AdaMCT: Adaptive Mixture of CNN-Transformer for Sequential Recommendation

Juyong Jiang[*†]
The Hong Kong University of Science and Technology (Guangzhou)
csjuyongjiang@gmail.com

Peiyan Zhang[*]
The Hong Kong University of Science and Technology
pzhangao@cse.ust.hk

Yingtao Luo
Carnegie Mellon University
yingtaoluo@cmu.edu

Chaozhuo Li[‡]
Microsoft Research Asia
cli@microsoft.com

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

Kai Zhang[‡]
East China Normal University
kzhang@cs.ecnu.edu.cn

Senzhang Wang
Central South University
szwang@csu.edu.cn

Xing Xie
Microsoft Research Asia
xing.xie@microsoft.com

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

ABSTRACT

Sequential recommendation (SR) aims to model users' dynamic preferences from a series of interactions. A pivotal challenge in user modeling for SR lies in the inherent variability of user preferences. An effective SR model is expected to capture both the long-term and short-term preferences exhibited by users, wherein the former can offer a comprehensive understanding of stable interests that impact the latter. To more effectively capture such information, we incorporate locality inductive bias into the Transformer by amalgamating its global attention mechanism with a local convolutional filter, and adaptively ascertain the mixing importance on a personalized basis through layer-aware adaptive mixture units, termed as AdaMCT. Moreover, as users may repeatedly browse potential purchases, it is expected to consider multiple relevant items concurrently in long-/short-term preferences modeling. Given that softmax-based attention may promote unimodal activation, we propose the Squeeze-Excitation Attention (with sigmoid activation) into SR models to capture multiple pertinent items (keys) simultaneously. Extensive experiments on three widely employed benchmarks substantiate the effectiveness and efficiency of our proposed approach. Source code is available at https://github.com/juyongjiang/AdaMCT.

CCS CONCEPTS

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

[*]Equal contribution.
[†]Work done when Juyong was AI Global Residency of Upstage.
[‡]Corresponding authors.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '23, October 21–25, 2023, Birmingham, United Kingdom
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0124-5/23/10…$15.00
https://doi.org/10.1145/3583780.3614773

KEYWORDS

Sequential Recommendation; CNNs; Transformer

ACM Reference Format:
Juyong Jiang, Peiyan Zhang, Yingtao Luo, Chaozhuo Li, Jae Boum Kim, Kai Zhang, Senzhang Wang, Xing Xie, and Sunghun Kim. 2023. AdaMCT: Adaptive Mixture of CNN-Transformer for 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, 13 pages. https://doi.org/10.1145/3583780.3614773

1 INTRODUCTION

Recommender systems facilitate the provision of tailored information to users, taking into account their preferences as evidenced by historical interactions [12], which are widely applied in e-commerce websites, web searches, and so forth [10, 65]. Analogous with words of sentences, a user's interactions with items naturally constitute a behavioral sequence. Thus, sequential recommendation (SR) has been proposed to encapsulate the sequential characteristics of user behaviors, thereby modeling users' preferences more effectively [19, 25].

A vital challenge of user modeling for SR lies in the inherent variability of user preferences. In large-scale recommender systems, millions of new items are generated and incorporated into the candidate pool on a daily basis [63]. It is imperative for recommender systems to promptly and accurately discern users' short-term preferences, as their attention is readily captivated by emerging trends and popular topics. In contrast, users' long-term preferences offer a valuable insight into their aggregated, stable interests, effectively complementing their short-term interests. Hence, it is crucial for SR models to adeptly capture both the dynamic nature of short-term preferences and the enduring stability of long-term preferences exhibited by users.

Fortunately, recent studies in SR have endeavored on this target by modeling and integrating long- and short-term preferences prior to making recommendations. The initial line of research [9, 49, 62]

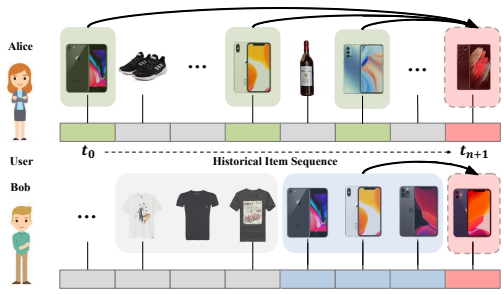


Figure 1: Personalized global interactions and local features dependency. Alice and Bob have different interest patterns. For Alice, the next item relies more on her global interactions interest, while for Bob, he shows a local features dependency on his previous historical item sequences.

concentrates on capturing long-term preferences while employing the most recently interacted item as a representation of short-term preferences. However, due to the SR-related inductive bias, specifically local invariance [4], which suggests that the local order of the last few items is inconsequential, it becomes arduous to comprehensively represent a user's short-term interests using only the final item. Furthermore, these works merely fuse the long-/short-term preferences through summation or concatenation [4, 18, 21, 32, 55, 56, 58]. As a result, these approaches fail to obtain personalized weights for long-/short-term preferences pertaining to different items for each user.

Another line of works adopts a parallel paradigm to model the long-/short-term preferences respectively with recurrent and attentive models [23, 36, 66] or even explicit user behavior graphs [53]. Subsequently, they implement a gating mechanism [11] to facilitate the integration of personalized long-term and short-term preferences. Nevertheless, the long-/short-term preference modeling processes are segregated. Given that users' long-term preferences can offer a comprehensive understanding of stable interests that influence short-term interests and vice versa [1, 8, 59], this paradigm may adversely affect the comprehension of users' overall preferences. Therefore, it is reasonable that the long-/short-term preferences should be mutually reinforced in the context of user modeling.

In particular, modeling the long-/short-term preferences in SR is non-trival, primarily encountering the following three challenges: (1) *How can long-term and short-term preferences be modeled conjointly?* As users' long-term preferences could provide users' aggregated stable interests that influence the short-term interests and vice versa [53], these preferences should engage in a mutual reinforcement within user modeling. (2) *How to integrate the short-term and long- term preferences effectively?* Both types of interests play a crucial role in real-world recommendations, despite the inherent biases between them. Models must be capable of discerning which interest is more pertinent in varying situations. Previous studies [4, 21, 53, 55] have relied on simple summation, concatenation, or a singular gating mechanism, leading to restricted model capacity. (3) *How can multiple relevant items be considered concurrently in long-term and short-term preference modeling?* As illustrated in Figure 1, users may repeatedly browse and compare potential purchases, resulting in more than one item being highly correlated with their preferences. However, the softmax activation in the attention scheme necessitates that the sum of all item scores equals 1, creating mutually exclusive relations [22] that impede the simultaneous activation of multiple highly-related items. These three challenges are of significant importance in real-world scenarios, yet there is a scarcity of research that systematically addresses them in unison.

To address these challenges, we introduce a novel Adaptive Mixture of CNN-Transformer for Sequential Recommendation (AdaMCT) that explicitly learns personalized behavior preferences to adaptively leverage locality inductive bias (CNNs) [47] and global interactions (Transformer) [28, 45] modeling. Specifically, we employ a local-global dependencies mechanism comprising both (local) convolutional layers and (global) self-attention layers, facilitating the joint modeling of long-term and short-term preferences. The CNN layer exhibits heightened sensitivity to short-term interest patterns, while the self-attention layer is better suited for identifying long-term dependency patterns. To effectively integrate long-term and short-term preferences, we devise adaptive mixture units that decouple the fusion process into multiple layers, thereby enhancing expressibility and adaptively aggregating long-term and short-term preferences for personalized user modeling. Furthermore, we propose the Squeeze-Excitation Attention to generate attention scores for both short-term and long-term information branches. Squeeze-Excitation Attention eliminates the sum-up-to-1 constraint, allowing for the consideration of multiple relevant purchases simultaneously and enhancing model expressibility. We validate the AdaMCT approach on three public benchmarks. Substantial improvements across multiple indicators demonstrate the efficacy of the proposed methods. The main contributions of this paper are as follows:

- We incorporate the locality inductive bias (CNNs) into the Transformer structure to model the long-/short-term preferences conjointly, and devise the adaptive mixture units that decouple the fusion process into multiple layers, thereby enhancing expressibility and adaptively aggregating long-term and short-term preferences for personalized user modeling.
- We propose Squeeze-Excitation Attention (SEAtt) to replace the softmax operation in both local and global information branches to simultaneously take into account multiple relevant purchases to effectively improve the representation learning of sequential dependencies.
- Extensive experiments on three public datasets demonstrate that our proposed solution achieves comparable (or better) performance than the previous Transformer and CNNs-based models. Ablation studies show the efficacy of our proposed components.

2 RELATED WORK

2.1 Sequential Recommendation

Sequential recommendation aims to predict the future item or item list based on a historical user behavior sequence. In the literature, early works mainly use Markov Chain models [16, 29, 42] that make personalized next recommendations conditioned on the previous one's behavior. These methods only capture adjacent local

sequential patterns. Later on, the genre of neural network-based models starts to rise. Recurrent Neural Networks (RNNs) improve the recommendation performance in many tasks with sequential dependencies and memory mechanisms [7, 34, 35, 57]. RNNs impose strict sequential order in representation learning and face the global dependency problem when the sequence is too long. Meanwhile, Convolutional Neural Networks (CNNs) are used to capture local features across the sequence, emphasizing the impact of more recent behaviors [47, 61]. More recently, attentive modules are extensively studied in the sequential recommendation venue, resulting in extraordinary performances [27, 28]. Self-attention models allow point-to-point feature interaction within the item sequence, which solves the global dependency problem and uses a longer sequence with more information [26, 28, 38]. Meanwhile, graph neural networks have been attempted to discover the graph-based topological interaction of sequences [40, 50, 52]. Transformer and Bidirectional Transformer (BERT) that combine multiple blocks including attention layers, feed-forward networks, and position embeddings also report state-of-the-art performances in sequential recommendation tasks [3, 45, 51]. Mixed models [4, 55, 60] that consider both global and local preferences have also received extensive attention in recent years.

