Attention Calibration for Transformer-based Sequential Recommendation

Peilin Zhou*
The Hong Kong University of Science and Technology (Guangzhou)
zhoupalin@gmail.com

Qichen Ye*
Peking University
yeeeqichen@pku.edu.cn

Yueqi Xie
The Hong Kong University of Science and Technology
yxieay@connect.ust.hk

Jingqi Gao
Upstage
mrgao.ary@gmail.com

Shoujin Wang
University of Technology Sydney
shoujin.wang@uts.edu.au

Jae Boum Kim
The Hong Kong University of Science and Technology
jbkim@cse.ust.hk

Chenyu You
Yale University
chenyu.you@yale.edu

Sunghun Kim†
The Hong Kong University of Science and Technology (Guangzhou)
hunkim@cse.ust.hk

ABSTRACT

Transformer-based sequential recommendation (SR) has been booming in recent years, with the self-attention mechanism as its key component. Self-attention has been widely believed to be able to effectively select those informative and relevant items from a sequence of interacted items for next-item prediction via learning larger attention weights for these items. However, this may not always be true in reality. Our empirical analysis of some representative Transformer-based SR models reveals that it is not uncommon for large attention weights to be assigned to less relevant items, which can result in inaccurate recommendations. Through further in-depth analysis, we find two factors that may contribute to such inaccurate assignment of attention weights: *sub-optimal position encoding* and *noisy input*. To this end, in this paper, we aim to address this significant yet challenging gap in existing works. To be specific, we propose a simple yet effective framework called **A**ttention **C**alibration for **T**ransformer-based **S**equential **R**ecommendation (**AC-TSR**). In AC-TSR, a novel spatial calibrator and adversarial calibrator are designed respectively to directly calibrates those incorrectly assigned attention weights. The former is devised to explicitly capture the spatial relationships (i.e., order and distance) among items for more precise calculation of attention weights. The latter aims to redistribute the attention weights based on each item's contribution to the next-item prediction. AC-TSR is readily adaptable and can be seamlessly integrated into various existing transformer-based SR models. Extensive experimental results on four benchmark real-world datasets demonstrate the superiority of our proposed AC-TSR via significant recommendation performance enhancements. The source code is available at https://github.com/AIM-SE/AC-TSR.

*Both authors contributed equally.
†Corresponding author.

CIKM '23, October 21–25, 2023, Birmingham, United Kingdom

ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3614785

CCS CONCEPTS

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

KEYWORDS

Sequential Recommendation, Attention Mechanism, Transformer

ACM Reference Format:
Peilin Zhou, Qichen Ye, Yueqi Xie, Jingqi Gao, Shoujin Wang, Jae Boum Kim, Chenyu You, and Sunghun Kim. 2023. Attention Calibration for Transformer-based Sequential Recommendation. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management (CIKM '23), October 21–25, 2023, Birmingham, United Kingdom.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3583780.3614785

1 INTRODUCTION

Sequential Recommender Systems (SRS) have been widely applied to various online services (*e.g.*, e-commerce [14], news websites [5], social media [9]) to make recommendations on the next item which may interest a given user based on her/his historical interactions with items [22, 45]. In recent years, significant advancements have been made in the field of SRS by employing deep learning techniques [43, 46]. Various deep learning models, including recurrent neural networks (RNN) [13, 37], convolutional neural networks (CNN) [40, 58], memory networks [4], graph neural networks (GNN) [1, 50], and attention networks [44] have been introduced to build a variety of SRS and have achieved great success. These SRS methods typically try to predict the next item of interest by well capturing the intricate sequential dependencies among items which have been sequentially interacted by users [42, 47].

In recent years, transformer-based SRS methods have gained significant attention. Benefiting from the particular self-attention mechanism, transformer-based methods are quite effective in capturing both short-term and long-term dependencies among items in a sequence. The pioneering work, SASRec [17], introduced the self-attention mechanism to identify users' dynamic preferences based on their sequential interactions with items. BERT4Rec [38] extended SASRec to a bidirectional self-attentive architecture and incorporated a cloze task to capture contextual information from both the left and right sides of the target item. Subsequent relevant studies have explored additional influential factors in the modeling process, such as time intervals [23], side information [52], and local

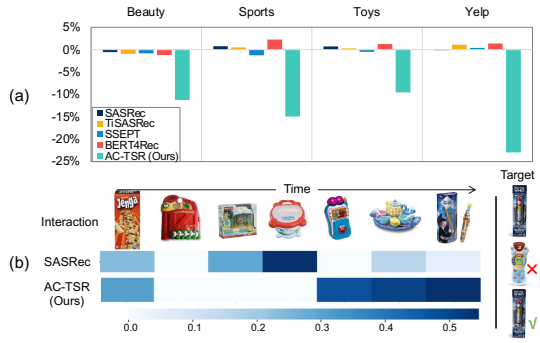

Figure 1: (a) Removing the highest attention weight from transformer-based SRS does not lead to a significant decrease in model performance and even improves performance in some cases; (b) Visualization of the attention weights from SASRec and our proposed AC-TSR.

constraints [12], as well as techniques for reducing the model complexity [7, 24, 60]. Self-attention mechanism, as the key component of transformer-based SRS methods, is thought to have the ability to accurately identify the influence of historical items on the next-item prediction by correctly assigning larger attention weights to those items which are more relevant to the target next item. However, our observations suggest that this may not always hold true in reality. In order to prove this statement, we conduct an empirical study that explicitly demonstrates our findings.

To be specific, we carefully carried out an "*erasing*" experiment to examine the quality of the attention weights learned by the self-attention module in transformer-based SRS methods. In this experiment, we removed the highest attention weight learned by the self-attention module in various transformer-based SRS models and then examined the performance change caused by this removal. As shown in Fig. 1(a), the removal operation only causes a marginal decrease in recommendation performance and even improves performance in some cases, as indicated by the commonly used metric, Recall@20. This proved that the so-called most relevant and decisive item identified by the self-attention module in most transformer-based SRS actually has very little influence on the next-item prediction. This further indicates that self-attention mechanism may be deficient in identifying the decisive items within user behavior sequences. In order to further verify this statement, we visualized the attention weights learned by self-attention module in SASRec in Fig. 1(b). Specifically, when given a sequence sample from the Amazon Toys dataset, which consists of a user's sequence of interacted items "*Jenga, book, aquarium, drum, cell phone, tea set, laser screwdriver*" and the target next item "*sonic screwdriver*," the self-attention mechanism of SASRec inaccurately assigns a larger attention weight to the item "*drum*," which is totally irrelevant to the target next item.

After careful and in-depth analysis, we found the aforementioned unreliable or inaccurate assignment of attention weights could be mainly attributed to the following two factors: (1) *Sub-optimal position encoding*. To capture the sequential dependencies over items, conventional transformer-based SRS methods directly incorporate item position embeddings into item embeddings and then use the integrated ones to compute the attention weights, suffering from the rank bottleneck [52] and noisy correlations [7]. Recent studies [7, 52] suggest decoupling the position encoding to calculate position correlations independently. However, such treatment is still sub-optimal since it fails to explicitly leverage low-level spatial information, including order and distance information, which have been found useful in enhancing the representation power of positional encoding in error-prone data [21]. (2) *Noisy input*. Existing SRS works often assume a correlation between the target item and all historically interacted items. However, in real-world scenarios, this assumption might not always hold since users may unconsciously interact with some items that deviate from their interests or preferences, resulting in noisy interaction data [39, 48]. Multiple factors, including users' moods, social needs, personal conditions, etc., could lead to this noisy input phenomenon. For instance, users may randomly play popular videos or songs on a website that do not necessarily align with their preferences, or they might make purchases based on their mood or for their friends. These sources of noise are challenging to identify as they often exhibit vague patterns, posing difficulties in accurately learning users' true preferences through the self-attention mechanism. Additionally, the self-attention mechanism is prone to overfitting on noisy input, further complicating the problem [65].

