CARCA: Context and Attribute-Aware Next-Item Recommendation via Cross-Attention

Ahmed Rashed
ahmedrashed@ismll.uni-hildesheim.de
Information Systems and Machine Learning Lab, University of Hildesheim
Germany

Shereen Elsayed
elsayed@ismll.uni-hildesheim.de
University of Hildesheim
Germany

Lars Schmidt-Thieme
schmidt-thieme@ismll.uni-hildesheim.de
Information Systems and Machine Learning Lab, University of Hildesheim
Germany

ABSTRACT

In sparse recommender settings, users' context and item attributes play a crucial role in deciding which items to recommend next. Despite that, recent works in sequential and time-aware recommendations usually either ignore both aspects or only consider one of them, limiting their predictive performance. In this paper, we address these limitations by proposing a context and attribute-aware recommender model (CARCA) that can capture the dynamic nature of the user profiles in terms of contextual features and item attributes via dedicated multi-head self-attention blocks that extract profile-level features and predicting item scores. Also, unlike many of the current state-of-the-art sequential item recommendation approaches that use a simple dot-product between the most recent item's latent features and the target items embeddings for scoring, CARCA uses cross-attention between all profile items and the target items to predict their final scores. This cross-attention allows CARCA to harness the correlation between old and recent items in the user profile and their influence on deciding which item to recommend next. Experiments on four real-world recommender system datasets show that the proposed model significantly outperforms all state-of-the-art models in the task of item recommendation and achieving improvements of up to 53% in Normalized Discounted Cumulative Gain (NDCG) and Hit-Ratio. Results also show that CARCA outperformed several state-of-the-art dedicated image-based recommender systems by merely utilizing image attributes extracted from a pre-trained ResNet50 in a black-box fashion.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; **Learning from implicit feedback**; • **Information systems** → **Recommender systems**.

KEYWORDS

Sequential Recommendation, Context-Aware Recommendation, Attribute-Aware Recommendation, Cross Multi-Head Attention

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Nowadays, sequential recommendation systems play an essential role in many online platforms, including but not limited to online shops, online media providers, and social networks. In such platforms, users usually exhibit strong dynamic behavior heavily influenced by ever-changing users' contexts and content information. Due to such dynamic profiles, recent time-aware sequential recommendation approaches that capture sequential patterns in the user profiles [12, 23, 27, 28, 36] have shown superior performance when compared against traditional non-sequential techniques [19, 20] and context-aware models [18]. However, despite their competitive performance, many of those approaches only utilize the sequential order of consumed items and assume equal time gaps between the interactions. Such an approach completely ignores the actual time gaps between the consumed products and the context in which the user interacted with them. These two aspects encapsulate vital information, which is crucial for determining what item to recommend next. A third aspect that usually gets left out is the item attributes, which are indispensable for any model to overcome highly sparse settings and achieve superior performance [16, 17, 32, 36]. Additionally, many recent sequential ranking models for item recommendation share a principal limitation as they rely only on the latent features of the most recent item in the user profile to predict scores for the target items. Such a scoring approach significantly downweights the older items' influence on the next items to be recommended.

To tackle these limitations and address the three aspects, we propose a flexible context and attribute-aware recommendation model (CARCA) that captures user profiles' dynamic nature and contextual changes seamlessly alongside any available item attributes. CARCA utilizes several multi-head attention blocks to capture the evolving patterns in the user profile and utilizes a separate dedicated cross-attention block to capture the influence of all previous historical interactions on the target items to be recommended.

The contributions of this paper can be summarized as follows:

- We introduce a versatile context and attribute-aware model for item recommendation (CARCA), which can be applied to diverse settings and can leverage any additional item attributes.
- We evaluate the proposed model on four real-world datasets for item recommendation. Results show that the proposed CARCA model significantly outperforms all state-of-the-art models in item recommendation and achieves improvements of up to 53% in Normalized Discounted Cumulative Gain

(NDCG) and Hit-Ratio. Results also show that CARCA outperforms several state-of-the-art image-based recommender systems by merely using precomputed image features.
- We conduct a comprehensive ablation study to show the effect of the different model components on the prediction performance.

2 RELATED WORK

There has been a plethora of context and attribute-aware recommendation models that have shown consistent competitive performance on a wide variety of recommendation tasks, including, but not limited to, next-item recommendation [17, 23, 27, 32, 36], rating prediction [16, 18, 31], and click-through rate prediction [6, 30, 34, 35]. Context-aware models [6, 18, 30, 34, 35] can achieve superior performance compared with other models due to their ability to capture the high variability of users' behaviors and anticipate their next preferences. On the other hand, attribute-aware models [8, 16, 17, 31, 32, 36] utilize the additional item, and user attributes to generate rich latent representations and provide better recommendations. These additional attributes proved crucial in getting high-quality recommendations in highly sparse settings with rich item attributes such as in online fashion stores [8] and unique item recommendation settings like online auctions [17].

Context-aware models can be categorized into two main groups, models that utilize contextual features and other additional attributes in plain single vector format [6, 18, 30] and time-aware models that utilize the time and the sequential order of user's interactions. A popular example that belongs to the first group is the factorization machines (FM) [18] relying on mining interaction between latent embeddings of the various attributes and contextual features. Such a features extraction approach was later improved by adding deep neural networks [6, 9] and attention mechanisms [29] to capture higher-order interactions between the different features' latent vectors. DeepFM [6] another popular model that utilizes a separate deep neural network to extract non-linear attributes representations along with a traditional FM component. We further shed light on this model in the experiments section, as we propose to use it as a baseline. Similarly, NFM [9] uses a dedicated deep neural network component on top of the latent features extracted from an FM layer to capture higher-order interactions. Finally, CFM [30] which is one of the latest model in this family, utilized convolutional neural networks on the attributes' latent vectors outer product for better representations learning. However, recent studies [5] have shown that it performed significantly worse when compared against well tune simple baselines. Nevertheless, despite the competitiveness of such context-aware approaches in item rating prediction and click-through rate prediction tasks, they are significantly outperformed by time-aware sequential models when employed in item recommendation tasks in implicit feedback settings [12, 32, 36]. Such inferior performance is due to their inability to capture the sequential patterns in the user's historical interactions as they only treat the contextual features as static input features vectors.