2.2 Local-Global Awareness

Local-global awareness is a topic that has raised discussions in the venue of recommendation systems. Many prior works [24, 39, 44] propose and design local-global memory mechanisms to learn different-level item sequences, while they use a simple concatenation of neural modules without an adaptive learnable preference factor. However, some works [18, 31] also consider the adaptive mixture of local-global representation learning and show some advantages, but the difference from previous works is that we first attempt to adopt CNNs and Transformer to capture local and global dependencies in SR. Moreover, Local-global regularization is used to learn the local and global item correlations with both deep neural network and matrix factorization [33, 64]. These works design user-specific strategies to fuse local and global information into the recommendation model [6, 46]. Other works focus on improving neural architectures. Global and local sparse linear method models are combined to make recommendations for different user preferences [21]. A recurrent convolutional network is proposed to learn complex global sequential dependency with RNNs and extract local dependency using CNNs [55], which marks a combined use of sequence order bias and locality bias. An encoder for modeling the global cross-session dependency is proposed to improve the self-attention model [56]. Local generative models that capture different user preferences are combined to make a global prediction [32]. Local invariance is considered in session-based recommendation [4] as extra local information to substitute the main model. Attention and graph network are combined to model different user interests [58] with graph topology inductive bias. Recently, some papers [37, 60] find that the personalized and adaptive use of model components can effectively improve the model performance on RNNs. However, most works combining multiple neural modules, especially the state-of-the-art Transformer, still rarely consider the user-specific learnable adaptation. Moreover, they hardly consider the comprehensive inductive biases from locality to global dependency, and from sequential order to positional equivalence.

3 METHODOLOGY

3.1 Problem Statement

$\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ denotes users, $\mathcal{V} = \{v_1, v_2, ..., v_{|\mathcal{V}|}\}$ denotes items, and $\mathcal{S}^{(u)}_{1:n} = [v^{(u)}_1, v^{(u)}_2, ..., v^{(u)}_n]$ denotes the interaction sequence in chronological order of user $u \in \mathcal{U}$. Each $v^{(u)}_i \in \mathcal{V}$ refers to an item that u has interacted with at time step i. Sequential recommendation aims to predict the item that user u will interact with at the $(n+1)$th timestamp. The conditional probability of each item $v \in \mathcal{V}$ becoming the $(n+1)$th item is denoted as $\mathcal{P}_\theta(v^{(u)}_{n+1} = v|\mathcal{S}^{(u)}_{1:n})$, where θ represents the model parameters.

3.2 Model Architecture

The overall model architecture is shown in Figure 2. Our proposed AdaMCT consists of 1) an embedding module that learns the dense representations of items, including item embeddings and positional embeddings; 2) a multi-stacked Adaptive Mixture of CNN-Transformer blocks, in which both comprise a global attention module (Transformer) that learns the global user interest preferences with squeeze-excitation attention (SEAtt), a local convolutional module (CNNs) that learns user local interest pattern with SEAtt, and a module- and layer-aware adaptive mixture units, that makes a personalized balanced use of local and global dependencies modules; and 3) an output layer with cross-entropy loss that computes the matching probability.

3.3 Embedding Module

The embedding module contains an item embedding layer and a positional embedding layer. The item embedding layer $E_{item} \in \mathbb{R}^{|\mathcal{V}| \times d_{model}}$ with the latent dimension of d_{model} encodes each scalar item ID $v^{(u)}_i \in \mathbb{R}$ into a dense latent vector representation $e^{(u)}_i \in \mathbb{R}^{d_{model}}$, aiming to reduce computation and improve representation learning. The positional embeddings $\mathcal{P}os \in \mathbb{R}^{n \times d_{model}}$ we follow [28] to employ learnable embeddings and maintain the same embedding dimensions of the item embedding. The overall embedding module transforms the sequence of $\mathcal{S}^{(u)}_{1:n} = [v^{(u)}_1, v^{(u)}_2, ..., v^{(u)}_n] \in \mathbb{R}^n$ into $E^{pos}_{item}(\mathcal{S}^{(u)}_{1:n}) = [E_{item}(\mathcal{S}^{(u)}_{1:n}) + \mathcal{P}os(\mathcal{S}^{(u)}_{1:n})] \in \mathbb{R}^{n \times d_{model}}$.

3.4 Adaptive Mixture of CNN-Transformer

AdaMCT has three sub-layers, a global attention module (Transformer) with SEAtt referred to as $\mathcal{G}_{\text{Trm}}$, a local convolutional module (CNNs) with SEAtt referred to as $\mathcal{L}_{\text{Conv}}$, and adaptive mixture units referred to as $\mathcal{A}_{\text{Mix}}$, as shown in the middle of Figure 2.

3.4.1 Global Attention Module. As shown in Figure 2, we first utilize a linear layer, followed by layer normalization [2], to adjust latent dimensionality and enhance the non-linear representation. Then, a multi-head self-attention layer (MHSA) followed by layer normalization is adopted to capture global dependencies of sequence. Finally, a squeeze-excitation attention layer (SEAtt) to re-weight the importance of each item from the sequence level.

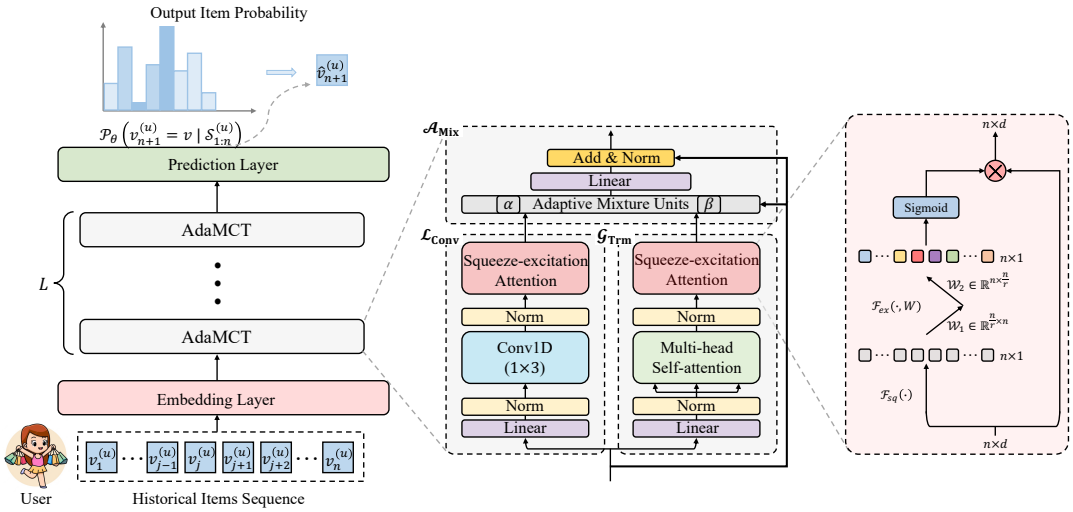


Figure 2: The model architecture of the proposed AdaMCT. AdaMCT first generates item embedding with positional embedding by embedding layer, then aggregates sequence representations through $L\times$ AdaMCT layers, in which each layer includes three sub-layers, *i.e.*, global attention module ($\mathcal{G}_{\text{Trm}}$), local convolutional module ($\mathcal{L}_{\text{Conv}}$), and adaptive mixture units ($\mathcal{A}_{\text{Mix}}$). Finally, a prediction layer generates a recommendation score for each candidate item.