To this end, in this paper, to address the aforementioned significant gaps in existing SRS studies, we propose a framework called **A**ttention **C**alibration for **T**ransformer-based **S**equential **R**ecommendation (**AC-TSR**). AC-TSR utilizes two well-designed calibrators, i.e., **Spatial Calibrator (SPC)** and **Adversarial Calibrator (ADC)** , to calibrate the unreliable attention weights illustrated above. The SPC is designed to address the problem of sub-optimal position encoding by explicitly leveraging spatial relationships, such as the order and distance between items in a user sequence, to calculate attention weights that possess greater structural significance. Specifically, SPC incorporates sequential relations directly into the attention matrix, eliminating the need for additional position embeddings such as absolute, relative, or decoupled ones. The ADC tackles the issue of noisy input by redistributing attention weights based on the contribution of each historical item to the model's prediction. The term "adversarial" is employed because the reallocation process is performed in an adversarial manner. The joint use of both calibrators can enhance the robustness of the original attention map against noisy input and enables more precise capture of user preferences.

Experiments on four benchmark datasets show that AC-TSR outperforms both non-transformer-based SRS methods and representative transformer-based SRS methods by calibrating the attention weights learned by the self-attention mechanism. Moreover, we demonstrate that both calibrators are plug-and-play and thus can be seamlessly incorporated into existing transformer-based SRS models to enhance their performance. Furthermore, we delve into each calibrator's influence and hyperparameters' effect in AC-TSR. A comprehensive analysis is carried out to shed light on why AC-TSR can achieve such performance gains.

In summary, we make the following contributions:

- We propose the AC-TSR framework, which can effectively reduce the impact of sub-optimal position encoding and

noisy input on the existing transformer-based SRS models with limited overhead.
- We develop two plug-and-play calibrators, namely spatial calibrator and adversarial calibrator, to rectify the unreliable attention. This ensures that the model focuses more on informative items when predicting the next item.
- We conduct comprehensive experiments on four benchmark datasets, demonstrating the superiority of AC-TSR over state-of-the-art SRS methods.

2 RELATED WORK

2.1 Sequential Recommendation

Sequential recommender systems (SRS) aim to predict the next item a user will likely interact with based on their past interactions. Early works, including pattern mining based [54] and Markov chains [10, 34] based approaches focused on mining simple and low-order sequential dependencies, constrained by rigid assumptions and thus cannot deal with complex data with high order dependencies. Later, with the advancements in neural networks, there has been a shift towards utilizing complex models such as Convolutional Neural Networks (CNNs) [40, 53, 58], Recurrent Neural Networks (RNNs) [13, 28, 31, 32], and Graph Neural Networks (GNNs) [1, 8, 50] to deal with the complex sequence patterns in the sequential recommendation task. For example, Caser [40] is a convolutional sequence model to learn sequential patterns using both horizontal and vertical convolutional filters. GRU4Rec [13] employs a gated recurrent unit to study the temporal behaviors of users. And SR-GNN [50] converts session sequences into graphs and uses graph neural networks to capture complex item-item transitions.

More recently, transformer-based methods [12, 16, 17, 24, 26, 38, 66] have become the mainstream solutions due to their great potential in capturing user's sequential behavior through self-attention mechanism. SASRec [17] first adopts self-attention mechanism to capture users' sequential behaviors. And BERT4Rec [38] extends it to a bidirectional model with the help of Cloze task. In the follow-up studies, enhancements were made by incorporating time intervals [23], personalization [49], importance sampling [25], consistency [11], multiple interests [51], evolutionary preference [8], continue learning [61, 62] and decoupled positional encoding [7]. However, very few studies pay attention to the quality of the learned attention weights in transformer-based SRS. In this study, we discover that in current transformer-based SRS models, the historical items with high attention weights do not consistently contribute to accurately predicting the target item. Based on this observation, we propose to improve the quality of attention weights by injecting calibrators into transformer-based SRS models.

2.2 Debates on Attention Mechanism

The debate surrounding the attention mechanism originates in the field of deep learning: for one thing, some researchers find that replacing high attention weights with lower ones does not affect the model's prediction performance [30, 36, 55–57], possibly because the attention mechanism tends to assign higher weights to tokens that are not important, such as punctuation and stop words; for another, some studies in text classification [3, 15] have revealed weak correlation between attention weights and gradient-based feature importance metrics. Furthermore, a recent machine translation work [27] also observe that the attention mechanism fails to accurately identify the decisive inputs for each prediction, leading to incorrect or excessive translation in NMT. These debates on attention mechanisms inspire us to re-examine the learned attention weights in transformer-based SRS models.

In fact, there have been some SRS works that discuss the limitations of the attention mechanism: Locker [12] contends that the self-attention modules struggle to capture the short-term user dynamics accurately. As a result the authors introduce an inductive local bias into the self-attention mechanism, improving the modeling of short-term user dynamics while maintaining long-term semantic information. Rec-Denoiser [2] alleviates the influence of noise in the data by sparsifying the attention map, with the implicit assumption that not all attention weights carry important information in the self-attention layer. Different from these methods, our proposed AC-TSR effectively utilizes low-level spatial information through a spatial calibrator, and adopts an adversarial calibrator to adaptively identify the decisive items within the interaction sequence and automatically adjust the distribution of attention weights without relying on prior knowledge.

3 PRELIMINARY

3.1 Problem Setup

Let $\mathcal{U}, \mathcal{I}$ denote the sets of users and items, respectively. For a user $u \in \mathcal{U}$, the historical interactions of this user can be represented as $\mathcal{S}^u = [v_1^u, v_2^u, \ldots, v_{|\mathcal{S}^u|}^u]$, where $v_i^u \in \mathcal{I}$ is the *i-th* interaction in the chronologically ordered sequence $\mathcal{S}^u$ and $|\mathcal{S}^u|$ denotes the sequence length. The set of users' actions can be represented as $\mathcal{S} = \{\mathcal{S}^1, \mathcal{S}^2, \ldots, \mathcal{S}^{|\mathcal{U}|}\}$, where $|\mathcal{U}|$ is the number of users.

Given the historical interaction sequence $\mathcal{S}^u$ of user u, the goal of sequential recommendation is to predict the next item $v_{next} \in \mathcal{I}$ that user u will interact with at the $(|\mathcal{S}^u|+1)$-*th* time step, denoted as $p\left(v_{next} \mid \mathcal{S}^u\right)$.

3.2 Transformer-based Recommenders

Due to the remarkable capability in modeling sequential data, the transformer architecture has attracted a lot of attention and has been widely explored in the sequential recommendation [17, 23, 38]. Among these efforts, SASRec [17] is a particularly noteworthy piece of work, as it was the first transformer-based SR model and achieved competitive performance on many public datasets. Therefore, in this session, we take SASRec as an example to offer a succinct overview of the transformer architecture.

3.2.1 Embedding Layer. The transformer-based recommenders maintain an item embedding table $\mathbf{T} \in \mathbb{R}^{|\mathcal{I}| \times d}$ to convert items from discrete ids to dense vectors, where d represents the embedding size. First, a user interaction sequence $\mathcal{S}^u = [v_1^u, v_2^u, \ldots, v_{|\mathcal{S}^u|}^u]$ is transformed into a fixed-length sequence $\hat{\mathcal{S}}^u = [v_1^u, v_2^u, \ldots, v_n^u]$ by keeping most recent n items or padding items, where n is the maximum sequence length. Then, the sequence embedding $\mathbf{E} \in \mathbb{R}^{n \times d}$ of $\hat{\mathcal{S}}^u$ is generated through the item embedding table $\mathbf{T}$. Finally, to consider the impact of different positions within the sequence, a learnable position embedding $\mathbf{P} \in \mathbb{R}^{n \times d}$ is added to $\mathbf{E}$ to obtain the