The second group of **context-aware models** are the **time-aware sequential models** [12, 13, 15, 23, 25–27, 32, 36] which achieve state-of-art performances on item recommendation tasks. Even though these models do not use explicit contextual features, they can capture the evolving user behavior by mining the sequential patterns in their historical interactions. Earlier approaches such as GRU4Rec [10] relied on recurrent neural networks to mine such sequential patterns. This approach was later improved by using bidirectional transformer blocks in BERT4Rec [23]. Another recent approach proposed by Kang et al. [12] that utilizes self-attention blocks to extract the sequential patterns in past user interactions is SASRec. SASRec also utilized a dot-product between the sequential latent features of the most recent item in the user profile and the target items' embeddings for scoring. SASRec was later extended by adding personalized latent user vectors in the current state-of-the-art model SSE-PT [28], adding more robust regularization through stochastic shared embeddings in SSE-SASRec [27], adding the ability to model the time intervals between interactions in TiSASRec [13], and finally adding the ability to handle sparse categorical attributes in the state-of-the-art hybrid S^3Rec model [36]. All of those approaches, however, maintained the same limited scoring approach that was used in the original SASRec. Lastly, a similar parallel work by Wang et al. [26] proposed an occasion-aware model (OAR) that utilizes the timestamps embeddings and recurrent memory network to predict the next item to be recommended. OAR also utilized the same dot-product scoring similar to SASRec. We propose using all of the later seven models as baselines in our experimental section as their implementations were readily available.

Besides context-aware models, **attribute-aware models** have also shown very competitive performance in various settings despite their inability to capture the users' contexts and their evolving behavior. Earlier approaches such as mSDA-CF [14] utilized a dedicated denoising auto-encoder component for extracting latent item features from its attributes. This approach was later improved by using a contracted autoencoder in the AutoSVD model [31]. Also, the VPBR model [8] using pre-computed item attributes such as its latent image features has significantly outperformed many non-attribute-based approaches. This approach was later outperformed by a dedicated image-based item recommendation model [11], which utilizes two CNN networks ensembled to extract global features and local features from different regions of interest. We likewise propose using both approaches as baselines in our experiments. Another recent state-of-the-art attribute-aware model (GraphRec) was proposed by Rashed et al. [16] that utilizes graph features along with users and items attributes for rating prediction. We further shed light on this model in the experimental section as we propose using it as an additional baseline.

With motivation set forth from the literature review, we draw the following insight: plenty of works where the user context, time, and item attributes have been actively used to model the user preferences in various recommendation tasks. However, many of these works only focus on one or two of these three aspects, ignoring the correlation between complete contextual information and the crucial item attributes. Also, many of the recent sequential next-item recommendation models rely only on the sequential latent features of the most recent item for scoring, which only captures a limited view of the full user profile. Another essential feature that needs to be considered is the ability to handle not just sparse categorical attributes besides the contextual information but also any number of real-valued attributes to ensure the mode generalizability for

different settings. Our work is the first to tackle the all of these aspects and limitations simultaneously for the next-item recommendation task while having the ability to utilize any numerical item attributes.

3 PROBLEM DEFINITION

An **item recommendation problem** consists of a set $\mathcal{U} := \{1, \dots, U\}$ of **users**, a set $\mathcal{I} := \{1, \dots, I\}$ of **items** and a sequence $\mathcal{D} := ((u_1, i_1), \dots, (u_N, i_N)) \in (\mathcal{U} \times \mathcal{I})^*$ of their past **interactions** originating from an unknown distribution p on user/item pairs (with $U, I, N \in \mathbb{N}$). Sought is a model $\hat{p} : \mathcal{U} \to (\mathbb{R}_0^+)^{\mathcal{I}}$ for the unknown conditional density $p(i \mid u)$, i.e., given a loss function $\mathcal{L} : \mathcal{I} \times (\mathbb{R}_0^+)^{\mathcal{I}} \to \mathbb{R}$ with minimal expected loss

$$\mathbb{E}_{(u,i)\sim p}\, \mathcal{L}(i, \hat{p}(u))$$

One calls the problem **having item attributes**, if additionally there is a given matrix $A^{\text{IT}} \in \mathbb{R}^{I \times j}$ containing for item an attribute vector with $j \in \mathbb{N}$ attribute values each.

One calls the problem **having context**, if each *interaction* has additional attributes, i.e., $\mathcal{D} := ((u_1, i_1, c_1), \dots, (u_N, i_N, c_N)) \in (\mathcal{U} \times \mathcal{I} \times \mathbb{R}^l)^*$ (with $l \in \mathbb{N}$) is a sample from an unknown distribution on user/item/context triples and the goal is to find a model $\hat{p} : \mathcal{U} \times \mathbb{R}^l \to (\mathbb{R}_0^+)^{\mathcal{I}}$ for the unknown conditional density $p(i \mid u, c)$, i.e., given a loss function $\mathcal{L}$ with minimal expected loss

$$\mathbb{E}_{(u,i,c)\sim p}\, \mathcal{L}(i, \hat{p}(u, c))$$

The most frequently encountered context is an absolute **timestamp** at which the user interacted with an item (for example measured as a real number in Unix Time).

Many papers simplify the problem, dropping the sequential nature of the problem. Consequently, they model the data as a *set* of user/item interactions (instead of as a sequence) and also evaluate on *non-sequential splits*, i.e., take out items at random positions (instead of only at the end). While this allows simpler models, it has the significant disadvantage that these splits cannot occur in reality. We will therefore look only at the sequential problem as stated above.

Sequential approaches on the other hand usually consider all users to have profiles P_t^u that contain the sequence of their previously interacted items $P_t^u := \{i_1^P, i_2^P, ..., i_{|P_t^u|}^P\}$ along with their attributes $A_t^u \in \mathbb{R}^{|P_t^u| \times j}$ and their interactions' contextual features $C_t^u \in \mathbb{R}^{|P_t^u| \times l}$ such as timestamps. The main goal of the sequential item recommendation task will be to rank a target list of items $O_{t+1}^u := \{i_1^O, i_2^O, ..., i_{|O_{t+1}^u|}^O\}$ based on their likelihood of being interacted with by the target user u at time $t+1$ while similarly considering their attributes and contextual features existing at that time point.