Formally,

$$\mathcal{G}^{l}_{\text{Trm}} = \text{SEAtt}(\text{MHSA}(\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)})) \otimes \text{MHSA}(\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)}))), \tag{1}$$

$$\text{MHSA}(\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)})) = \text{LN}(\text{Dpt}(Conat(\text{head}_1, ..., \text{head}_h)\mathcal{W}^{\mathcal{O}})), \tag{2}$$

$$\text{head}_i = \text{Att}(\underbrace{\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)})\mathcal{W}^{\mathcal{Q}}_i}_{\mathcal{Q}}, \underbrace{\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)})\mathcal{W}^{\mathcal{K}}_i}_{\mathcal{K}}, \underbrace{\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)})\mathcal{W}^{\mathcal{V}}_i}_{\mathcal{V}}), \tag{3}$$

$$\text{Att}(\mathcal{Q}, \mathcal{K}, \mathcal{V}) = \text{softmax}(\frac{\mathcal{Q}\mathcal{K}^{\mathcal{T}}}{\sqrt{d_{model}/h}} + \mathcal{M}_{mask})\mathcal{V}, \tag{4}$$

$$\mathcal{E}^{en}_{global}(\mathcal{H}^{l-1}_{(u)}) = \text{LN}(\text{Dpt}(\mathcal{L}inear_{en}(\mathcal{H}^{l-1}_{(u)}))), \mathcal{H}^{0}_{(u)} = \mathcal{E}(\mathcal{S}^{(u)}_{1:n}). \tag{5}$$

$$\mathcal{M}_{mask} = [a_{ij}] = \begin{cases} a_{ij} = -\infty & \text{for } i < j \\ a_{ij} = 0 & \text{otherwise} \end{cases} \tag{6}$$

where $\mathcal{G}_{\text{Trm}} \in \mathbb{R}^{n\times d_{model}}$ is the output of the l-th global attention module, $\text{SEAtt}(\cdot) \in \mathbb{R}^{n}$ denotes sequence-level attention scores. The details of SEAtt module will be discussed in the following section. "$\otimes$" denotes the element-wise operation with the broadcasting mechanism. $\mathcal{H}^{l-1}_{(u)} \in \mathbb{R}^{n\times d_{model}}$ represents the $(l-1)$-th AdaMCT layer's output, LN and Dpt denote layer normalization and dropout operations, respectively. h represents the number of attention layers. $\mathcal{W}^{\mathcal{Q}}_i \in \mathbb{R}^{d_{model}\times d_{model}/h}$, $\mathcal{W}^{\mathcal{K}}_i \in \mathbb{R}^{d_{model}\times d_{model}/h}$, $\mathcal{W}^{\mathcal{V}}_i \in \mathbb{R}^{d_{model}\times d_{model}/h}$, and $\mathcal{W}^{\mathcal{O}}_i \in \mathbb{R}^{d_{model}\times d_{model}/h}$ are the parameters that are unique per layer and attention head and used to transform the Query $\mathcal{Q}$, Key $\mathcal{K}$, Value $\mathcal{V}$, and the output of MHSA layer. The effect of "temperature" $\sqrt{d_{model}/h}$ is to avoid overly large values of the inner product and extremely small gradients to produce a softer attention distribution [28, 45, 49]. For a casual attention mask $\mathcal{M}_{mask}$, the upper triangular part of it is set to $-\infty$ otherwise 0, representing each item only attends to itself and preceding items.

3.4.2 Local Convolutional Module. This module is shown in the left part of the AdaMCT in the middle of Figure 2. Similar to the global attention module ($\mathcal{G}_{\text{Trm}}$), each local convolutional module ($\mathcal{L}_{\text{Conv}}$) first utilize a linear layer, followed by layer normalization [2], to adjust latent dimensionality and enhance the non-linear representation. Formally,

$$\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}) = \text{LN}(\text{Dpt}(\mathcal{L}inear_{en}(\mathcal{H}^{l-1}_{(u)}))), \mathcal{H}^{0}_{(u)} = \mathcal{E}(\mathcal{S}^{(u)}_{1:n}), \tag{7}$$

Subsequently, we adopt a 1D convolutional layer with $k = 3$ kernel size followed by layer normalization on $\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}) \in \mathbb{R}^{n\times d_{model}}$. Formally, considering m 1D convolutional filters $\mathcal{F}^{i} \in \mathbb{R}^{k\times d_{model}}$, $i \in \{1, 2, ..., m\}$ with k kernal size, $1 \le k \le n$. $\mathcal{F}^{i}$ will slide from left to right to interact $\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)})$ with every successive k item along the size (length)-axis. Notably, we use the zero paddings and stride to 1 to maintain the size (length) dimension invariant in the convolutional layer. Thus, after sliding in all the positions $j \in \{1, 2, ..., n\}$, the final output of 1D convolutional layer can be formalized by

$$\text{Conv}^{k=3}_{mid}(\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)})) = \text{LN}(\text{Dpt}([\text{Conv}^{k=3}_{mid}(\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}))_1, ..., \text{Conv}^{k=3}_{mid}(\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}))_j, ..., \text{Conv}^{k=3}_{mid}(\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}))_n])) \tag{8}$$

$$\text{Conv}^{k=3}_{mid}(\mathcal{E}^{en}_{local}(\mathcal{H}^{l-1}_{(u)}))_j = Concat(C^1_j, ..., C^m_j) \in \mathbb{R}^{m}, \tag{9}$$

$$C_j^i = \Phi(\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1})_{[j:j+k-1]}\mathcal{F}^i), \tag{10}$$

where $\Phi(\cdot)$ denotes the activation function of convolutional layers in which we use the *ReLU* by default. $\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1})_{[j:j+k-1]} \in \mathbb{R}^{k\times d_{model}}$ represents the sub-sequence with indexes from j to $j+k-1$. $\text{Conv}_{mid}^{k=3}(\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1}))_j \in \mathbb{R}^m$ denotes the convolutional value in the position j.

Furthermore, we input the above output of convolutional layer to the SEAtt module to re-weight the importance of each item from the sequence level. Formally,

$$\mathcal{L}_{\text{Conv}} = \text{SEAtt}(\text{Conv}_{mid}^{k=3}(\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1}))) \otimes \text{Conv}_{mid}^{k=3}(\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1})). \tag{11}$$

where $\mathcal{L}_{\text{Conv}} \in \mathbb{R}^{n\times m}$ represents the output of local convolutional module. In this work, we set $m = d_{model}$ to make L_{Conv} have the same dimension as $\mathcal{G}_{\text{Trm}}$, *i.e.*, $\mathcal{L}_{\text{Conv}} \in \mathbb{R}^{n\times d_{model}}$.

3.4.3 Squeeze-Excitation Attention. In practical scenarios, users' behaviors are dynamic over time which leads to more than one historical user-item interaction having a high correlation to the next item. However, many previous works [9, 13] neglect the impact of *multi-high-related* user-item interactions on the next item, which results in sub-optimal recommendation performance. As can be seen in Figure 1, Alice and Bob have *multiple* historical items *iPhone* which are high-related to the next items *iPhone*. To address this issue, inspired by [22] which recalibrates channel-wise feature by explicitly modelling inter-dependencies between channels, we propose Squeeze-Excitation Attention. To adapt to the recommendation scenario, we extend it to propose an *item-wise* statistics. Formally, the attention scores can be yielded by

$$\text{SEAtt}(\text{Mod}(\mathcal{H}_{(u)}^{l-1})) = \mathcal{F}_{ex}(\mathcal{Z}^{(u)}), \tag{12}$$

$$\mathcal{F}_{ex}(\mathcal{Z}^{(u)}) = \sigma(g(\mathcal{Z}^{(u)})) = \sigma(\mathcal{W}_2\delta(\mathcal{W}_1\mathcal{Z}^{(u)})), \tag{13}$$

$$\mathcal{Z}^{(u)} = \mathcal{F}_{sq}(\text{Mod}(\mathcal{H}_{(u)}^{l-1})) = \frac{1}{d_{model}}\sum_{j=1}^{d_{model}} \text{Mod}(\mathcal{H}_{(u)}^{l-1})_{[:,j]}. \tag{14}$$

where $\text{SEAtt}(\text{Mod}(\mathcal{H}_{(u)}^{l-1})) \in \mathbb{R}^{n\times 1}$ denotes the attention scores. $\mathcal{Z}^{(u)} \in \mathbb{R}^{n\times 1}$ denotes *item-wise* statistics produced by average pooling of $\text{Mod}(\mathcal{H}_{(u)}^{l-1}) \in \mathbb{R}^{n\times d}$ which denotes the input of SEAtt in $\mathcal{G}_{\text{Trm}}$ or $\mathcal{L}_{\text{Conv}}$, *i.e.*, $\text{MHSA}(\mathcal{E}_{global}^{en}(\mathcal{H}_{(u)}^{l-1}))$ or $\text{Conv}_{mid}^{k=3}(\mathcal{E}_{local}^{en}(\mathcal{H}_{(u)}^{l-1}))$. $\mathcal{W}_1 \in \mathbb{R}^{\frac{n}{r}\times n}$ and $\mathcal{W}_2 \in \mathbb{R}^{n\times\frac{n}{r}}$ denote squeeze and excitation parameters, respectively. The reduction ratio r is a hyper-parameter with a default value 2 to limit model complexity and benefit generalization. δ and σ refer to the *ReLU* and *Sigmoid* activation function, respectively.

Discussion Squeeze-Excitation Attention (SEAtt) differs from the widely used attention [9, 25, 27] in two aspects: **a)** adopting effective *squeeze-excitation* operation to improve the representation capability [22]; **b)** employing the *non-mutually-exclusive* relations for recommendation which allows the next item to interact with multiple historical items. In contrast, the softmax activation requires the sum of all item scores to be 1, which are *mutually exclusive* relations that hinder multiple high-related items to be activated simultaneously.