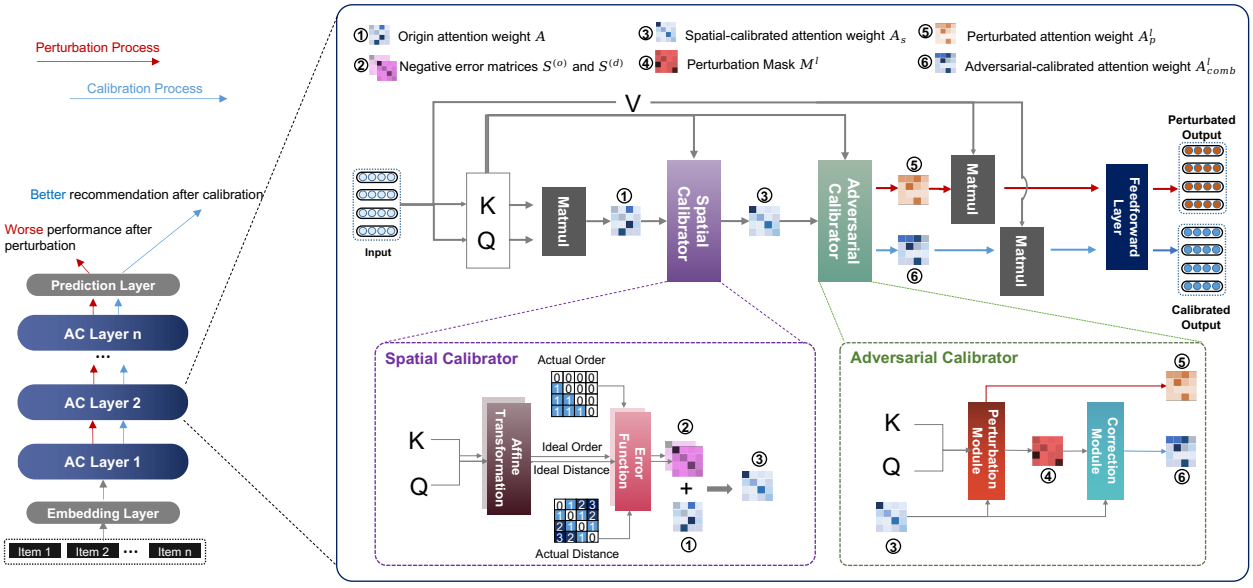

Figure 2: Overview of the proposed AC-TSR framework. The SASRec model functions as the backbone, where its self-attention layer is converted into an Attention Calibration (AC) layer for improved performance. Each AC layer contains a spatial calibrator (purple dotted box) and an adversarial calibrator (green dotted box). The spatial calibrator is responsible for incorporating spatial information such as order and distance into the attention weights. The adversarial calibrator aims to identify decisive items and adjust the distribution of attention weights.

final sequence embedding $\hat{\mathbf{E}}$, which serves as the input for the first transformer block.

3.2.2 Transformer Block. Transformer-based SR models usually use several stacked transformer blocks to capture the hierarchical dependencies between items within the input sequence. Each transformer block consists of two components: a self-attention layer and a point-wise feed-forward layer.

Self-attention Layer: The core of this layer is self-attention mechanism, which is designed to uncover the dependencies among items within a sequence [41]. Specifically, the output item representation $\mathbf{H} \in \mathbb{R}^{n \times d}$ is calculated as follows:

$$\mathbf{H} = \text{Self-Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}, \tag{1}$$

where $\mathbf{Q} = \hat{\mathbf{E}}\mathbf{W}_Q$, $\mathbf{K} = \hat{\mathbf{E}}\mathbf{W}_K$ and $\mathbf{V} = \hat{\mathbf{E}}\mathbf{W}_V$, $\{\mathbf{Q}, \mathbf{K}, \mathbf{V}\} \in \mathbb{R}^{n \times d}$ are the queries, keys and values respectively, and $\{\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V\} \in \mathbb{R}^{d \times d}$ are three learnable projection matrices, $\sqrt{d}$ is the scale factor. Thus, the attention weight $\mathbf{A} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)$ establishes a whole-range connection between all items within sequence $\hat{\mathcal{S}}^u$. Note that the Eq.1 can be further extended to the multi-head self-attention to obtain better expressiveness (*i.e.* each attention head focuses on a different type of attention pattern [19]). Here we only show the single-head version for simplicity, and readers can check more details in the original paper [41].

Point-wise Feed-forward Layer: This layer has the capability to learn complex feature representations, thereby enhancing the expressive power of the transformer architecture. After the calculation of self-attention, the output item representation $\mathbf{H}$ is fed to the point-wise feed-forward layer:

$$\mathbf{F}_i = \text{FFN}(\mathbf{H}_i) = \text{ReLU}(\mathbf{H}_i\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \tag{2}$$

where $\mathbf{F}_i$ denotes the *i-th* output embedding, $i \in [1, n]$; $\mathbf{W}_*$ and $\mathbf{b}_*$ are learnable weights and bias, respectively. Note that in Eq.2, we leave out the residual connection, dropout, and layer normalization for simplicity. In practice, these techniques can be adopted to enhance stability and speed up the training process.

Transformer blocks are usually stacked to learn hierarchical item dependencies, and the output of *L-th* block can be represented as:

$$\mathbf{F}_i^L = \text{FFN}\left(\mathbf{H}_i^L\right). \tag{3}$$

3.2.3 Learning Objective. In SASRec, the prediction of the next item $v_{next} \in \mathcal{I}$ is based on the last element of $\mathbf{F}^L$, denoted as $\mathbf{F}_n^L$. Specifically, the probability of interaction between user u and each item is calculated through the inner product of $\mathbf{F}_n^L$ and the item embedding obtained from the item embedding table $\mathbf{T} \in \mathbb{R}^{|\mathcal{I}| \times d}$. This process can be formulated as follows:

$$\hat{\mathbf{y}} = \text{softmax}\left(\mathbf{T}\mathbf{F}_n^{L^T}\right), \tag{4}$$

where $\hat{\mathbf{y}} \in \mathbb{R}^{|\mathcal{I}|}$ denotes the predicted probability. Afterwards, cross-entropy is chosen as the loss function to measure the discrepancy between the prediction $\hat{\mathbf{y}}$ and the ground truth $\mathbf{y}$:

$$\mathcal{L} = -\sum_{i=1}^{|\mathcal{I}|} y_i \log(\hat{y}_i) \tag{5}$$

4 AC-TSR FRAMEWORK

4.1 Overall Architecture

The overall architecture of AC-TSR is depicted in Fig. 2, which introduces two new components, namely **Spatial Calibrator** (Sec. 4.2) and **Adversarial Calibrator** (Sec. 4.3), into the self-attention layer of TSR. This modification transforms the original self-attention layer into an Attention Calibration (AC) layer. Inside each AC layer, the attention weights computed by the original attention mechanism are initially calibrated in sequence, starting with the Spatial Calibrator and followed by the Adversarial Calibrator. Specifically, the Spatial Calibrator integrates spatial information, including order and distance, into the attention weights, while the Adversarial Calibrator aims to identify critical items and adjust the distribution of attention weights. Notably, the Adversarial Calibrator's output comprises both the perturbed attention weights and the calibrated attention weights for enhanced performance. These two weights are utilized to compute perturbed and calibrated representations, respectively. The perturbed representations are employed to disrupt the recommendation performance, whereas the calibrated representations are used to improve the recommendation performance. A more accurate attention distribution is ultimately obtained through the adversarial process of perturbation and calibration.

4.2 Spatial Calibrator

AC-TSR abandons traditional position encoding techniques, including absolute and relative embeddings, as they are hindered by the rank bottleneck [52] and the noisy correlations [7]. Instead, it adopts a spatial calibrator (SPC) to empower the self-attention layer with the ability to recognize spatial relationships within the input sequence without the need for positional embeddings. To this end, we first compute the *order* and *log-distance* between pairs of items with respect to the position in the input sequence and then directly use these low-level features to adjust the pre-softmax attention weights (*i.e.,* the $\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}$ in Eq. 1). Specifically, the low-level features of actual *order* o_{ij} and actual *log-distance* d_{ij} between position i and j in the input sequence are defined as follows:

$$o_{ij} = \mathbb{I}(i < j) = \begin{cases} 1, & i < j \\ 0, & otherwise \end{cases} \tag{6}$$

$$d_{ij} = \ln\left(1 + |i - j|\right). \tag{7}$$

Then we use the query $\mathbf{q}_i^l$ and key $\mathbf{k}_j^l$ in each self-attention layer to predict the orders and distances the items should have if there exists a meaningful dependency between them:

$$\hat{o}_{ij} = \text{sigmoid}\left(\text{affine}^{(o)}\left(\left[\mathbf{q}_i^l; \mathbf{k}_j^l\right]\right)\right), \tag{8}$$

$$\hat{d}_{ij} = \text{affine}^{(d)}\left(\left[\mathbf{q}_i^l; \mathbf{k}_j^l\right]\right), \tag{9}$$

where $\hat{o}_{ij}$ and $\hat{d}_{ij}$ denote predicted order and distance, respectively. Finally, sigmoid cross-entropy and L_2 loss are adopted to calculate the discrepancies between the predictions and the ground truths, which are added into the origin attention weights:

$$s_{ij}^{(o)} = o_{ij} \ln\left(\hat{o}_{ij}\right) + (1 - o_{ij})(1 - \ln\left(\hat{o}_{ij}\right)), \tag{10}$$

$$s_{ij}^{(d)} = -\frac{\theta^2\left(d_{ij} - \hat{d}_{ij}\right)^2}{2}, \tag{11}$$

$$\mathbf{A}_s = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}} + \mathbf{s}^{(o)} + \mathbf{s}^{(d)}\right), \tag{12}$$

where θ is a learnable scalar and $\mathbf{A}_s$ is the calibrated attention weights after spatial calibrator.