4 METHODOLOGY

To capture the evolving users' behaviors existing in their profiles P_t^u, the proposed CARCA model utilizes two analogous multi-head self-attention based branches. The left branch is a series of self-attention blocks that extract the user's profile's contextual information and item features. On the other hand, the right branch consists of a multi-head cross-attention block that captures the influence of the left branch's profile-level features on the target items O_{t+1}^u while taking into consideration the target items' attributes and contextual features. The second branch is also responsible for generating the ranking score for each target item in O_{t+1}^u.

Figure 1 illustrates the architecture of the CARCA model, which will be discussed in detail in the following subsections.

4.1 Embedding Layers

The first part of the features extraction pipeline in both branches is the embedding functions that will extract the initial items latent features to be fed to the self-attention blocks. To achieve that, we utilize two separate dedicated embedding functions ϕ and ψ. The first embedding function $\phi : \mathbb{R}^I \to \mathbb{R}^d$ is used to extract the first half of the item's latent features $z_i \in \mathbb{R}^d, i \in P_t^u \cup O_{t+1}^u$ from the item's one-hot encoded vectors $x_i \in \mathbb{R}^I$. The second function $\psi : \mathbb{R}^{j+l} \to \mathbb{R}^g$ extracts the second half of the latent features $q_i \in \mathbb{R}^g$ from the item's contextual features $c_i \in \mathbb{R}^l$ and attributes $a_i \in \mathbb{R}^j$. After extracting the two partial latent feature vectors, both of them are concatenated and fed into a third embedding layer $\omega : \mathbb{R}^{g+d} \to \mathbb{R}^d$ to generate the final item's latent features $e_i \in \mathbb{R}^d$ as follows:

$$z_i = \phi(x_i) = x_i W^\phi + b^\phi, \;\; W^\phi \in \mathbb{R}^{I \times d}, \; b^\phi \in \mathbb{R}^d \tag{1}$$

$$q_i = \psi(a_i, c_i) = \text{concat}_{col}(a_i, c_i) W^\psi + b^\psi, \; W^\psi \in \mathbb{R}^{(j+l) \times g}, \; b^\psi \in \mathbb{R}^g \tag{2}$$

$$e_i = \omega(z_i, q_i) = \text{concat}_{col}(z_i, q_i) W^\omega + b^\omega, W^\omega \in \mathbb{R}^{(g+d) \times d}, b^\omega \in \mathbb{R}^d \tag{3}$$

where W^ϕ, W^ψ, W^ω are the weight matrices of the embedding functions, and b^ϕ, b^ψ, b^ω are their bias vectors. concat_{col} represents column-wise concatenation of vectors.

Finally, since both the user profile P_t^u and the target items O_{t+1}^u have the same structure and format, we utilize the same embedding pipeline on both of them while sharing the network's weights. It is worthy to note that we also tried to use a single embedding layer for all input features concatenated into one long vector. However, the performance was inferior to this setup.

4.2 Self-Attention Blocks

After extracting the embeddings of all items in the user profile $E^P := \{e_1^P, e_2^P, ..., e_{|P_t^u|}^P\}$ and the embeddings of the target items $E^O := \{e_1^O, e_2^O, ..., e_{|O_{t+1}^u|}^O\}$, we use two separate self-attention components to extract profile-level features and ranking the target items.

4.2.1 Profile-Level Self-Attention Blocks. To extract the profile-level features, we feed the item embeddings E^P into a series of multi-head self-attention blocks. We first feed the item embeddings into the first part of a block, which is a multi-head self-attention layer that utilizes the scaled dot product. [24]

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{\frac{d}{H}}}\right)\mathbf{V} \tag{4}$$

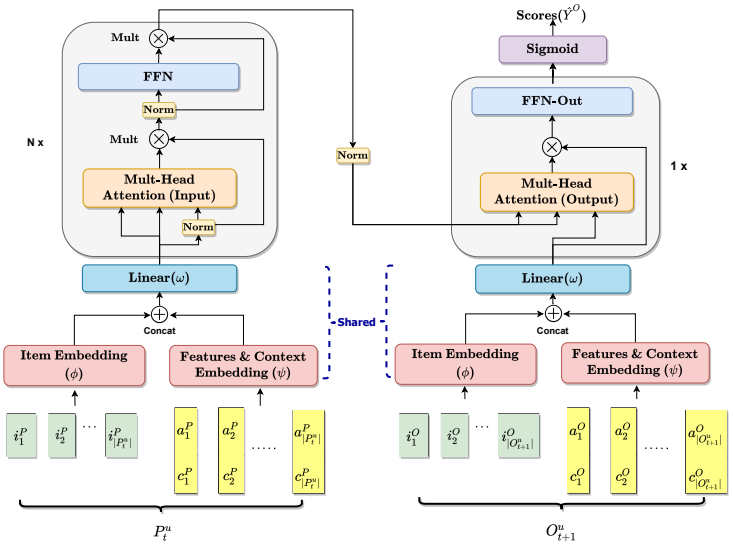


Figure 1: Illustration of the CARCA model, which is composed of two main branches, namely the profile-level features extraction branch on the left and the target items cross-attention scoring branch on the right.

$$
\begin{aligned}
S^P &= \text{SA}(\mathbf{E}^P) \\
&= \text{concat}_{col}\left(\text{Attention}(\mathbf{E}^P\mathbf{W}_h^Q, \mathbf{E}^P\mathbf{W}_h^K, \mathbf{E}^P\mathbf{W}_h^V)\right)_{h=1:H}
\end{aligned} \tag{5}
$$

where $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ represents the queries, keys and values respectively. $\mathbf{W}_h^Q, \mathbf{W}_h^K, \mathbf{W}_h^V \in \mathbb{R}^{d\times\frac{d}{H}}$ represent the linear projection matrices of the head at index h and H represents the total number of heads in the attention block. Additionally, concat_{col} concatenates vectors column-wise and $\sqrt{\frac{d}{H}}$ is a scaling factor that controls the values of the inner products.