3.4.4 Adaptive Mixture Units. This module is shown in the top part of the AdaMCT in the middle of Figure 2. The adaptive mixture units adjust the personalized trade-off between the global attention module $\mathcal{G}_{\text{Trm}}$ and the local convolutional module $\mathcal{L}_{\text{Conv}}$. Specifically, it first leverages the average pooling operator to produce dimension-wise statistics, and then a linear layer with Ω activation function is employed to get the adaptive coefficient for each user.

$$\text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1}) = \Omega(\mathcal{L}inear_{mid}(\mathcal{P}ool(\mathcal{H}_{(u)}^{l-1}))), \tag{15}$$

$$\mathcal{P}ool(\text{Out}(\mathcal{H}_{(u)}^{l-1}))) = \frac{1}{n}\sum_{i=1}^{n}\text{Out}(\mathcal{H}_{(u)}^{l-1})_i, \tag{16}$$

where $\mathcal{P}ool(\mathcal{H}_{(u)}^{l-1})) \in \mathbb{R}^{d_{model}}$ denotes user preference representation at $(l-1)$-th layer. $\Omega(\cdot)$ denote the *Sigmoid* activation function which is used to scale the value to the range 0 to 1. $\mathcal{H}_{(u)}^{l-1}$ represents the output of $(l-1)$-th layer. $\text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1}) \in \mathbb{R}$ is a personalized coefficient changing over different users since user preference representation is formulated by the average pooling of historical user behaviors. Thus, if we have $|\mathcal{U}|$ users, it will produce $|\mathcal{U}|$ coefficient values. Besides, the adaptive coefficient values are unique per the AdaMCT layer, which further extends the capacity of representation learning. In all, adaptive mixture units are module- and layer-aware with personalized inductive bias.

Finally, we mix the output of global attention module ($\mathcal{G}_{\text{Trm}}$) and local convolutional module ($\mathcal{L}_{\text{Conv}}$) with the adaptive coefficient ($\text{Ada}_{(u)}$) to produce local-global dependencies representation.

$$\mathcal{M}_{\text{Trm}}^{\text{Conv}} = \text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1}) \otimes \mathcal{L}_{\text{Conv}} + (1 - \text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1})) \otimes \mathcal{G}_{\text{Trm}} \tag{17}$$

where $\text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1})$ and $1 - \text{Ada}^{(u)}(\mathcal{H}_{(u)}^{l-1})$ are denoted by α and β respectively in Figure 2. Furthermore, to maintain the dimension invariant, we apply a linear layer to adjust the latent dimension of $\mathcal{M}_{\text{Trm}}^{\text{Conv}}$ and leverage a residual connection [14] around the input to make the $(l-1)$-th layer's output $\mathcal{H}_{(u)}^{l} \in \mathbb{R}^{n\times d}$ followed by layer normalization.

$$\mathcal{H}_{(u)}^{l} = \text{LN}(\mathcal{H}_{(u)}^{l-1} + \text{Dpt}(\mathcal{L}inear_{out}(\mathcal{M}_{\text{Trm}}^{\text{Conv}}))). \tag{18}$$

3.5 Output Layer

After L layers hierarchically aggregate information across all positions, we get the final output $\mathcal{H}_{(u)}^{L} \in \mathbb{R}^{n\times d_{model}}$ for all items of the input sequence. Then, we use the contextual hidden representation at the n-th time step of $\mathcal{H}_{(u)}^{L}[n]$ as the input of the prediction layer, which consists of a two-layer feed-forward network and a softmax function to produce the *full-ranked* items probability distribution over the next items.

$$\mathcal{P}_{out}(\hat{v}_{n+1}^{(u)}) = \sigma(\delta(\mathcal{H}_{(u)}^{L}[n]\mathcal{W}^{Pred} + b^{Pred})E_{table}^{\mathcal{T}} + b^{O}), \tag{19}$$

where $\mathcal{W}^{Pred} \in \mathbb{R}^{d_{model}\times d_{model}}$, $b^{Pred} \in \mathbb{R}^{d_{model}}$, and $b^{O} \in \mathbb{R}^{|\mathcal{V}|}$ are parameters and $E_{table}^{T} \in \mathbb{R}^{d\times|\mathcal{V}|}$ is an item embedding table. We follow [28, 45] to use a shared item embedding table of the embedding module to avoid overfitting and speed up model convergence. δ and σ denote the *GELU* and $softmax$ function, respectively.

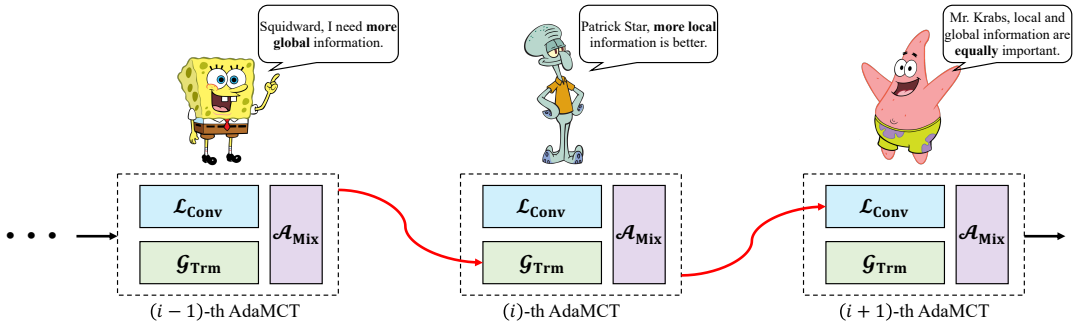


Figure 3: A toy example to show how the model self-adaptively coordinates the mixture of local (CNNs) and global (Transformer) information between different layers. The cartoon characters are from an American animated television series SpongeBob SquarePants.

3.5.1 Model Training. In this work, we adopt cross-entropy loss to supervised model training. Formally, the objective loss is formulated as follows:

$$\mathcal{L}_{ce}(\theta) = -\sum_{u\in\mathcal{U}} \log \mathcal{P}_\theta(v_{n+1}^{(u)} = v|\mathcal{S}_{1:n}^{(u)}) = -\sum_{u\in\mathcal{U}} \log \mathcal{P}_{out}(\hat{v}_{n+1}^{(u)}). \tag{20}$$

where θ represents the model parameters.

4 EXPERIMENTS

In this section, we present the empirical evaluations AdaMCT on three widely used benchmarks with varied domains and statistics, aiming to answer the following research questions.

- **RQ1:** Can our proposed AdaMCT outperform the state-of-the-art baselines for sequential recommendation?
- **RQ2:** What is the influence of various component designs in AdaMCT?
- **RQ3:** How do different hyper-parameters affect AdaMCT?
- **RQ4:** How efficient is AdaMCT compared to the baselines? How about the model complexity of AdaMCT?
- **RQ5:** Does the visualization of attention matrices of Transformer provide some interpretability for feature representations in the AdaMCT framework?

4.1 Experimental Settings

4.1.1 Dataset. In this work, our experiments are conducted on three real-world and widely used benchmarks.

- **Amazon Toys, Sports, Beauty**[1]**:** This is a series of datasets [41] that consist of large corpora of product reviews crawled from Amazon.com. The datasets are split by the top-level product categories. We adopt the "*Toys and Games*", "*Sports and Outdoors*", and "*Beauty*" categories.

The selected datasets are the ones used in important prior works [5, 15, 28, 30, 45], from which we followed the same data preprocessing procedures. In essence, we convert each dataset into an implicit dataset by treating each rating or review as a presence of a user-item interaction. After that, we group the interactions by user IDs and sort them by timestamps to form a sequence for each user.

[1] http://snap.stanford.edu/data/amazon/

Table 1: Dataset statistics after preprocessing. '#Inter.' denotes the number of interaction. '#Len$_{\mathcal{U}}$' and '#Len$_{\mathcal{I}}$' denotes average actions of Users/Items.

Datasets	#Inter.	#Users	#Items	#Len$_{\mathcal{U}}$	#Len$_{\mathcal{I}}$	Sparsity
Toys	167,597	19,412	11,924	8.6	14.1	99.93%
Sports	296,337	35,598	18,357	8.3	16.1	99.95%
Beauty	198,502	22,363	12,101	8.9	16.4	99.93%

We follow the same practice as in [5, 17, 28, 43, 45, 48] to discard short sequences less than five interactions. Table 1 describes the statistics of the datasets. These three datasets are frequently used as public benchmarks and differ in size and density, which gives us the opportunity to test models on different kinds of datasets.

4.1.2 Evaluation Metrics. To evaluate the performance of each method, we employ the widely used *leave-one-out* evaluation task for each user's item sequence, which means we hold out the last item for the test, the second last item for validation, and the rest for training. Besides, we follow the common practice in [5, 28, 30, 45] to pair each ground truth item in the test set with 100 randomly sampled *negative* items that have not been interacted by the user. To make the task more realistic, we sample the negative items by uniform samplings. To score the ranked list, we employ two common evaluation metrics: *Hit Ratio* (HR), and *Normalized Discounted Cumulative Gain* (NDCG). Since we only have one ground truth item for each user, HR@K is equivalent to Recall@K. Therefore, both Recall@K and NDCG@K have large values when the ground truth item is ranked higher up in the top-K list. We report Recall and NDCG with $k = 1, 5, 10$.

4.1.3 Baseline Models. We choose some typical state-of-the-art SR methods for comparison, especially including individual CNNs (Caser, NextItNet) and Transformer (SASRec, BERT4Rec, GCSAN, and CORE). The baselines are introduced as follows.

- **GRU4Rec** [19]: A session-based recommender system based on recurrent neural network.
- **Caser** [47]: A CNN-based model with horizontal and vertical convolutional layers for personalized recommendation.
- **NextItNet** [61]: A simple convolutional generative network for next item recommendation.

Table 2: Comparison of recommendation performance with baseline models in numerical values. Bold scores are the best in each row, while underlined scores are the second best. * and † denote CNN-based and Transformer-based models, respectively.