The core idea of the spatial calibrator is to correct the attention weights by penalizing the attention edges that violate the order or distance constraints (*i.e.* o_{ij} or d_{ij}). Intuitively, if the self-attention mechanism can capture these low-level features, then we can assume that the spatial relationships are encoded in the query $\mathbf{Q}$ and key $\mathbf{K}$ since the attention weights are calculated based on them. In other words, the prediction errors calculated by Eq. 10 and Eq. 11 can reflect the potential weakness in the corresponding attention weights, so we calibrate it by adding a penalty to these positions.

4.3 Adversarial Calibrator

The adversarial calibrator (ADC) aims to mitigate the noisy input issue mentioned in Sec. 1 by making the self-attention mechanism more focused on the informative and decisive historical items. To achieve this goal, we design a **Perturbation Module** and a **Correction Module** (*cf.* Fig. 2). Specifically, the perturbation module first automatically identifies the decisive historical items by adding limited perturbations to the original attention weights. The correction module then calibrates the attention weights through highlighting the critical inputs (*i.e.*, the perturbed positions) detected by the perturbation module.

4.3.1 Perturbation Module. The core idea of the perturbation module is to detect the decisive part in user sequence by perturbating original attention weights. For a transformer-based SR model, its performance is expected to be poor if the attention weights corresponding to the decisive parts are perturbed. Based on this, for the *l-th* layer, a perturbation mask $\mathbf{M}^l$ is utilized to introduce uniform distribution μ to the spatial carlibrated attention weight $\mathbf{A}_s^l$, which simulates the process of perturbation:

$$\mathbf{A}_p^l = \mathbf{M}^l \odot \mathbf{A}_s^l + (1 - \mathbf{M}^l) \odot \mu, \tag{13}$$

$$\mathbf{M}^l = \text{sigmoid}\left(\frac{\mathbf{Q}^l\mathbf{W}_{Q_p}^l\left(\mathbf{K}^l\mathbf{W}_{K_p}^l\right)^T}{\sqrt{d}}\right), \tag{14}$$

where $\mathbf{A}_p^l$, $\mathbf{Q}^l$ and $\mathbf{K}^l$ are the perturbed attention weight, the query and the key in the l-th self-attention layer; $\odot$ denotes the element-wise multiplication; the $\mathbf{W}_{Q_p}^l \in \mathbb{R}^{d\times d}$ and $\mathbf{W}_{K_p}^l \in \mathbb{R}^{d\times d}$ are two learnable matrices.

4.3.2 Correction Module. As aforementioned, the perturbation module aims to deteriorate the model's performance by removing the decisive information through perturbation mask $\mathbf{M}^l$. In other words, $\mathbf{M}^l$ reveals the critical part in $\mathbf{A}_s^l$. To this end, we propose to adjust the spatial carlibrated attention weights $\mathbf{A}_s^l$ by highlighting the essential part:

$$\mathbf{A}_c^l = \mathbf{A}_s^l \odot e^{1-\mathbf{M}^l}. \tag{15}$$

In the above equation, we increase the attention weight in the spatial carlibrated attention $\mathbf{A}_s^l$ where the perturbation module module assigns large perturbation to (*i.e.,* informative items that are critical for the model output). Afterwards, inspired by [52], we adopts a gating function to dynamically determine the amount of adversarial calibrated attention weight $\mathbf{A}_c^l$ to be fused into the spatial carlibrated attention weight $\mathbf{A}_s^l$, and the **combined** attention weight $\mathbf{A}_{comb}$ is calculated as follows:

$$\mathbf{A}_{comb}^l = \mathbf{g} * \mathbf{A}_s^l + (1 - \mathbf{g}) * \mathbf{A}_c^l, \tag{16}$$

$$\mathbf{g} = \sigma\left(\mathbf{Q}^l \mathbf{W}_g^l + \mathbf{b}_g^l\right), \tag{17}$$

where $\mathbf{W}_g^l \in \mathbb{R}^{d \times d}$, $\mathbf{b}_g^l \in \mathbb{R}^d$ are trainable parameters.

4.4 Training Objective

After obtaining the perturbed attention weight $\mathbf{A}_p$ and the combined attention weight $\mathbf{A}_{comb}$, we can further calculate the perturbed output embedding $\tilde{\mathbf{F}}_p$ and calibrated output embedding $\tilde{\mathbf{F}}_c$ by replacing the origin attention weight $\mathbf{A}$ in Eq. 1 with $\mathbf{A}_p$ and $\mathbf{A}_{comb}$, respectively. Then we can calculate the perturbed loss $\mathcal{L}_P$ and calibrated loss $\mathcal{L}_C$ as follows:

$$\mathcal{L}_P = -\sum_{i=1}^{|\mathcal{I}|} y_i \log\left(\hat{y}_i^P\right), \tag{18}$$

$$\mathcal{L}_C = -\sum_{i=1}^{|\mathcal{I}|} y_i \log\left(\hat{y}_i^C\right), \tag{19}$$

where $\hat{y}_i^P$ and $\hat{y}_i^C$ are calculated by replacing $\mathbf{F}_n^L$ in Eq. 4 with $\tilde{\mathbf{F}}_{p,n}^L$ and $\tilde{\mathbf{F}}_{c,n}^L$, respectively.

On the one hand, we want the model's performance based on the perturbed attention to be worse. On the other hand, we want to deteriorate the model's performance with as small perturbation as possible. Based on these two concerns, we define the final learning objective for the perturbation module as:

$$\mathcal{L}_{P_{final}}\left(\theta^P\right) = -\mathcal{L}_P\left(\theta\right) + \alpha \mathcal{L}_{norm}\left(\theta^P\right), \tag{20}$$

$$\mathcal{L}_{norm}\left(\theta^P\right) = \sum_{l=0}^{L} ||1 - \mathbf{m}^l||_2, \tag{21}$$

where θ^P denotes the parameters for perturbation module (*i.e.,* $\left\{\mathbf{W}_{Q_p}, \mathbf{W}_{K_p}\right\}$ in all layers), and θ represents all other model parameters. L is the number of transformer blocks and α is a hyper-parameter that balances $\mathcal{L}_P$ and $\mathcal{L}_{norm}$.

Finally, we use the following loss to supervise our AC-TSR:

$$\mathcal{L}_{final} = \mathcal{L}_{P_{final}} + \mathcal{L}_C. \tag{22}$$

4.5 Model Complexity

The complexity of the AC-TSR model is derived from three components: the original transformer, the spatial calibrator, and the adversarial calibrator. The complexity of the original transformer remains consistent with that of backbone models such as SASRec and BERT4Rec. Inevitably, employing both calibrators simultaneously in the original transformer layer would increase the number of parameters and computational costs. To mitigate this issue, we propose several strategies for the training and inference stages to enhance computational efficiency:

Training. The two calibrators can be selectively applied to each transformer layer based on available computational resources, due to the effectiveness of using a single type of calibrator demonstrated by ablation study (Tab. 3). For example, if temporal relationships are more significant in a dataset, one might prefer to only use the spatial calibrator. In extremely resource-constrained scenarios, one can opt to integrate a calibrator at just one layer rather than all layers. These strategies can effectively alleviate the computational burden introduced by the calibrators during the training stage, while keeping the time complexity in the same order as the original Transformer-based SR models.