The next step is to feed the output of the self-attention layer into the second part, which is a point-wise two-layer feed-forward network that is applied identically to all elements S_r^P of S^P with sharing parameters similar to the original transformers [24]

$$
\begin{aligned}
F^P &= \text{FFN}(S^P) \\
&= \text{concat}_{row}\left(\text{Leaky\_ReLU}(S_r^P W^{(1)} + b^{(1)})W^{(2)} + b^{(2)}\right)_{r=1:|P_t^u|}
\end{aligned} \tag{6}
$$

where $W^{(1)}$, $W^{(2)} \in \mathbb{R}^{d\times d}$ are the weight matrices of the two feed-forward layers, and $b^{(1)}$, $b^{(2)} \in \mathbb{R}^d$ are their bias vectors. Additionally, concat_{row} concatenates vectors row-wise. These feed-forward networks introduce essential non-linearity, which allows us to capture higher-order interactions between the features.

Finally, to capture more expressive profile-level features, we stack a series of those self-attention blocks where the b-th ($b > 1$) blocks are defined as follows:

$$
S^{P,(b)} = \text{SA}(\mathbf{F}^{P,(b-1)}) \tag{7}
$$

$$
F^{P,(b)} = \text{FFN}(S^{P,(b)}) \tag{8}
$$

where the 1-st block is defined as $S^{P,(1)} = S^P$ and $F^{P,(1)} = F^P$ Inspired by [12, 24], we utilized residual connections [7], layer normalization [1] and dropout [21] to alleviate the problems of overfitting and instability. One slight difference, though, is that we used multiplicative residual connections instead of additive ones as they provided better performance and CARCA's architecture is relatively not very deep compared to the ResNet models. We also omitted the causality constrain to allow bidirectional representation learning similar to BERT[4] and S^3Rec [36] models. It is also important to note that we omitted the positional encoding since this information already exists explicitly in the contextual features such as the interaction's timestamp.

4.2.2 Target-Level Cross-Attention Layer. Following the original transformers architecture and inspired by recent click-through-rating prediction models [34], and in contrast to many of the recent ranking-based sequential and context-aware item recommendation approaches [12, 25, 26, 28, 36], we utilize a dedicated multi-head cross-attention block to predict the likelihood scores of the target items. The dedicated block aims to capture the interaction between all profile-level features $F^{P,(b)}$ and the targets items instead of the widely used approach of only using the dot product between the last most recent element in $F^{P,(b)}$ and the target items initial embeddings E^O. We further shed light on the comparison between these two approaches in the experimental section.

To calculate the scores $\hat{Y}^O \in \mathbb{R}^{|O_{t+1}^u|}$ of the target items O_{t+1}^u, we feed their embeddings in a multi-head cross-attention block as the query input while using the normalized profile features $F^{P,(b)}$ from the left branch's last self-attention block as the keys and values

$$
\begin{aligned}
S^O &= \text{CA}(\mathbf{E}^O, \mathbf{F}^{P,(b)}) \\
&= \text{concat}_{col}\left(\text{Attention}(\mathbf{E}^O\mathbf{W}_h^Q, \mathbf{F}^{P,(b)}\mathbf{W}_h^K, \mathbf{F}^{P,(b)}\mathbf{W}_h^V)\right)_{h=1:H}
\end{aligned} \tag{9}
$$

$$\hat{Y}^O = \text{FFN-Out}(S^O) = \text{concat}_{row}\left(\sigma(S_r^O W^O + b^O)\right)_{r=1:|O^u_{t+1}|} \tag{10}$$

where σ is the sigmoid activation. $W^O \in \mathbb{R}^{d\times 1}$ and $b^O \in \mathbb{R}$ are the weight matrices and bias vector of the output layer.

Such a setup allows us to generate different latent representations for the target items that capture the interactions between the full profile-level features and their embeddings. It also allows us to have an arbitrarily sized list of target items as an input O^u_{t+1}. It is also worth noting that we omitted the initial multi-head self-attention block that exists on top of the original transformers architecture's output tokens because the target items in our settings are scored independently. However, this component might be useful in other scenarios where the interactions between the target items are relevant to the prediction task, such as in the next basket recommendation task.

4.3 Optimizing CARCA

To simplify the training process, we followed the same training protocol proposed by SASRec and TiSASRec [12, 13]. For each user we exclude his last interaction and we convert the user profile sequence into a fixed-length input list of items $P^u = \{i^P_1, i^P_2, ..., i^P_{|P^u_t|-1}\}$ via truncation or padding. On the other hand, the list of target items is constructed by combining a list of positive items $O^{u(+)}$ and another list of negative items $O^{u(-)}$ with equal length. The positive items list is constructed by right shifting the input list P^u to include the user's last interaction $O^{u(+)} = \{i^P_2, i^P_3, ..., i^P_{|P^u_t|}\}$ while the negative items list is generated by selecting random negative items $i \notin P^u$ and they are given the same contextual features as their corresponding positive ones. It worth noting that we also tried using the last interaction as the only positive target in $O^{u(+)} = \{i^P_{|P^u_t|}\}$, however, the performance was inferior to the list-wise setup. We further shed the light on the comparison between the two splitting approaches in Section 5.5.

Finally, we optimize the CARCA model by minimizing the binary cross-entropy loss using an ADAM optimizer, and the padded items are masked to prevent them from contributing to the loss function.

$$\mathcal{L} = -\sum_{u\in U}\sum_{r\in O^{u(+)}\cup O^{u(-)}}\left(Y^O_r \log(\hat{Y}^O_r) + (1 - Y^O_r)\log(1-\hat{Y}^O_r)\right) \tag{11}$$

Table 1: Datasets Statistics

Dataset	Users	Items	Interactions	Item Attributes
Men	34,244	110,636	254,870	2048
Fashion	45,184	166,270	358,003	2048
Games	31,013	23,715	287,107	506
Beauty	52,204	57,289	394,908	6507

5 EXPERIMENTS

In this section, we conduct multiple experiments to evaluate the performance of CARCA and to answer the following research questions.

