Table 4. We observe that AttrFormer is efficient in handling eight types of attributes, while DIF-SR [46] and SASRecF [52] are designed for two categorical and item-related attributes, MT4SR [10] is designed for four categorical and behavior-related attributes, and TiSASRec [25] is designed for numerical and behavior-related temporal relations. All methods use comparable model sizes. SASRec has the lowest training time as it does not consider any attributes. It's worth noting that there are two types of implementations based on method characteristics. One is interaction count-based, while the other is sequence-based, as described in Section 2. The former includes methods like BERT4Rec and lightSANs, where bidirectional attention or low-rank decomposition is needed for a target, limiting the target to the last item in the sequence. The latter, as implemented in other methods, is more efficient, allowing all shifted items to be loss targets in a sequence. While TiSASRec and MT4SR consider fewer attributes, their implementation processes these attributes into relations in each training epoch [10, 25], reducing efficiency, particularly for MT4SR. In contrast, our approach extracts explicit relations from attributes, allowing preprocessing before training, thus improving training model efficiency. Although FEARec only acceptable epochs