Inference. We propose a lightweight version of AC-TSR, namely AC-TSR-lite, which excludes the two types of calibrators during the inference stage. As a result, AC-TSR-lite maintains the same number of parameters and inference speed as the backbone TSR model. The performance of AC-TSR-lite will be elaborated in Sec 5.2.

5 EXPERIMENTS

We conduct extensive experiments on four real-world and widely-used datasets to answer the following research questions:

- **RQ1:** Does the proposed AC-TSR exhibit competitive performance compared to current state-of-the-art transformer-based SR methods?
- **RQ2:** Can the Spatial Calibrator and Adversarial Calibrator be effectively integrated into representative transformer-based SR models and improve their performance?
- **RQ3:** What are the impacts of different components and hyper-parameters on AC-TSR's performance?
- **RQ4:** Why can AC-TSR achieve superior performance compared to other methods?

5.1 Experimental Setup

5.1.1 Datasets. The experiments are conducted on a well-known business recommendation dataset called **Yelp** [1], and three categories from the Amazon Review dataset [29]: **Beauty**, **Sports** and **Toys**. [2] All the interactions are regarded as implicit feedback. For Yelp dataset, we follow [64] and only retain the transaction records after Jan. 1st, 2019 for our experiment. To align with previous studies [17, 59, 64], we remove all items and users that occur less than 5 times in these datasets. Readers can check the statistics of all these four datasets after preprocessing from [52].

5.1.2 Evaluation Metrics. In our experiments, following the prior works [17, 38], we use *leave-one-out* strategy for evaluation. Specifically, for each user-item interaction sequence, the last two items are reserved as validation and testing data, respectively, and the rest are utilized for training SR models. The performances of SR models are evaluated by top-K Recall (Recall@K) and top-K Normalized Discounted Cumulative Gain (NDCG@K) with K chosen from $\{10, 20\}$, which are two commonly used metrics. To ensure a fair comparison, we adhere to the suggestion from [6, 20] by evaluating the model performance in a full ranking manner, where the

[1] https://www.yelp.com/dataset
[2] http://jmcauley.ucsd.edu/data/amazon/

Table 1: Overall performance. The highest results are denoted in bold, while the runner-up results are underscored. "*" denotes the statistical siginificance for $p < 0.01$ compared to the best baseline methods with paired t-test.

SR Model	Beauty				Sports				Toys				Yelp			
	Recall		NDCG		Recall		NDCG		Recall		NDCG		Recall		NDCG	
	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20	@10	@20
PopRec	0.0157	0.0242	0.0076	0.0097	0.0146	0.0244	0.0078	0.0103	0.0105	0.0172	0.0060	0.0077	0.0099	0.0161	0.0051	0.0067
BPR	0.0375	0.0590	0.0168	0.0222	0.0302	0.0480	0.0144	0.0188	0.0344	0.0560	0.0151	0.0205	0.0589	0.0830	0.0324	0.0384
GRU4Rec	0.0654	0.1002	0.0322	0.0410	0.0386	0.0609	0.0195	0.0251	0.0449	0.0708	0.0221	0.0287	0.0418	0.0679	0.0206	0.0271
Caser	0.0474	0.0731	0.0239	0.0304	0.0227	0.0364	0.0118	0.0153	0.0361	0.0566	0.0186	0.0238	0.0380	0.0608	0.0197	0.0255
LightSANS	0.0770	0.1177	0.0358	0.0461	0.0509	0.0781	0.0226	0.0294	0.0768	0.1116	0.0354	0.0442	0.0630	0.0904	0.0385	0.0453
Locker	0.0802	0.1197	0.0365	0.0464	0.0508	0.0753	0.0225	0.0286	0.0755	0.1094	0.0345	0.0430	0.0603	0.0869	0.0380	0.0446
SASRec	0.0779	0.1152	0.0353	0.0447	0.0504	0.0760	0.0224	0.0289	0.0776	0.1100	0.0352	0.0434	0.0618	0.0879	0.0387	0.0453
w/ AC	0.0817*	0.1218*	**0.0375***	0.0454*	0.0532*	0.0817*	0.0235*	0.0307*	0.0825*	0.1166*	0.0371*	0.0456*	**0.0664***	**0.0955***	**0.0407***	**0.0480***
Improve.	4.88%	5.73%	6.23%	1.57%	5.56%	7.50%	4.91%	6.23%	6.31%	6.00%	5.40%	5.07%	7.44%	8.65%	5.17%	5.96%
BERT4Rec	0.0557	0.0868	0.0279	0.0358	0.0313	0.0502	0.0155	0.0202	0.0489	0.0769	0.0253	0.0324	0.0467	0.0710	0.0264	0.0325
w/ AC	0.0628*	0.0929*	0.0318*	0.0394*	0.0381*	0.0607*	0.0196*	0.0253*	0.0643*	0.0924*	0.0339*	0.0410*	0.0481*	0.0769*	0.0265*	0.0337*
Improve.	12.73%	7.03%	13.98%	10.06%	21.73%	20.92%	26.45%	25.25%	31.49%	20.16%	33.99%	26.54%	3.00%	8.31%	0.38%	3.69%
SSE-PT	0.0587	0.0936	0.0278	0.0366	0.0363	0.0580	0.0184	0.0239	0.0560	0.0837	0.0255	0.0325	0.0556	0.0779	0.0323	0.0379
w/ AC	0.0629*	0.1001*	0.0293*	0.0387*	0.0379*	0.0589*	0.0191*	0.0244*	0.0614*	0.0896*	0.0282*	0.0353*	0.0565*	0.0821*	0.0330*	0.0394*
Improve.	7.16%	6.94%	5.40%	5.74%	4.41%	1.55%	3.80%	2.09%	9.64%	7.05%	10.59%	8.62%	1.62%	5.39%	2.17%	3.96%
TiSASRec	0.0794	0.1208	0.0356	0.0461	0.0523	0.0799	0.0230	0.0300	0.0819	0.1171	0.0367	0.0456	0.0618	0.0909	0.0387	0.0460
w/ AC	**0.0823***	**0.1227***	0.0373*	**0.0474***	**0.0548***	**0.0837***	**0.0241***	**0.0313***	**0.0831***	**0.1208***	**0.0375***	**0.0470***	0.0654*	0.0939*	0.0401*	0.0473*
Improve.	3.65%	1.57%	4.78%	2.82%	4.78%	4.76%	4.78%	4.33%	1.47%	3.16%	2.18%	3.07%	5.83%	3.30%	3.62%	2.83%

Table 2: Model Complexity.

Model	# Parameters	Inference speed	Recall@20			
			Beauty	Sports	Toys	Yelp
SASRec	0.87M	2482.33/s	0.1152	0.0760	0.1100	0.0879
AC-SASRec	0.90M	917.54/s	0.1218	0.0817	0.1166	0.0955
AC-SASRec-lite	0.87M	2482.33/s	0.1164	0.0768	0.1150	0.0913

ranking results are obtained over the entire item set rather than a sampled subset.

5.1.3 Baseline Methods. We compare our proposed AC-TSR with 10 different baselines, including 2 general methods: **PopRec** and **BPR** [33], 2 basic SR methods: **GRU4Rec** [13] and **Caser** [40], and 6 representative transformer-based approaches: **LightSANS** [7], **Locker** [12], **BERT4Rec** [38], **SASRec** [17], **SSE-PT** [49] and **TiSASRec** [23]. We do not select Rec-Denoiser [2] as a baseline because the authors have not released their codes and our own implementation is unsuccessful due to an out-of-memory error when computing the Jacobian matrix. In addition, it is worth mentioning that there have been numerous transformer-based SR models introduced in recent years. However, we have not considered all of them as baselines due to the fact that some of these methods are not suitable for direct comparison due to their reliance on different training paradigms.