Dataset	Metric	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)	(j)	(k)
		GRU4Rec	Caser*	NextItNet*	SASRec†	BERT4Rec†	HGN	GCSAN†	SINE	CORE†	AdaMTC	Improv.
Toys	Recall@1	0.1650	0.1264	0.1092	0.2120	0.1298	0.1679	0.1840	0.1745	0.1785	**0.2184**	+3.02%
	Recall@5	0.3535	0.3144	0.2893	0.3956	0.3040	0.3609	0.3530	0.3678	0.3812	**0.4132**	+4.45%
	Recall@10	0.4591	0.4231	0.3952	0.4886	0.4097	0.4683	0.4469	0.4665	0.4860	**0.5088**	+4.13%
	NDCG@5	0.2630	0.2237	0.2019	0.3080	0.2195	0.2679	0.2718	0.2748	0.2842	**0.3203**	+3.99%
	NDCG@10	0.2971	0.2588	0.2361	0.3380	0.2535	0.3024	0.3021	0.3066	0.3180	**0.3511**	+3.88%
Beauty	Recall@1	0.1848	0.1503	0.1580	0.2154	0.1457	0.1835	0.1832	0.1903	0.1826	**0.2178**	+1.11%
	Recall@5	0.3688	0.3364	0.3560	0.4014	0.3176	0.3793	0.3629	0.3868	0.3949	**0.4207**	+4.81%
	Recall@10	0.4687	0.4370	0.4605	0.4933	0.4163	0.4792	0.4554	0.4806	0.4951	**0.5178**	+4.58%
	NDCD@5	0.2806	0.2469	0.2611	0.3127	0.2348	0.2859	0.2771	0.2937	0.2938	**0.3241**	+3.65%
	NDCD@10	0.3129	0.2793	0.2949	0.3424	0.2666	0.3181	0.3069	0.3239	0.3262	**0.3554**	+3.80%
Sports	Recall@1	0.1423	0.1239	0.1255	0.1716	0.1178	0.1593	0.1369	0.1641	0.1453	**0.1862**	+8.51%
	Recall@5	0.3386	0.3173	0.3275	0.3760	0.3036	0.3657	0.3303	0.3751	0.3713	**0.4097**	+8.96%
	Recall@10	0.4606	0.4417	0.4578	0.4990	0.4269	0.4890	0.4457	0.4935	0.4933	**0.5412**	+8.46%
	NDCG@5	0.2434	0.2226	0.2287	0.2775	0.2128	0.2660	0.2365	0.2728	0.2616	**0.3019**	+8.79%
	NDCG@10	0.2828	0.2627	0.2708	0.3172	0.2525	0.3059	0.2737	0.3110	0.3010	**0.3444**	+8.58%

- **SASRec** [28]: A Transformer-based model that uses self-attention network for the sequential recommendation.
- **BERT4Rec** [45]: A bidirectional Transformer-based model for the sequential recommendation.
- **HGN** [39]: A hierarchical gating networks for sequential recommendation.
- **GCSAN** [54]: A graph contextualized self-atttention network for session-based recommendation.
- **SINE** [46]: A sparse-interest network for sequential recommendation.
- **CORE** [20]: A simple and effective session-based recommendation with unifying representation space.

4.1.4 Implementation Details. We evaluate baseline models based on the famous open-source RecBole framework[2]. We apply grid search to find the optimal hyper-parameters for each model. We consider hidden dimension size $d_{model} \in \{16, 32, 64, 96, 128\}$, number of heads $h \in \{2, 4, 8, 16, 32\}$, reduction ratio $r \in \{1, 2, 5, 10, 25\}$, kernel size $k \in \{3, 5, 7, 9, 11\}$, hidden activation $\Phi \in \{`GELU', `ReLU', `Swish', `TanH', `Sigmoid'\}$, hidden dropout ratio $d_1 \in [1e-1, 1]$, attention dropout ratio $d_2 \in [1e-1, 1]$ and learning rate $lr \in [1e-8, 1]$. We report the results of each baseline under its optimal hyper-parameter settings. For our AdaMCT, the optimal hyper-parameters for each benchmark are **Toys** : $\{L = 2, k = 3, r = 2, \Phi = `GELU', h = 4, d_{model} = 32, d_1 = 0.50, d_2 = 0.50, lr = 0.001\}$, **Beauty** : $\{L = 2, k = 5, r = 2, \Phi = `Swish', h = 2, d_{model} = 16, d_1 = 0.43, d_2 = 0.57, lr = 0.02\}$, and **Sports** : $\{L = 2, k = 3, r = 5, \Phi = `GELU', h = 4, d_{model} = 64, d_1 = 0.76, d_2 = 0.51, lr = 0.001\}$, respectively. We train the model using the Adam algorithm with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and l_2 weight decay of 0.0, and linear decay of the learning rate. To ensure fairness, we used the same maximum sequence length ($N = 50$ for Amazon) and the number of layers ($L = 2$) as in [28]. We train each model with early stop strategies until the validation accuracy does not improve for 20 epochs on a single NVIDIA A100 SXM4 80GB GPU with a batch size of 2048. The results of the test sets are reported in the following section.

[2]https://github.com/RUCAIBox/RecBole

4.2 Overall Performance (RQ1)

Table 2 shows the recommendation performance of our model and baselines on the three datasets. Here, we use the averaged performance run by five random seeds. Consequently, the improvement provided by AdaMCT is statistically and consistently significant over all datasets.

We highlight two important baselines Caser and SASRec which belong to the individual architecture of CNNs and Transformer respectively. AdaMCT surpasses Caser by a margin exceeding 4% on the Toys and Beauty datasets, and even over 8% on the Amazon Sports dataset, considering all metrics on average. Caser performs fairly well compared to GRU4Rec, implying that models containing local inductive bias are nontrivial in predicting the next item. Self-attention based models such as SASRec and BERT4Rec show better performances over the RNN-based GRU4Rec and CNN-based Caser. This suggests that global attentive representation learning is relatively more important than the use of only one inductive bias. Meanwhile, GNN-based models, *i.e.,* GCSAN outperforms RNN-based and CNN-based models, underscoring the importance of capturing pairwise relationships between items. Nevertheless, these models fall short of surpassing SASRec, indicating that sequential relationships prevail when modeling user preferences in sequential recommendation.

On average, AdaMCT outperforms SASRec across all three datasets. As such, AdaMCT emerges as the most effective model, adeptly considering both *local* and *global* dependencies of user interest patterns in an adaptive manner.

If we extend the comparison to ablated AdaMCT, we find that AdaMCT performs better than SASRec while allowing either the local module or the global module to be adaptive, as shown in Table 3. This means that with our proposed adaptive mixture units, only CNN-based components can already be comparable with SASRec. On the contrary, when we fix the global and local trade-off

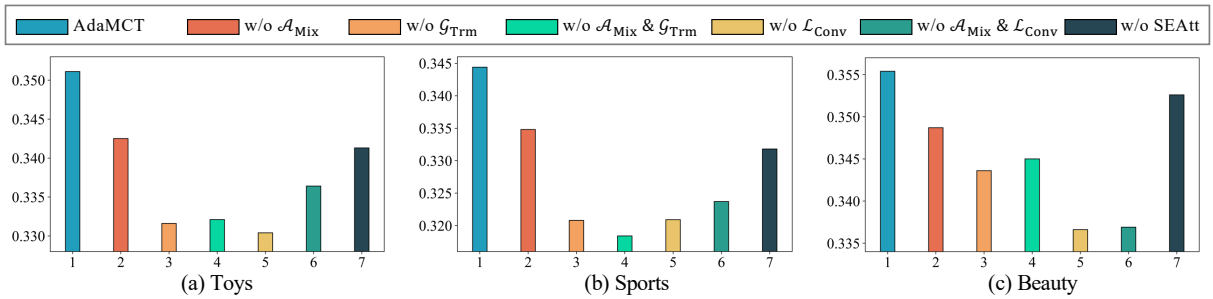


Figure 4: Performance (NDCG@10) comparison of with(w) or without(w/o) proposed components including a global attention module (Transformer) $\mathcal{G}_{\text{Trm}}$, a local convolutional module (CNNs) $\mathcal{L}_{\text{Conv}}$, an adaptive mixture unit $\mathcal{A}_{\text{Mix}}$, and two squeeze-excitation attention (SEAtt) per layer of AdaMCT on three datasets.

Table 3: Performance (NDCG@10) comparison of the adaptive or fixed mixture of local (CNNs) $\mathcal{L}_{\text{Conv}}$ and global (Transformer) $\mathcal{G}_{\text{Trm}}$ on three datasets. The best performance in each column is boldfaced.

Setting	$\mathcal{L}_{\text{Conv}}$	$\mathcal{G}_{\text{Trm}}$	Toys	Beauty	Sports
Adaptive	✗	✗	0.3425	0.3487	0.3348
	✓	✗	0.3429	0.3466	0.3341
	✗	✓	0.3404	0.3494	0.3297
	✓	✓	**0.3511**	**0.3554**	**0.3444**
Fixed	×1.0	×0.0	0.3322	0.3450	0.3184
	×0.8	×0.2	0.3403	0.3484	0.3330
	×0.5	×0.5	0.3412	0.3442	0.3314
	×0.2	×0.8	0.345	0.3477	0.3334
	×0.0	×1.0	0.3364	0.3369	0.3237

to a constant value without user-specific learnable adaptability, the recommendation performance drops drastically. The best fixed AdaMCT using 80% global and 20% local modules without personalized adaptability is only slightly better than BERT4Rec but worse than the self-adaptive AdaMCT. Table 3 also shows that the proposed AdaMCT that adaptively exploits the mixing importance of CNNs and Transformer layer by layer can improve the overall representation learning effectively.