RQ1 How well does CARCA perform compared to the state-of-the-art recommender system models on item recommendation tasks?
RQ2 How well does CARCA perform compared to the state-of-the-art image-based recommender system models while only using pre-computed item's image attributes?
RQ3 What is the impact of adding the item's attributes and contextual features?
RQ4 What are the impacts of the different components and design choices of the CARCA architecture?

5.1 Datasets

To evaluate the performance of CARCA and compare its performance against published results, we used the following four diverse and widely used real-world datasets extracted from products' reviews crawled from Amazon.com [8, 11, 12, 22, 28, 36]. All datasets' contextual features were extracted from the interactions' timestamps (i.e., day, month, year, day of week, day of year and week).

(1) **Men** [8]: This dataset contains all items that belong to men's clothing including all subcategories (gloves, scarves, sunglasses, etc.). The item attributes in this dataset are image based features extracted by using the output of the last fully connected layer of a pre-trained ResNet50 [7] on the ImageNet dataset [3].
(2) **Fashion**[8, 11]: This dataset contains six fashion categories (men/women's tops, bottoms and shoes). This dataset's item attributes are also image-based features extracted by the same pre-trained ResNet50 as the Men dataset.
(3) **Games** [12, 28]: This dataset contains all items that belong to video games category. The item attributes in this dataset are the price, fine-grained categories, and the item's brand. Most of the items' attributes are discrete and categorical in contrast to the first two datasets.
(4) **Beauty** [12, 28, 36]: This dataset contains all items that belong to beauty category. The item attributes in this dataset are the fine-grained categories and the item's brand. All of the items' attributes are also discrete and categorical.

To have a fair comparison against S^3Rec [36] that uses only categorical attributes, we discretized all real-valued input features (attributes) into different levels. The number of levels was selected as the maximum number after which the S^3Rec fails to run because of exceeding the total available memory of the GPU (1080Ti: 11 GB) or the system RAM (64GB). According to this, the maximum number of the selected equally spaced discretization levels were 10, 10, and 50 for the Men, Fashion, and Games datasets, respectively. Table 1 presents a summary of the most important statistics of the datasets.

5.2 Performance comparison with state-of-the-art item recommendation models (RQ1)

In this section, we compare the performance of the CARCA model against multiple state-of-the-art ranking-based next-item recommendation models with different capabilities.

5.2.1 Evaluation Protocol. To evaluate CARCA and other baseline models' performance, we used the widely adopted leave-one-out protocol [8, 11, 12, 26, 28, 36]. In this protocol, the two last interactions of each user are held out for validation and test while the rest of the interactions are used for training. To evaluate the model's performance, we sample 100 negative items that were not interacted with by the user, give them all the same context as their corresponding positive test item, and rank the positive test item among them. Lastly, for each user, we truncate the ranked list at a threshold value of 10. We measure the overall quality using the average Hit-Ratio (HR) and the Normalized Discounted Cumulative Gain (NDCG) across users.

To ensure the statistical significance of the reported results, we report the average metrics across five different runs on the test set, each with a different set of random negative items, and we used a paired t-test for measuring the significance. The hyper-parameters of all models were tuned on the validation set using grid search. We also tried the best hyper-parameters reported in the baselines' original papers if they were available.

5.2.2 Models.

(1) **Random**: A simple baseline model that ranks items randomly.
(2) **TopPopular**: A naive baseline model that ranks items based on their popularity.
(3) **EASE** [22]: A shallow auto-encoder based model that utilize the closed-form solution of the Frobenius norm objective function in highly sparse settings.
(4) **GraphRec** [16]: This is an extended version of the state-of-the-art attribute-aware GraphRec model for item recommendation in implicit feedback settings. We replaced the means squared error (MSE) loss function with a logistic loss function, and we trained the model by sampling positive and negative items.
(5) **DeepFM** [6]: A widely used model for click-through rate prediction that relies on learning high order feature interactions using an ensemble of Factorization Machines and deep neural networks. We modified this model for next-item recommendation by optimizing it using negative sampling and minimizing a logistic loss function.
(6) **SASRec** [12]: A state-of-the-art sequential recommendation model that utilizes self-attention blocks to predict the next item to be recommended. It also uses the dot-product between the most recent item's sequential latent features and the target item's embeddings as the scoring function.
(7) **TiSASRec** [13]: An improved version of SASRec that utilizes time-aware positional embeddings for modeling time intervals between interactions.
(8) **OAR** [26]: A state-of-the-art sequential recommendation model that utilizes recurrent memory networks and gating layers. This model also utilizes the timestamp as contextual features.
(9) **SSE-SASRec** [27]: An improved version of SASRec that utilizes stochastic shared embeddings for regularization.
(10) **BERT4Rec** [23]: A state-of-the-art model that utlizes bidirectional transformers for next item recommendations.
(11) **SSE-PT** [28]: A state-of-the-art extended version of SASRec that utilizes the latent user vectors along with their historical interactions.
(12) **S^3Rec** [36]: A state-of-the-art attribute-aware sequential recommendation model that utilizes multiple self-supervised mutual information maximization loss components for better representation learning of item attributes. This model can handle categorical attributes only, and real-valued attributes will require discretization in order to be applicable.
(13) **SASRec++(Ours)** [12]: This is our context and attribute-aware extended version of SASRec. We used the same initial feature extraction pipeline used by CARCA, and we replaced the one-hot encoded input vectors of SASRec with the extracted items' embedding vectors.
(14) **CARCA (w/o CA) (Ours)**: A version of our proposed model CARCA that does not utilize the cross-attention component and only utilize the dot-product scoring function between the sequential features of the most recent item and target items similar to other sequential recommendation models.
(15) **CARCA (Ours)**: This is our proposed method with cross-attention between all profile-level features and the target items' embeddings.