5.1.4 Implementation Details. For fair comparisions, both baselines and our proposed AC-TSR are implemented using the popular recommendation framework RecBole [63] and evaluated with the same setting. For all baselines and our proposed method, we train them with Adam optimizer for 200 epochs, with a batch size of 256 and a learning rate of 1e-4. The max sequence length of Sports, Toys, Beauty, and Yelp is set to 50. For LOCKER, we use a CNN as the local encoder (same as *LOCKER+Conv* in their official implementation[3]). For our proposed AC-TSR and other transformer-based baselines (*e.g.*, SASRec), we perform grid search on other hyper-parameters to find the best combination. The searching space is: number of self-attention layers $\in \{2, 3\}$, number of self-attention heads $\in \{2, 4\}$, hidden size $\in \{64, 128\}$ and inner size $\in \{64, 128\}$. Both baselines and our method are carefully tuned on the used datasets for best performance.

[3]https://github.com/AaronHeee/LOCKER

5.2 Overall Performance (RQ1&2)

The results of various methods across all datasets are summarized in Tab. 1. Our proposed AC-TSR method exhibits superior performance on all datasets. Furthermore, we have the following observations:

Most transformer-based methods, such as SASRec and TiSASRec, consistently outperform non-transformer-based methods like Caser and GRU4Rec by a large margin. This observation highlights the superiority of transformer-based methods in capturing long-range item dependencies in sequential data. Additionally, despite BERT4Rec being a promoted version of SASRec, its performance is not as impressive as SASRec under the full-ranking evaluation setting, which has also been noted in previous studie [24, 52]. One potential reason is that the original BERT4Rec paper employs a popularity-based sampling strategy for model evaluation. This strategy benefits BERT4Rec since its bi-directional encoder, combined with the Cloze task, allows for improved learning of representations for popular items.

Auxiliary information can enhance the performance of transformer-based models. By comparing the performance of SASRec, TiSASRec, and SSE-PT, we observe that transformer-based models can benefit from the integration of auxiliary information such as time intervals and user personality. This is due to auxiliary information providing essential cues for a more profound comprehension of users' dynamic behaviors. Capturing these behaviors accurately would be otherwise difficult, relying solely on the standard self-attention mechanism.

Spatial Calibrator and Adversarial Calibrator can be seamlessly incorporated into existing transformer-based SR models (TSR) and boost their performance. In our experiments, we choose SASRec, BERT4Rec, TiSASRec, and SSE-PT as backbones, which represent the unidirectional model, bidirectional model, and auxiliary information enhanced model (*i.e.,* time interval and user

Table 3: Ablation study of AC-TSR on Beauty dataset.

Settings	Spatial		Adv.	Recall		NDCG	
	order	distance		@10	@20	@10	@20
(A)	✔	✔	✔	0.0817	0.1218	0.0375	0.0476
(B)	✔	✗	✔	0.0791	0.1210	0.0364	0.0470
(C)	✗	✔	✔	0.0792	0.1201	0.0372	0.0475
(D)	✗	✗	✔	0.0800	0.1202	0.0367	0.0469
(E)	✔	✔	✗	0.0802	0.1197	0.0365	0.0464
(F)	✔	✗	✗	0.0776	0.1168	0.0353	0.0452
(G)	✗	✔	✗	0.0806	0.1188	0.0367	0.0463
(H)	✗	✗	✗	0.0779	0.1152	0.0353	0.0447

Table 4: Impact of different positional encoding strategies. The SASRec is chosen as the backbone.

Position Encoding Strategy	Sports		Toys	
	Recall@20	NDCG@20	Recall@20	NDCG@20
Remove Position	0.0775	0.0294	0.1170	0.0456
Absolute Position	0.0760	0.0289	0.1100	0.0434
Relative Position	0.0753	0.0285	0.1172	0.0461
Decoupled Position	0.0769	0.0295	0.1153	0.0449
Spatial Calibrator (Ours)	**0.0785**	**0.0298**	**0.1193**	**0.0462**

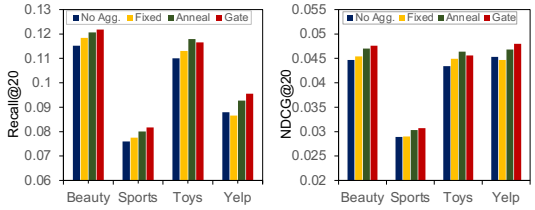

Figure 3: Impact of different aggregation strategies in Correction Module.

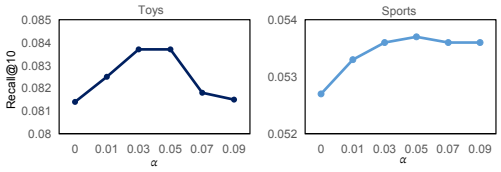

Figure 4: Effect of balance parameter α.

embedding), respectively. And we adapt them into our proposed AC-TSR framework by dropping their position encoding module and replacing their transformer layer with our proposed Attention Calibration layer. As shown in Tab. 1, the TSR models integrated into the AC-TSR framework (highlighted in gray) exhibit significant performance improvements compared to the original TSR model. Specifically, AC-BERT4Rec achieves an average relative improvement of **17.24%** and **18.7%** in terms of Recall@10 and NDCG@10, respectively, across the four datasets. Similarly, AC-SASRec demonstrates average relative improvements of **6.05%** and **5.43%** for Recall@10 and NDCG@10, respectively. Comparable results are also observed in AC-TiSASRec and AC-SSE-PT. These results demonstrate the effectiveness of our proposed method and its potential as a plug-in module for state-of-the-art transformer-based recommendation models.

In Sec. 4.5, we discussed the impact of incorporating two calibrators into transformer layers, which inevitably increases computational cost and reduces inference speed. This presents a challenge for implementing AC-TSR in industrial scenarios. To mitigate this issue, we propose a lightweight version of AC-TSR, called AC-TSR-lite. The key distinction between the lite version and the original AC-TSR lies in their usage of calibrators. In AC-TSR-lite, calibrators are employed only during the model training phase and removed during the inference phase, thereby maintaining the same structure as TSR. To illustrate this, we compare the performance of SASRec, AC-SASRec, and AC-SASRec-lite on four datasets in Tab.2. We observe that in most cases, although AC-SASRec-lite exhibits lower Recall@20 compared to AC-SASRec, it still consistently outperforms SASRec and retains the same inference speed as SASRec. This provides evidence supporting the feasibility and flexibility of deploying AC-TSR in an industrial setting.

5.3 Ablation and Hyper-parameter Studies (RQ3)

5.3.1 Contribution of Different Components. As shown in Table 3, we investigate 8 settings from a combination of (1) Whether to include order information in the Spatial Calibrator, (2) Whether to include distance information in the Spatial Calibrator, and (3) Whether to incorporate the Adversarial Calibrator. We can draw the following conclusions: First, by comparing the results of *setting (F)* and *setting (G)* with *setting (H)*, which represents the initial SASRec, we can see that both the inclusion of order information and distance information in the Spatial Calibrator improve the model's performance. Combining order information and distance information together leads to even better performance compared to using them separately, thus validating the rationale behind our proposed Spatial Calibrator. Second, models equipped with our proposed Adversarial Calibrator consistently outperform those without it, while keeping other factors constant. For instance, we can compare the performance between *setting (A)* and *setting (E)*, or between *setting (B)* and *setting (F)*. This observation confirms that our Adversarial Calibrator effectively adjusts the attention weights in transformer-based SR models, resulting in improved performance.

5.3.2 Comparison Between Spatial Calibrator and Different Positional Encoding Strategies. We test 4 representative position embedding approaches and our proposed spatial calibrator using SASRec as the backbone. Due to limited space, we only present the comparison results on the Sports and Toys datasets in Tab. 4. To our surprise, we find that removing the position embedding from SASRec does not compromise the recommendation performance, but rather outperforms some methods using position embedding. For instance, on the Sports dataset, removing the position embedding achieves higher Recall@20 than the three methods using position embedding. On the Toys dataset, both relative position embedding and removing positions achieved competitive performance. Nonetheless, our proposed spatial calibrator still outperforms all compared position encoding strategies. We attribute this to the low-level features, including order and distance, being more effectively utilized by the self-attention mechanism, especially in noisy input scenarios.