4.3 Ablation Study (RQ2)

To understand the importance of different modules in our model, we conduct a detailed ablation study. We analyze the importance of self-adaptive local (CNNs) and global (Transformer) awareness in Table 3, analyze the functions of various computational components in Figure 4, and analyze the effectiveness of squeeze-excitation attention (SEAtt) in Table 4.

As can be seen in Figure 4, when we remove any of the proposed components of AdaMCT, the performance drops by a significant margin, which demonstrates the effectiveness of each designed component. Specifically, the ablation results reveal that removing the global attention module (Transformer) $\mathcal{G}_{\text{Trm}}$ and local convolutional module (CNNs) $\mathcal{L}_{\text{Conv}}$ will significantly drop the model performance across all three datasets. It confirms our intuition that

Table 4: Performances (NDCG@10) comparison of Squeeze-Excitation Attention (SEAtt) with local (CNNs) $\mathcal{L}_{\text{Conv}}$ and global (Transformer) $\mathcal{G}_{\text{Trm}}$ awareness on three datasets. '♣' and '♢' indicates adopting *Sigmoid* and *Softmax* activation function, respectively. The best performance in each column is boldfaced.

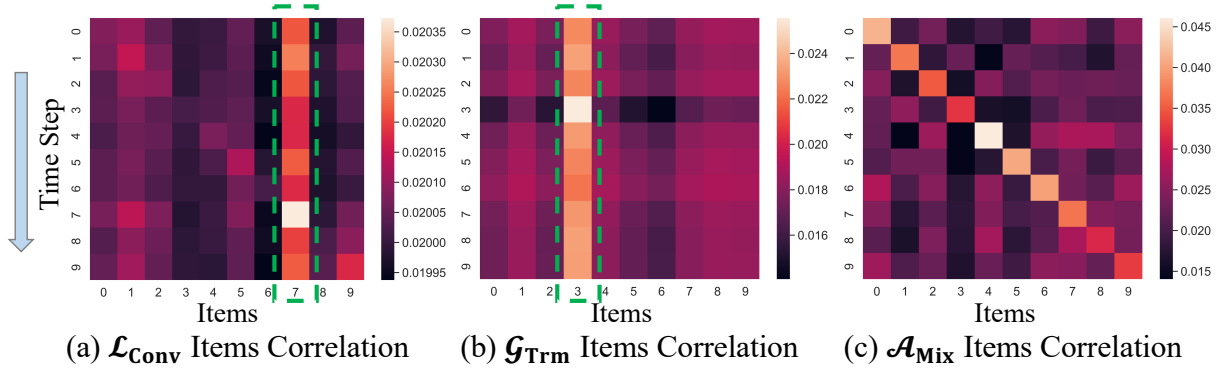
Setting	$\mathcal{L}_{\text{Conv}}$	$\mathcal{G}_{\text{Trm}}$	Toys	Beauty	Sports
(1)	♢	♢	0.3343	0.3447	0.3226
(2)	♣	♢	0.3357	0.3433	0.3316
(3)	♢	♣	0.3355	0.3525	0.3255
(4)	♣	♣	**0.3511**	**0.3554**	**0.3444**

local and global features are important for sequential recommendation tasks. Moreover, adaptive mixture unit $\mathcal{A}_{\text{Mix}}$ plays a vital role in mixing the local (CNNs) and global (Transformer) features. Removing two squeeze-excitation attention (SEAtt) also leads to performance decline. However, different datasets have shown different sensitivity patterns. It seems that the simpler Beauty dataset is easier to be learned the importance of each item compared to the other two datasets.

We discuss the effectiveness of the squeeze-excitation attention (Sigmoid) in Table 4. The proposed SEAtt can improve the original Softmax approach by approximately 5.03%, 2.73%, and 6.29% over three datasets respectively, in terms of NDCG@10. Besides, we observe that Softmax is more beneficial to the local branches (CNNs), and Sigmoid is more favorable for the global branches (Transformer). In addition, the performance of Sigmoid on the local branch still outperforms the performance of Softmax on the global branch, and the hybrid of Softmax and Sigmoid is better than simply adopting the Softmax on both branches except the Beauty dataset. These demonstrate that considering multiple relevant items of equal importance simultaneously on user historical behavior is conducive to capturing user interest preference accurately.

4.4 Hyperparameter Study (RQ3)

In this section, we conduct a stability study to discuss the effect of a variety of hyperparameters on model performance in Table 5. We also visualize the learning procedure of local-global coefficients to



(a) $\mathcal{L}_{\text{Conv}}$ Items Correlation (b) $\mathcal{G}_{\text{Trm}}$ Items Correlation (c) $\mathcal{A}_{\text{Mix}}$ Items Correlation

Figure 5: Visualizations of item correlation patterns in the proposed $\mathcal{L}_{\text{Conv}}$, $\mathcal{G}_{\text{Trm}}$, and $\mathcal{A}_{\text{Mix}}$ components of AdaMCT on Beauty dataset. It is calculated by the feature representation matrics of items of each component in the last layer multiplying its transposed matrics followed by $Softmax$ normalization.

Table 5: Comprehensive hyperparameter study of kernel size (k), reduction ratio (r), hidden activation (Φ), head number (h), and hidden dimension (d_{model}) on Beauty datasets (more datasets results can be found in Appendix). The NDCG@10 performance is reported. The best performance is boldfaced.

k	3	5	7	9	11
	0.3489	0.3521	0.3541	0.3495	0.3528
r	1	2	5	10	25
	0.3487	0.3526	0.3493	0.3452	0.3520
Φ	*GELU*	*ReLU*	*Swish*	*TanH*	*Sigmoid*
	0.3461	0.3541	0.3526	0.3491	0.3496
h	2	4	8	16	32
	0.3526	0.3470	0.3525	0.3496	0.3485
d_{model}	16	32	64	96	128
	0.3554	0.3526	0.3309	0.3216	0.3224

get the gist of how AdaMCT adaptively mixes the feature of CNNs and Transformers, as can be seen in Figure 6.

Specifically, we analyze the effect of the kernel size (k), reduction ratio (r), hidden activation (Φ), head number (h), and hidden dimension (d_{model}) in Table 5. It can be seen that on all three datasets, the performance of AdaMCT improves by a narrow margin with changes in dimensionality. This shows that our model is generally insensitive to dimensionality. In particular, the smaller size of the hidden dimension (d_{model}) shows a better performance, indicating that for simpler datasets (*e.g.*, Beauty) the hidden dimensionality does not need to be too large. From the results, we can tell that AdaMCT is overall stable.

One of the novelties in the proposed AdaMCT model lies in the fusion of local (CNNs) and global (Transformer) information through a self-adaptive approach. To validate our intuition, we visualize the learning procedure of coefficients of the local and global branches for two randomly chosen users (denoted as user1 (P_1) and user2 (P_2)), two layers (denoted as L_1 and L_2) and the α and β of the last layer with average scores, as shown in Figure 6. We observe that AdaMCT updates the coefficients through learning

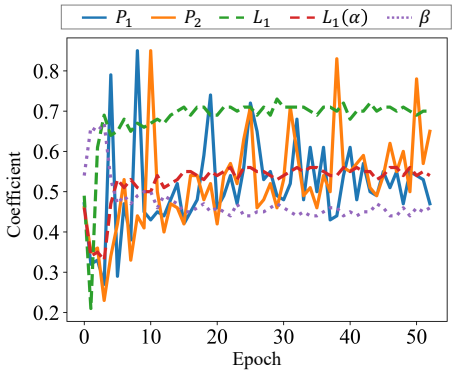


Figure 6: Visualization of learning procedure of adaptive coefficient for two users (P_1 and P_2), two layers (L_1 and L_2), and the α and β in the last layer (L_2) on Toys dataset during training iterations.

Table 6: Parameters number and execution efficiency analysis of models on Beauty dataset.

Beauty	AdaMCT	Caser	SASRec	CORE
# Parameters	0.21M	2.91M	0.88M	0.88M
# GPU Memory	0.79GB	0.62GB	2.31GB	2.33GB
# FLOPs ($\times 10^6$)	0.34	11.38	4.98	4.99
Training Cost (per epoch)	4.75s	34.02s	3.65s	3.97s
Inference Cost (per epoch)	26.95s	157.07s	22.48s	22.57s
NDCG@10	**0.3554**	0.2793	0.3424	0.3262

iterations. The fluctuating curves indicate that the local and global information serve distinct functions at varying training phases: the CNNs components are given higher weights at the beginning and then AdaMCT gradually learns to assign weights to the Transformer module. This suggests that capturing local dependency may hold greater significance than the Transformer during the initial stages, and the convolutional components could potentially furnish valuable information to guide the optimization of the Transformer,

thereby enhancing performance. Furthermore, the coefficients exhibit variation across different users, signifying that the model self-adaptively learns distinct patterns for individualized users. This further illustrates that it is not an ideal way to set a static and equal trade-off between local and global branches, as many works [4, 18, 21, 32, 55, 56, 58] did in the past. In our adaptive paradigm, the model can fuse information with the dynamical use of locality bias (CNNs) to improve the performance of the Transformer.