It is worth noting that we could not reproduce the original results of the SSE-PT model on the Beauty and Games datasets, although we used the authors' implementation because best hyper-parameters were neither mentioned in the paper nor the code. The authors are also no longer able to recover the best hyper-parameters as per the following GitHub issue [1] that other community members raised. To mitigate this issue, we report both our results and published results on those respective datasets.

5.2.3 Results. Results in Table 2 show that CARCA with cross attention significantly outperforms all state-of-the-art context, sequential and attribute-aware models on various settings and with different item attribute types. CARCA is also able to achieve improvements up to 53% when compared against SSE-PT as it utilizes the full pre-computed item's image features on the Men and Fashion datasets without needing any data discretization, unlike the S^3Rec that only handle categorical attributes. Additionally, CARCA was also able to significantly outperform S^3Rec in speed ($\approx$ 22 times faster) and accuracy on settings with categorical attributes such as the Games and Beauty datasets despite the competitive performance of S^3Rec on them.

Results also show that the CARCA without the cross-attention branch, which utilizes the same dot-product scoring approach adopted by many recent sequential models, can outperform all baselines, including the SASRec++ extended version on three out of the four datasets. However, this version is still outperformed by the cross-attention version, which provides a further lift of 2% to 6%

[1] https://github.com/wuliwei9278/SSE-PT/issues/1

Table 2: Performance comparison of the CARCA against state-of-the-art sequential (SEQ), context (CXT) and attribute-aware (ATT) recommendation models.

				Men		Fashion		Games		Beauty	
Model	ATT	CXT	SEQ	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10	HR@10	NDCG@10
Random				0.098	0.044	0.099	0.045	0.100	0.045	0.099	0.045
TopPop				0.415	0.269	0.407	0.262	0.519	0.314	0.451	0.261
EASE [22]				0.193	0.133	0.213	0.146	0.623	0.465	0.299	0.222
GraphRec [16]	✓			0.374	0.219	0.419	0.244	0.613	0.400	0.435	0.273
DeepFM [6]	✓	✓		0.334	0.237	0.283	0.185	0.736	0.494	0.464	0.266
SASRec [12]			✓	0.397	0.259	0.381	0.245	0.742	0.541	0.485	0.322
OAR [26]		✓	✓	0.355	0.225	0.340	0.214	0.704	0.496	0.485	0.329
TiSASRec [13]		✓	✓	0.333	0.194	0.384	0.234	0.748	0.533	0.492	0.333
BERT4Rec [23]			✓	0.315	0.193	0.328	0.209	0.705	0.509	0.478	0.318
SSE-SASRec [27]			✓	0.397	0.257	0.385	0.248	0.754	0.549	0.481	0.330
SSE-PT [28]			✓	0.397	0.258	0.381	0.246	0.748(0.775)	0.545(0.566)	0.443(0.502)	0.302(0.337)
S^3Rec [36]	✓		✓	0.365	0.238	0.367	0.239	0.765	0.549	0.538	0.371
SASRec++ (Our extension)	✓	✓	✓	0.500	0.315	0.546	0.344	0.752	0.533	0.545	0.351
CARCA (w/o CA) (Ours)	✓	✓	✓	0.521	0.322	0.568	0.359	0.738	0.517	0.556	0.358
CARCA (Ours)	✓	✓	✓	**0.550***	**0.349***	**0.591***	**0.381***	**0.782***	**0.573***	**0.579***	**0.396**
Improv. vs best published baseline (%)				38.65	35.87	53.71	53.24	2.20	4.38	7.70	6.74
Improv. vs SASRec++ (%)				10.09	10.79	8.25	10.67	3.96	7.64	6.31	12.95

(*) Significantly outperforms the best baseline at the 0.01 levels.
Published results of SSE-PT are indicated in parentheses.

in NDCG and HitRatio. This difference between the two CARCA versions shows that older items in the user profile have a significant influence on the next items to be interacted with, and such influence should not be ignored.

One interesting finding is that S^3Rec was found to be slightly inferior to SASRec on the Men and Fashion datasets. This is mainly because of the features discretization step that S^3Rec needs which leads to a negative impact on its performance due to the unavoidable information loss. On the other hand, this also explains why S^3Rec is superior on the Games and Beauty datasets, which has mostly categorical features. Another interesting finding is that BERT4Rec was also inferior to SASRec although it is a more recent approach. These findings match similar results in recently published comparative studies [36].

Finally, it is worth mentioning that the TopPopular model was found to be very competitive on the Men, Fashion, and Beauty datasets due to unevenly distributed items popularities in these datasets similar to the Epinion dataset in the famous study by [2]. We conducted a similar analysis on our test sets items, and we found out that the Gini indexes of the item's popularity in the Fashion and Men datasets (Gini indexes = 0.71 and 0.72) are significantly high, which explains the competitive performance of TopPopular model. Also, the Gini index on Beauty is slightly higher than Games (Beauty = 0.67 and Games = 0.63), which explains why the TopPopular achieved a very competitive performance on that dataset compared to its performance on the games dataset.

5.3 Performance comparison with state-of-the-art dedicated image-based item recommendation models (RQ2)

In this section, we compare the performance of the CARCA model against multiple state-of-the-art image-based item recommendation models on the Fashion dataset. Regarding the evaluation protocol, we used the same as the one proposed by SAERS [11] with sampling 500 negative items instead of 100 and using the area under the curve (AUC) score instead of the HitRatio. This allows us to compare CARCA's performance against their published results directly.

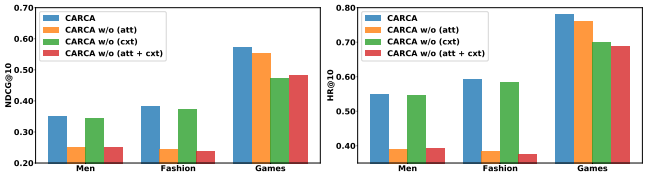
5.3.1 Baselines.

(1) **VBPR** [8]: Image-based attribute aware model that utilizes a pre-trained CaffeNet network for extracting item's image features and a BPR model for item recommendation
(2) **JRL** [33]: A State-of-the-art neural recommender system that leverages the item's image features for Top-N recommendation.
(3) **SAERS** [11]: State-of-the-art image-based item recommendation model that utilizes two CNN networks ensembled to extract global and local image features from different regions of interest.