5.3.3 Comparison Between Gating Mechanism and Other Aggregation Strategies. We also investigate the impact of various attention aggregation strategies used in the correction module. In particular, we compare the gating mechanism with two commonly employed

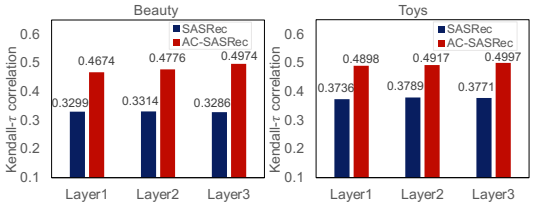

Figure 5: Comparison of the mean Kendall-τ correlation between attention weights and gradient importance measures. The results verify that our AC method can improve Kendall-τ correlation by a large margin.

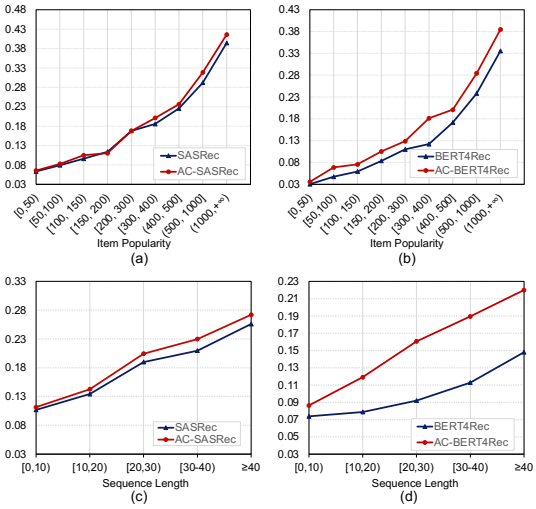

Figure 6: Performance comparison (Recall@20) between AC-TSR and TSR under different sequence lengths (*i.e.*, number of training interactions of users) and item popularity (*i.e.*, number of training interactions of items) on Amazon Beauty.

aggregation strategies: summation [52] and annealing learning [27]. The comparison results are presented in Fig. 3. We can observe that the gating mechanism consistently outperforms the other two strategies in most cases, which can be attributed to its ability to dynamically control the proportions of original and calibrated attention weights. As a result, we ultimately chose the gating mechanism as the attention aggregation strategy in the correction module.

5.3.4 Hyper-parameter Studies. In this experiment, we aim to study the impact of hyper-parameter α in Eq. 20. Fig.4 presents the Recall@10 and NDCG@10 scores of AC-TSR with different α on Toys and Sports datasets. From the figures, we can observe that the best performance is achieved when the balance parameter α is set to 0.03 or 0.05. Moreover, we observe that the impact of the balance parameter α varies across different datasets. Specifically, the performance fluctuates within a narrower range on the Sports dataset compared to the Toys dataset.

5.4 In-depth Analysis (RQ4)

In this section, we further validate the effectiveness of our method from three different perspectives, including visualization, Kendall-τ correlations, and the performance comparison of AC-TSR across different sequence lengths and item popularity.

Visualization and "erasing" experiment. We visually compare the original attention weights with the calibrated ones generated by AC-TSR in Fig. 1(b). It can be observed that the calibrated attention better aligns with the user's interests, resulting in improved recommendations. Moreover, Fig. 1(a) indicates that removing the highest scoring items in the AC-TSR model noticeably degrades the model's performance on all the datasets, demonstrating that the learned attention weights in AC-TSR effectively aligns with the actual importance of items in the recommendation process.

Correlation between attention weights and feature importance metrics. Another way to examine whether attention mechanism focus on decisive inputs is computing the Kendall-τ correlation between attention weights and feature importance measures [15]. Here we use gradient-based importance measures as feature importance indicators because they can effectively showcase feature significance with well-defined semantics [18]. A lower value of Kendall-τ correlation indicates a higher inconsistency between attention weights and feature importance measures. This implies that items with high attention weights may not contribute significantly to the model's predictions [35]. As shown in Fig. 5, for each layer, our AC-TSR approach significantly improves the correlation score, implying that attention calibration can help model focus on the decisive items.

Performance w.r.t sequence lengths and item popularity. We compare the original TSR model and our proposed AC-TSR model under varying sequence lengths and item popularity on the Amazon Beauty dataset, using Recall@20 as the performance metric. As shown in Fig 6, the enhancements brought by AC-TSR over TSR are consistently observed across diverse sequence lengths and item popularity levels. We also note that the performance gains from Attention Calibration increase as the sequence length in the test set becomes longer. As for item popularity, the most prominent performance improvement is observed when the item popularity falls between 300 and 400.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose the AC-TSR framework, which can effectively calibrate the unreliable attention weights generated by existing transformer-based SR models. Specifically, AC-TSR adopts the spatial calibrator to substitute traditional positional embeddings, which directly utilizes low-level features including order and distance to yield position-aware attention weights. Additionally, the adversarial calibrator is devised to adjust the attention weights according to the reassessed contribution of each historical item to the model prediction, making attention weights more robust to noisy input. Comprehensive experiments on four benchmark SR datasets show that our approach outperforms competitive transformer-based SR methods, demonstrating the effectiveness of AC-TSR. In the future, we intend to investigate more lightweight calibrators and explore treating each calibrator as an adaptor to be incorporated into pre-trained transformer-based SR models, thereby achieving performance improvements with minimal computational expense.

REFERENCES

[1] Jianxin Chang, Chen Gao, Yu Zheng, Yiqun Hui, Yanan Niu, Yang Song, Depeng Jin, and Yong Li. 2021. Sequential recommendation with graph neural networks. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*. 378–387.

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

[3] Nuo Chen, Fenglin Liu, Chenyu You, Peilin Zhou, and Yuexian Zou. 2021. Adaptive bi-directional attention: Exploring multi-granularity representations for machine reading comprehension. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 7833–7837.

[4] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential recommendation with user memory networks. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 108–116.

[5] Qianfeng Chu, Gongshen Liu, Huanrong Sun, and Cheng Zhou. 2019. Next news recommendation via knowledge-aware sequential model. In *Chinese Computational Linguistics: 18th China National Conference, CCL 2019, Kunming, China, October 18–20, 2019, Proceedings 18*. Springer, 221–232.

[6] Alexander Dallmann, Daniel Zoller, and Andreas Hotho. 2021. A Case Study on Sampling Strategies for Evaluating Neural Sequential Item Recommendation Models. In *Fifteenth ACM Conference on Recommender Systems*. 505–514.

[7] 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* (Virtual Event, Canada) *(SIGIR '21)*. Association for Computing Machinery, New York, NY, USA, 1733–1737. https://doi.org/10.1145/3404835.3462978

[8] Jiayan Guo, Peiyan Zhang, Chaozhuo Li, Xing Xie, Yan Zhang, and Sunghun Kim. 2022. Evolutionary Preference Learning via Graph Nested GRU ODE for Session-based Recommendation. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. 624–634.

[9] Ido Guy, Naama Zwerdling, Inbal Ronen, David Carmel, and Erel Uziel. 2010. Social media recommendation based on people and tags. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. 194–201.

[10] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *ICDM*. IEEE, 191–200.

[11] Yun He, Yin Zhang, Weiwen Liu, and James Caverlee. 2020. Consistency-Aware Recommendation for User-Generated Item List Continuation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 250–258.

[12] Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. 2021. Locker: Locally Constrained Self-Attentive Sequential Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3088–3092.

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

[14] Hyunwoo Hwangbo, Yang Sok Kim, and Kyung Jin Cha. 2018. Recommendation system development for fashion retail e-commerce. *Electronic Commerce Research and Applications* 28 (2018), 94–101.

[15] Sarthak Jain and Byron C Wallace. 2019. Attention is not explanation. *arXiv preprint arXiv:1902.10186* (2019).

[16] Juyong Jiang, Jae Boum Kim, Yingtao Luo, Kai Zhang, and Sunghun Kim. 2022. AdaMCT: adaptive mixture of CNN-transformer for sequential recommendation. *arXiv preprint arXiv:2205.08776* (2022).

[17] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[18] Pieter-Jan Kindermans, Sara Hooker, Julius Adebayo, Maximilian Alber, Kristof T Schütt, Sven Dähne, Dumitru Erhan, and Been Kim. 2019. The (un) reliability of saliency methods. *Explainable AI: Interpreting, explaining and visualizing deep learning* (2019), 267–280.