4.5 Complexity and Efficiency Analysis(RQ4)

In order to understand whether the performance improvement is attributable to the model architecture design or the escalation in the quantity of parameters, we conduct an analysis on AdaMCT's complexity and execution efficacy, as delineated in Table 6. Comparing with CNN-based and Transformer-based baselines, we have the following observations: (1) AdaMCT has a relatively fewer parameters. This is reasonable because our Transformer removes Feed-forward Network (FFN) and becomes more lightweight. (2) AdaMCT has better efficiency in both training (86% less time) and inference (83% less time) compared with Caser. Meanwhile, AdaMCT achieves comparable training/inference time and much smaller memory footprint (66% less) and FLOPs (93% less) with SASRec and CORE. Moreover, our model has the minimum FLOPs compared with all baselines which further enhances the feasibility of practical deployment, especially for low-resource constraints devices. The results indicate that AdaMCT is capable of achieving a good balance between computational cost and recommendation accuracy.

4.6 Visualization (RQ5)

To provide insights into the AdaMCT's interpretability, we visualize the item correlation matrices based on feature representation of $\mathcal{L}_{\text{Conv}}$, $\mathcal{G}_{\text{Trm}}$, and $\mathcal{A}_{\text{Mix}}$ output in the last layer of AdaMCT, using test samples from the Beauty dataset, as shown in Figure 5. Based on our observations, we find that different components have different item correlation patterns. $\mathcal{L}_{\text{Conv}}$ tends to learn local item dependencies which concentrate more on recent items (red-orange area with higher correlation scores), while $\mathcal{G}_{\text{Trm}}$ prefers to learn items dependencies globally, *e.g.*, the 3rd item has the higher correlation with other items, as can be seen in Figure 5(a) and (b) green dash rectangular box, respectively.

5 CONCLUSION

In this paper, we design a novel Adaptive Mixture of CNN-Transformer (AdaMCT) that injects locality inductive bias into Transformer by combining its global attention mechanism with a local convolutional filter, and adaptively determines the mixing importance on a personalized basis through a module and layer-aware adaptive mixture units. Moreover, we propose Squeeze-Excitation Attention (SEAtt) to replace the mutually exclusive activation to improve the learning of sequence dependencies with multiple relevant items. Extensive experiments on three public real-world datasets demonstrate the effectiveness and efficiency of our proposal. For future studies, we will explore how to design more powerful adaptive mixture units in-depth and incorporate item-related (*e.g.*, price, date of production, producer) or behavior-related (*e.g.*, purchase, rate) side information into AdaMCT.

6 ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China (No. 62276099). We appreciate the NVIDIA A100 SXM4 80GB GPUs support of Upstage.

A APPENDICES

A.1 Hyperparameter Study (RQ3)

Table 7: Comprehensive hyperparameter study of kernel size (k), reduction ratio (r), hidden activation (Φ), head number (h), and hidden dimension (d_{model}) on Toys dataset. The NDCG@10 performance is reported. The best performance is boldfaced.

k	3	5	7	9	11
	0.3511	0.3440	0.3423	0.3491	0.3414
r	1	2	5	10	25
	0.3427	0.3406	0.3388	0.3392	0.3436
Φ	*GELU*	*ReLU*	*Swish*	*TanH*	*Sigmoid*
	0.3406	0.3397	0.3443	0.3425	0.3445
h	2	4	8	16	32
	0.3440	0.3406	0.3374	0.3388	0.3323
d_{model}	16	32	64	96	128
	0.3454	0.3406	0.3413	0.3391	0.3369

Table 8: Comprehensive hyperparameter study of kernel size (k), reduction ratio (r), hidden activation (Φ), head number (h), and hidden dimension (d_{model}) on Sports dataset. The NDCG@10 performance is reported. The best performance is boldfaced.

k	3	5	7	9	11
	0.3383	0.3429	0.3424	0.3403	0.3435
r	1	2	5	10	25
	0.3380	0.3388	0.3383	0.3398	0.3334
Φ	*GELU*	*ReLU*	*Swish*	*TanH*	*Sigmoid*
	0.3383	0.3349	0.3347	0.339	0.3350
h	2	4	8	16	32
	0.3383	**0.3444**	0.3392	0.3370	0.3361
d_{model}	16	32	64	96	128
	0.3081	0.3370	0.3383	0.3373	0.3381

A.2 Visualization (RQ5)

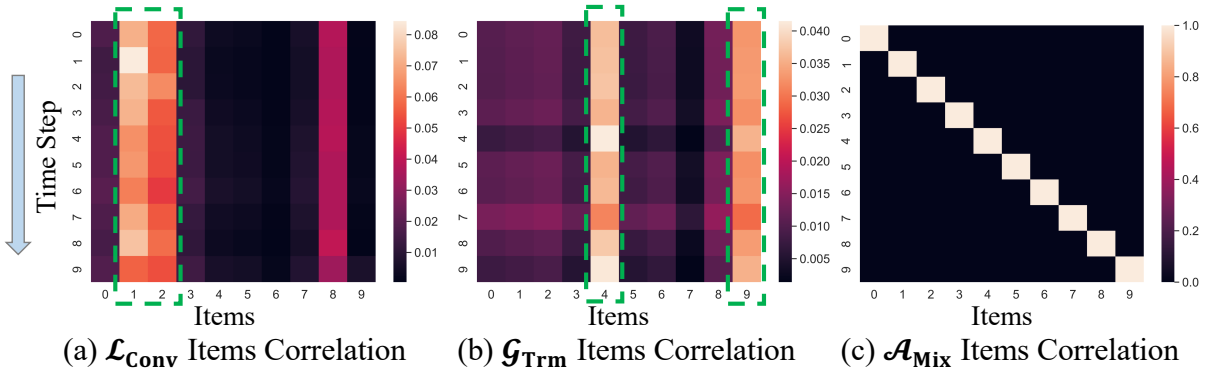


Figure 7: Visualizations of item correlation patterns in the proposed $\mathcal{L}_{\text{Conv}}$, $\mathcal{G}_{\text{Trm}}$, and $\mathcal{A}_{\text{Mix}}$ components of AdaMCT on Toys dataset. It is calculated by the feature representation matrics of items of each component in the last layer multiplying its transposed matrics followed by $Softmax$ normalization.

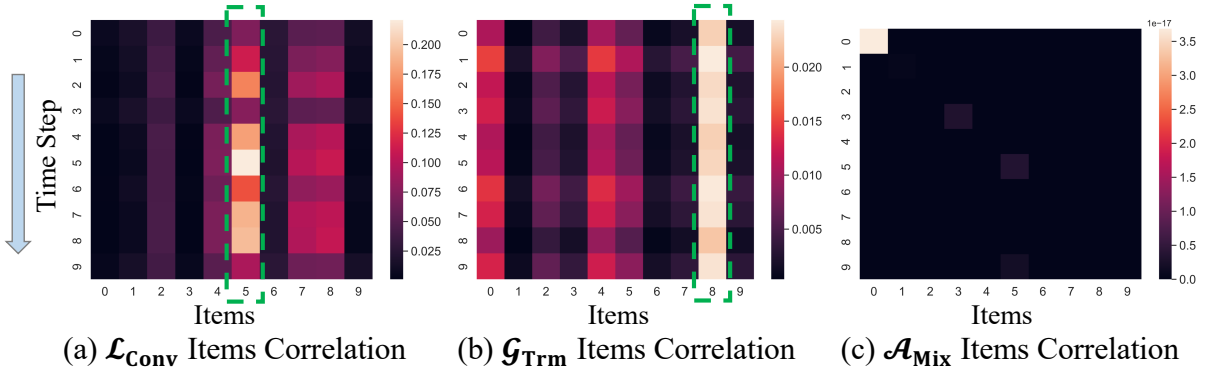


Figure 8: Visualizations of item correlation patterns in the proposed $\mathcal{L}_{\text{Conv}}$, $\mathcal{G}_{\text{Trm}}$, and $\mathcal{A}_{\text{Mix}}$ components of AdaMCT on Sports dataset. It is calculated by the feature representation matrics of items of each component in the last layer multiplying its transposed matrics followed by $Softmax$ normalization.

Table 9: Parameters number and execution efficiency analysis of models on Toys dataset.

Toys	AdaMCT	Caser	SASRec	CORE
# Parameters	0.41M	2.71M	0.87M	0.87M
# GPU Memory	1.33GB	0.62GB	2.31GB	2.33GB
# FLOPs ($\times 10^6$)	1.02	11.38	4.98	4.99
Training Cost (per epoch)	4.38s	28.33s	3.07s	3.28s
Inference Cost (per epoch)	27.91s	137.24s	19.62s	19.62s
NDCG@10	**0.3511**	0.2588	0.3380	0.3180

Table 10: Parameters number and execution efficiency analysis of models on Sports dataset.