5.3.2 Results. Results in Table 3 shows that CARCA is able to outperform SAERS and all other image-based models while merely utilizing pre-computed features and without any fine-tuned image-based deep neural network components. These findings show that even fine-tuned image-based features are not sufficient for capturing user interests, but we also need to include the contextual features of the interactions.

Table 3: Performance comparison of the CARCA against state-of-the-art image-based models on the Fashion dataset.

Model	NDCG@10	AUC
Random	0.012	0.501
TopPop	0.125	0.595
VBPR [8]	0.085	0.771
JRL [33]	0.127	0.771
SAERS [11]	0.171	0.816
CARCA	**0.184**	**0.841**
Improvement over best baseline (%)	7.54	3.03



(a) NDCG **(b) HitRatio**

Figure 2: Impact of the item's attributes and the contextual features on CARCA's performance.

5.4 Impact of item attributes and contextual features (RQ3)

To measure the impact of the item attributes and the contextual features on CARCA's performance, we conducted a comparative study between different CARCA versions that utilize different combinations of attributes and features. Figure 2 shows that the impact of item attributes and contextual features are different across the datasets. On the Men and Fashion datasets, item attributes such as their image features have a significant impact on the performance compared to the interactions contexts that had a lower impact. On the other hand, contextual features have a higher impact on CARCA's performance on the Games dataset than item attributes because video games are much more volatile than clothes and fashion-based products as they are susceptible to critics and the satisfaction of their player-bases.

5.5 Ablation Study (RQ4)

To measure the impact of each of the model components, multiple experiments on the Men dataset were conducted to compare the different possible configurations of the CARCA architecture. The hyper-parameters of each configuration were optimized separately on the validation set using grid search.

5.5.1 Configurations.

(1) **Default**: This the default configuration of CARCA that was described in Section 4.
(2) **Additive residual connections** : CARCA with additive residual connections in the multi-head attention blocks instead of multiplicative ones.
(3) **Concat. all features** : CARCA with all input vectors concatenated in one long feature vector that goes through one embedding layer.
(4) **Concat. item features** : CARCA with flipped feature extraction pipeline where item attributes are concatenated with the item's one-hot vectors instead of the contextual features.
(5) **Positional Encoding** : CARCA with positional encoding instead of contextual features.
(6) **Additional self-attention blocks on output** : CARCA with an additional multi-head attention block on top of the latent output vectors.
(7) **CARCA with single target split** ; CARCA model trained with only one target item in the target items lists $O^{u(+)} = \{i^P_{|P^u_t|}\}$ and $O^{u(-)} = \{i \notin P^u\}$.
(8) **CARCA with transformer architecture** : CARCA with a similar architecture to the original transformers, which is a combination of configurations (2), (3), (5), and (7).

5.5.2 Results. Table 4 shows that replacing the context features with traditional positional encodings has a slight negative effect on the performance, indicating that the positional encoding can be used as a substitute for contextual features of the interaction if timestamps are missing. Results also show that the multiplicative residual connection in the multi-head attention blocks provides better prediction performance as they can recover the element-wise multiplicative interactions between the latent features similar to the matrix factorization techniques. Additionally, results show that using a single target for training the CARCA model is significantly inferior to using list-wise target items because the model failed to capture the sequential correlation between the inputs and target lists during training as this suppresses its bidirectional auto-regressive capability. However, this can be solved by splitting the whole user's history into sequences with different lengths in a rolling window fashion. Moreover, in Table 4, we can see that the combination strategy of the item attributes and contextual features has different impacts on the overall performance, with the default configuration being the best. Finally, results also showed that using the transformer architecture or using additional attention blocks on top of the cross-attention block has a significant negative effect on the overall performance.

Table 4: Ablation analysis between different CARCA configurations on the Men dataset.

Configuration	HR@10	NDCG@10
Default (1)	**0.550**	**0.349**
Additive residual connections (2)	0.513	0.325
Concat. all features (3)	0.543	0.340
Concat. item features (4)	0.540	0.339
Positional encoding (5)	0.544	0.345
Additional self-attention blocks on output (6)	0.427	0.231
CARCA with single target split (7)	0.394	0.233
CARCA with transformer architecture (8)	0.459	0.276

6 CONCLUSION AND FUTURE WORK

In this paper, we propose CARCA, a context and attribute-aware model that captures user profiles' dynamic nature and contextual changes seamlessly alongside leveraging any available item attributes. CARCA also uses a cross-attention component for scoring items by capturing the correlation between old and recent items in the user profile and their influence on deciding which item to recommend next. Experimental results on four diverse datasets show that CARCA significantly outperforms multiple state-of-the-art models on the task of item recommendation.

In future works, we plan to extend CARCA for next-basket recommendations scenarios. We also plan to extend the model capacity by exploring different regularization techniques such as stochastic shared embeddings.

A RUNTIME COMPARISON

We compared the average run-time of a batch of size 128 between the proposed CARCA models and the most recent state-of-the-art sequential baseline models on the Games dataset using 1080Ti GPU, and E5-1660v4 CPU with 64GB of RAM. Results in table 6 show that the average batch run-time of CARCA is very close to other sequential models with very little overhead due to adding the cross-attention component and the additional features. Results also show that CARCA is about **22 times** faster than S^3Rec, which is the closest competitor because of S^3Rec's computationally expensive mutual information maximization procedure.

B REPRODUCIBILITY OF THE EXPERIMENTS

The source code of CARCA[2] is available at github. Regarding the hyper-parameters, all models were tuned using a grid search except the EASE model as it was optimized using a Bayesian optimization procedure existed in its source code[3] by Dacrema et al. [2]. The best found hyper-parmeters for our CARCA models is shown in Table 5

The source codes of all baseline models are available at their authors' GitHub repositories [4][5][6][7][8][9][10][11][12].

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *stat* 1050 (2016), 21.