[19] Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. 2019. Revealing the dark secrets of BERT. *arXiv preprint arXiv:1908.08593* (2019).

[20] 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.

[21] Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. Formnet: Structural encoding beyond sequential modeling in form document information extraction. *arXiv preprint arXiv:2203.08411* (2022).

[22] Yuxuan Lei, Xiaolong Chen, Defu Lian, Peiyan Zhang, Jianxun Lian, Chaozhuo Li, and Xing Xie. 2023. Practical Content-aware Session-based Recommendation: Deep Retrieve then Shallow Rank. In *Amazon KDD Cup 2023 Workshop*.

[23] 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. https://doi.org/10.1145/3336191.3371786

[24] 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.

[25] Defu Lian, Yongji Wu, Yong Ge, Xing Xie, and Enhong Chen. 2020. Geography-aware sequential location recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 2009–2019.

[26] Junling Liu, Chao Liu, Peilin Zhou, Renjie Lv, Kang Zhou, and Yan Zhang. 2023. Is chatgpt a good recommender? a preliminary study. *arXiv preprint arXiv:2304.10149v2* (2023).

[27] Yu Lu, Jiali Zeng, Jiajun Zhang, Shuangzhi Wu, and Mu Li. 2021. Attention calibration for transformer in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 1288–1298.

[28] Chen Ma, Peng Kang, and Xue Liu. 2019. Hierarchical gating networks for sequential recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 825–833.

[29] Julian J. 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, August 9-13, 2015*, Ricardo Baeza-Yates, Mounia Lalmas, Alistair Moffat, and Berthier A. Ribeiro-Neto (Eds.). ACM, 43–52. https://doi.org/10.1145/2766462.2767755

[30] Akash Kumar Mohankumar, Preksha Nema, Sharan Narasimhan, Mitesh M Khapra, Balaji Vasan Srinivasan, and Balaraman Ravindran. 2020. Towards transparent and explainable attention models. *arXiv preprint arXiv:2004.14243* (2020).

[31] Bo Peng, Zhiyun Ren, Srinivasan Parthasarathy, and Xia Ning. 2020. HAM: hybrid associations models for sequential recommendation. *arXiv preprint arXiv:2002.11890* (2020).

[32] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing session-based recommendations with hierarchical recurrent neural networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 130–137.

[33] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Montreal, Quebec, Canada) *(UAI '09)*. AUAI Press, Arlington, Virginia, USA, 452–461.

[34] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *WWW*. 811–820.

[35] Andrew Slavin Ross, Michael C Hughes, and Finale Doshi-Velez. 2017. Right for the right reasons: training differentiable models by constraining their explanations. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 2662–2670.

[36] Sofia Serrano and Noah A Smith. 2019. Is attention interpretable? *arXiv preprint arXiv:1906.03731* (2019).

[37] Wenzhuo Song, Shoujin Wang, Yan Wang, and Shengsheng Wang. 2021. Next-item recommendations in short sessions. In *Proceedings of the 15th ACM Conference on Recommender Systems*. 282–291.

[38] 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.

[39] Yatong Sun, Bin Wang, Zhu Sun, and Xiaochun Yang. 2021. Does Every Data Instance Matter? Enhancing Sequential Recommendation by Eliminating Unreliable Data.. In *IJCAI*. 1579–1585.

[40] 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.

[41] 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 *Advances in neural information processing systems*. 5998–6008.

[42] Nan Wang, Shoujin Wang, Yan Wang, Quan Z Sheng, and Mehmet A Orgun. 2022. Exploiting intra-and inter-session dependencies for session-based recommendations. *World Wide Web* 25, 1 (2022), 425–443.

[43] Shoujin Wang, Longbing Cao, Yan Wang, Quan Z Sheng, Mehmet A Orgun, and Defu Lian. 2021. A survey on session-based recommender systems. *ACM Computing Surveys (CSUR)* 54, 7 (2021), 1–38.

[44] Shoujin Wang, Liang Hu, Longbing Cao, Xiaoshui Huang, Defu Lian, and Wei Liu. 2018. Attention-based transactional context embedding for next-item recommendation. In *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*. Association for the Advancement of Artificial Intelligence, 2532–2539.

[45] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z. Sheng, and Mehmet Orgun. 2019. Sequential recommender systems: challenges, progress and prospects. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 6332–6338.

[46] Shoujin Wang, Gabriella Pasi, Liang Hu, and Longbing Cao. 2020. The era of intelligent recommendation: editorial on intelligent recommendation with advanced AI and learning. *IEEE Intelligent Systems* 35, 5 (2020), 3–6.

[47] Shoujin Wang, Xiaofei Xu, Xiuzhen Zhang, Yan Wang, and Wenzhuo Song. 2022. Veracity-aware and event-driven personalized news recommendation for fake news mitigation. In *Proceedings of the ACM Web Conference 2022*. 3673–3684.

[48] Shoujin Wang, Xiuzhen Zhang, Yan Wang, Huan Liu, and Francesco Ricci. 2022. Trustworthy recommender systems. *arXiv preprint arXiv:2208.06265* (2022).

[49] 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* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 328–337. https://doi.org/10.1145/3383313.3412258

[50] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 346–353.

[51] Yueqi Xie, Jingqi Gao, Peilin Zhou, Qichen Ye, Yining Hua, Jaeboum Kim, Fangzhao Wu, and Sunghun Kim. 2023. Rethinking Multi-Interest Learning for Candidate Matching in Recommender Systems. *arXiv preprint arXiv:2302.14532* (2023).

[52] Yueqi Xie, Peilin Zhou, and Sunghun Kim. 2022. Decoupled Side Information Fusion for Sequential Recommendation. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*.

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

[54] Ghim-Eng Yap, Xiao-Li Li, and Philip S Yu. 2012. Effective next-items recommendation via personalized sequential pattern mining. In *International conference on database systems for advanced applications*. Springer, 48–64.

[55] Chenyu You, Nuo Chen, and Yuexian Zou. 2020. Contextualized attention-based knowledge transfer for spoken conversational question answering. *arXiv preprint arXiv:2010.11066* (2020).

[56] Chenyu You, Nuo Chen, and Yuexian Zou. 2021. MRD-Net: Multi-Modal Residual Knowledge Distillation for Spoken Question Answering.. In *IJCAI*. 3985–3991.

[57] Chenyu You, Ruihan Zhao, Fenglin Liu, Siyuan Dong, Sandeep Chinchali, Ufuk Topcu, Lawrence Staib, and James Duncan. 2022. Class-aware adversarial transformers for medical image segmentation. *Advances in Neural Information Processing Systems* 35 (2022), 29582–29596.

[58] 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.

[59] Xu Yuan, Dongsheng Duan, Lingling Tong, Lei Shi, and Cheng Zhang. 2021. ICAI-SR: Item Categorical Attribute Integrated Sequential Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1687–1691.

[60] Peiyan Zhang, Jiayan Guo, Chaozhuo Li, Yueqi Xie, Jae Boum Kim, Yan Zhang, Xing Xie, Haohan Wang, and Sunghun Kim. 2023. Efficiently leveraging multi-level user intent for session-based recommendation via atten-mixer network. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 168–176.

[61] Peiyan Zhang and Sunghun Kim. 2023. A Survey on Incremental Update for Neural Recommender Systems. *arXiv preprint arXiv:2303.02851* (2023).

[62] Peiyan Zhang, Yuchen Yan, Chaozhuo Li, Senzhang Wang, Xing Xie, Guojie Song, and Sunghun Kim. 2023. Continual Learning on Dynamic Graphs via Parameter Isolation. *arXiv preprint arXiv:2305.13825* (2023).

[63] 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.

[64] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1893–1902.

[65] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Filter-enhanced MLP is all you need for sequential recommendation. In *Proceedings of the ACM Web Conference 2022*. 2388–2399.

[66] Peilin Zhou, Jingqi Gao, Yueqi Xie, Qichen Ye, Yining Hua, and Sunghun Kim. 2022. Equivariant Contrastive Learning for Sequential Recommendation. *arXiv preprint arXiv:2211.05290* (2022).