Sports	AdaMCT	Caser	SASRec	CORE
# Parameters	1.26M	4.16M	1.28M	1.28M
# GPU Memory	1.94GB	0.66GB	2.16GB	2.31GB
# FLOPs ($\times 10^6$)	3.81	11.38	4.98	4.99
Training Cost (per epoch)	7.77s	48.96s	4.89s	5.82s
Inference Cost (per epoch)	55.18s	250.69s	32.23s	35.53s
NDCG@10	**0.3444**	0.2627	0.3172	0.3010

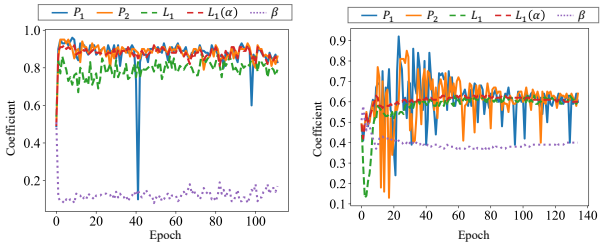


Figure 9: Visualization of learning procedure of adaptive coefficient for two users (P_1 and P_2), two layers (L_1 and L_2), and the α and β in the last layer (L_2) on Beauty (a) and Sports (b) dataset during training iterations.

REFERENCES

[1] Mingxiao An, Fangzhao Wu, Chuhan Wu, Kun Zhang, Zheng Liu, and Xing Xie. 2019. Neural news recommendation with long-and short-term user representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 336–345.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450* (2016).

[3] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[4] Tianwen Chen and Raymond Chi-Wing Wong. 2019. Session-based recommendation with local invariance. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 994–999.

[5] Sung Min Cho, Eunhyeok Park, and Sungjoo Yoo. 2020. MEANTIME: Mixture of Attention Mechanisms with Multi-temporal Embeddings for Sequential Recommendation. In *Fourteenth ACM Conference on Recommender Systems*. 515–520.

[6] Evangelia Christakopoulou and George Karypis. 2016. Local item-item models for top-n recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 67–74.

[7] Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. 2018. MV-RNN: A multi-view recurrent neural network for sequential recommendation. *IEEE Transactions on Knowledge and Data Engineering* 32, 2 (2018), 317–331.

[8] Robin Devooght and Hugues Bersini. 2017. Long and short-term recommendations with recurrent neural networks. In *Proceedings of the 25th conference on user modeling, adaptation and personalization*. 13–21.

[9] Yihe Dong, Jean-Baptiste Cordonnier, and Andreas Loukas. 2021. Attention is Not All You Need: Pure Attention Loses Rank Doubly Exponentially with Depth. *arXiv preprint arXiv:2103.03404* (2021).

[10] 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.

[11] Albert Gu, Caglar Gulcehre, Thomas Paine, Matt Hoffman, and Razvan Pascanu. 2020. Improving the gating mechanism of recurrent neural networks. In *International Conference on Machine Learning*. PMLR, 3800–3809.

[12] 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.

[13] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Yinglong Wang, Jun Ma, and Mohan Kankanhalli. 2019. Attentive long short-term preference modeling for personalized product search. *ACM Transactions on Information Systems (TOIS)* 37, 2 (2019), 1–27.

[14] 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.

[15] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*. 161–169.

[16] 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.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[18] 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.

[19] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based recommendations with recurrent neural networks. In *Proceedings of Fourth International Conference on Learning Representations (ICLR'16)*.

[20] Yupeng Hou, Binbin Hu, Zhiqiang Zhang, and Wayne Xin Zhao. 2022. Core: simple and effective session-based recommendation within consistent representation space. In *Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval*. 1796–1801.

[21] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Tianchi Yang. 2018. Local and global information fusion for top-n recommendation in heterogeneous information network. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1683–1686.

[22] Jie Hu, Li Shen, and Gang Sun. 2018. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7132–7141.

[23] Liang Hu, Longbing Cao, Shoujin Wang, Guandong Xu, Jian Cao, and Zhiping Gu. 2017. Diversifying Personalized Recommendation with User-session Context.. In *IJCAI*. 1858–1864.

[24] Xiaowen Huang, Shengsheng Qian, Quan Fang, Jitao Sang, and Changsheng Xu. 2020. Meta-path Augmented Sequential Recommendation with Contextual Co-attention Network. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 16, 2 (2020), 1–24.

[25] Mingi Ji, Weonyoung Joo, Kyungwoo Song, Yoon-Yeong Kim, and Il-Chul Moon. 2020. Sequential Recommendation with Relation-Aware Kernelized Self-Attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 4304–4311.

[26] Juyong Jiang, Yingtao Luo, Jae Boum Kim, Kai Zhang, and Sunghun Kim. 2021. Sequential Recommendation with Bidirectional Chronological Augmentation of Transformer. *arXiv preprint arXiv:2112.06460* (2021).

[27] Kyeongpil Kang, Junwoo Park, Wooyoung Kim, Hojung Choe, and Jaegul Choo. 2019. Recommender system using sequential and global preference via attention mechanism and topic modeling. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1543–1552.

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

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

[30] 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.

[31] Jing Lin, Weike Pan, and Zhong Ming. 2020. FISSA: fusing item similarity models with self-attention networks for sequential recommendation. In *Fourteenth ACM Conference on Recommender Systems*. 130–139.

[32] Huafeng Liu, Liping Jing, Jingxuan Wen, Zhicheng Wu, Xiaoyi Sun, Jiaqi Wang, Lin Xiao, and Jian Yu. 2020. Deep Global and Local Generative Model for Recommendation. In *Proceedings of The Web Conference 2020*. 551–561.

[33] Huiting Liu, Huimin Liu, Qiang Ji, Peng Zhao, and Xindong Wu. 2020. Collaborative deep recommendation with global and local item correlations. *Neurocomputing* 385 (2020), 278–291.

[34] Qiang Liu, Shu Wu, Diyi Wang, Zhaokang Li, and Liang Wang. 2016. Context-aware sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 1053–1058.

[35] Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. Predicting the next location: A recurrent model with spatial and temporal contexts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[36] Qiao Liu, Yifu Zeng, Refuoe Mokhosi, and Haibin Zhang. 2018. STAMP: short-term attention/memory priority model for session-based recommendation. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 1831–1839.

[37] Mi Luo, Fei Chen, Pengxiang Cheng, Zhenhua Dong, Xiuqiang He, Jiashi Feng, and Zhenguo Li. 2020. Metaselector: Meta-learning for recommendation with user-level adaptive model selection. In *Proceedings of The Web Conference 2020*. 2507–2513.

[38] Yingtao Luo, Qiang Liu, and Zhaocheng Liu. 2021. STAN: Spatio-Temporal Attention Network for Next Location Recommendation. In *Proceedings of The Web Conference*.

[39] 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.

[40] Chen Ma, Liheng Ma, Yingxue Zhang, Jianing Sun, Xue Liu, and Mark Coates. 2020. Memory Augmented Graph Neural Networks for Sequential Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 5045–5052.

[41] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*. 43–52.

[42] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.

[43] 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.

[44] Jun Song, Yueyang Wang, Siliang Tang, Yin Zhang, Zhigang Chen, Zhongfei Zhang, Tong Zhang, and Fei Wu. 2020. Local-Global Memory Neural Network for Medication Prediction. *IEEE transactions on neural networks and learning systems* (2020).

[45] 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.

[46] Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. 2021. Sparse-interest network for sequential recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 598–606.

[47] 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.

[48] 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.

[49] 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.

[50] Jianling Wang, Kaize Ding, Liangjie Hong, Huan Liu, and James Caverlee. 2020. Next-item Recommendation with Sequential Hypergraphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1101–1110.

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

[52] 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.

[53] Ruobing Xie, Yalong Wang, Rui Wang, Yuanfu Lu, Yuanhang Zou, Feng Xia, and Leyu Lin. 2022. Long short-term temporal meta-learning in online recommendation. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1168–1176.

[54] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation.. In *IJCAI*, Vol. 19. 3940–3946.

[55] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Jiajie Xu, Victor S Sheng S. Sheng, Zhiming Cui, Xiaofang Zhou, and Hui Xiong. 2019. Recurrent convolutional neural network for sequential recommendation. In *The World Wide Web Conference*. 3398–3404.

[56] Yong Xu, Jiahui Chen, Chao Huang, Bo Zhang, Hao Xing, Peng Dai, and Liefeng Bo. 2020. Joint Modeling of Local and Global Behavior Dynamics for Session-Based Recommendation. In *ECAI 2020*. IOS Press, 545–552.

[57] Feng Yu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2016. A dynamic recurrent model for next basket recommendation. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 729–732.

[58] Feng Yu, Yanqiao Zhu, Qiang Liu, Shu Wu, Liang Wang, and Tieniu Tan. 2020. TAGNN: Target attentive graph neural networks for session-based recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1921–1924.

[59] Lu Yu, Chuxu Zhang, Shangsong Liang, and Xiangliang Zhang. 2019. Multi-order attentive ranking model for sequential recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5709–5716.

[60] Zeping Yu, Jianxun Lian, Ahmad Mahmoody, Gongshen Liu, and Xing Xie. 2019. Adaptive User Modeling with Long and Short-Term Preferences for Personalized Recommendation.. In *IJCAI*. 4213–4219.

[61] 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.

[62] 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.

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

[64] Qinzhe Zhang, Jia Wu, Hong Yang, Weixue Lu, Guodong Long, and Chengqi Zhang. 2016. Global and local influence-based social recommendation. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*. 1917–1920.

[65] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM computing surveys (CSUR)* 52, 1 (2019), 1–38.

[66] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 5941–5948.