[2] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are we really making much progress? A worrying analysis of recent neural recommendation approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 101–109.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[5] Maurizio Ferrari Dacrema, Federico Parroni, Paolo Cremonesi, and Dietmar Jannach. 2020. Critically Examining the Claimed Value of Convolutions over User-Item Embedding Maps for Recommender Systems. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 355–363.

[6] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. In *IJCAI*.

[7] 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.

[8] Ruining He and Julian McAuley. 2016. VBPR: visual bayesian personalized ranking from implicit feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[9] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 355–364.

[10] Balázs Hidasi and Alexandros Karatzoglou. 2018. Recurrent neural networks with top-k gains for session-based recommendations. In *Proceedings of the 27th ACM international conference on information and knowledge management*. 843–852.

[11] Min Hou, Le Wu, Enhong Chen, Zhi Li, Vincent W. Zheng, and Qi Liu. 2019. Explainable Fashion Recommendation: A Semantic Attribute Region Guided Approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 4681–4688. https://doi.org/10.24963/ijcai.2019/650

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

[13] 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.

[14] Sheng Li, Jaya Kawale, and Yun Fu. 2015. Deep collaborative filtering via marginalized denoising auto-encoder. In *Proceedings of the 24th ACM international on conference on information and knowledge management*. 811–820.

[15] 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.

[16] Ahmed Rashed, Josif Grabocka, and Lars Schmidt-Thieme. 2019. Attribute-aware non-linear co-embeddings of graph features. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 314–321.

[17] Ahmed Rashed, Shayan Jawed, Lars Schmidt-Thieme, and Andre Hintsches. 2020. MultiRec: A Multi-Relational Approach for Unique Item Recommendation in Auction Systems. In *Fourteenth ACM Conference on Recommender Systems*. 230–239.

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

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

[20] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1161–1170.

[21] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[22] Harald Steck. 2019. Embarrassingly shallow autoencoders for sparse data. In *The World Wide Web Conference*. 3251–3257.

[23] 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.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[25] 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.

[26] Jianling Wang, Raphael Louca, Diane Hu, Caitlin Cellier, James Caverlee, and Liangjie Hong. 2020. Time to Shop for Valentine's Day: Shopping Occasions and Sequential Recommendation in E-commerce. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 645–653.

[27] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2019. Stochastic Shared Embeddings: Data-driven Regularization of Embedding Layers. *NeurIPS* (2019).

[2] https://github.com/ahmedrashed-ml/CARCA
[3] https://github.com/MaurizioFD/RecSys2019_DeepLearning_Evaluation
[4] https://github.com/ahmedrashed-ml/GraphRec
[5] https://github.com/ChenglongChen/tensorflow-DeepFM
[6] https://github.com/kang205/SASRec
[7] https://github.com/wangjlgz/Occasion-Aware-Recommenation
[8] https://github.com/wuliwei9278/SSE-PT/tree/master/SSE-SASrec
[9] https://github.com/wuliwei9278/SSE-PT
[10] https://github.com/RUCAIBox/CIKM2020-S3Rec
[11] https://github.com/FeiSun/BERT4Rec
[12] https://github.com/JiachengLi1995/TiSASRec

Table 5: Hyper-parameters configurations and search space of the proposed models.

Model	param	Dataset Best Parameters				Search Space
		Men	Fashion	Games	Beauty	
SASRec++	Learning Rate	0.00001	0.00001	0.0001	0.00002	{0.001, 0.0001, 0.00002, 0.00001}
	Max Seq. Length	50	35	50	50	[10, 200], Step size = 5
	Number of attention blocks	3	3	1	3	{1, 2, 3, 4, 5}
	Number of attention heads	3	1	1	1	{1, 2, 3, 4, 5}
	Dropout Rate	0.8	0.5	0.5	0.5	[0.0, 1.0], Step size = 0.1
	L2 reg. weight	0	0	0.001	0.001	[0, 0.001,0.0001,0.00001, 0.000001]
	Embeding Dimension d	60	30	50	90	[10, 500], Step size = 10
	Embeding Dimension g	300	150	250	450	[50, 500], Step size = 50
CARCA	Learning Rate	0.000006	0.00001	0.0001	0.0001	{0.001, 0.0001, 0.00001, 0.000006, 0.000002, 0.000001}
	Max Seq. Length	35	35	50	75	[10, 200], Step size = 5
	Number of attention blocks	3	3	3	3	{1, 2, 3, 4, 5}
	Number of attention heads	3	3	3	1	{1, 2, 3, 4, 5}
	Dropout Rate	0.3	0.3	0.5	0.5	[0.0, 1.0], Step size = 0.1
	L2 reg. weight	0.0001	0.0001	0	0.0001	[0, 0.001,0.0001,0.00001, 0.000001]
	Embeding Dimension d	390	390	90	90	[10, 500], Step size = 10
	Embeding Dimension g	1950	1950	450	450	[50, 2000], Step size = 50
	Residual connection in the cross-attention block	No	No	Yes	Yes	{No, Yes}

Table 6: Runtime Comparison on Games Dataset

Model	Average batch runtime in seconds
S^3Rec	0.580
SSE-PT	0.008
SASRec	0.013
TiSASRec	0.075
SSE-SASRec	0.015
OAR	0.018
SASRec++	0.028
CARCA (w/o CA)	0.015
CARCA	0.026

[28] 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.

[29] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 3119–3125.

[30] Xin Xin, Bo Chen, Xiangnan He, Dong Wang, Yue Ding, and Joemon Jose. 2019. CFM: Convolutional Factorization Machines for Context-Aware Recommendation.. In *IJCAI*, Vol. 19. 3926–3932.

[31] Shuai Zhang, Lina Yao, and Xiwei Xu. 2017. AutoSVD++ An Efficient Hybrid Collaborative Filtering Model via Contractive Auto-encoders. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. 957–960.

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

[33] Yongfeng Zhang, Qingyao Ai, Xu Chen, and W Bruce Croft. 2017. Joint representation learning for top-n recommendation with heterogeneous information sources. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 1449–1458.

[34] 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.

[35] Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. 2018. Deep interest network for click-through rate prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1059–1068.

[36] 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 *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*. ACM, 1893–1902.