RECOMMENDER TRANSFORMERS WITH BEHAVIOR PATHWAYS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sequential recommendation requires the recommender to capture the evolving behavior characteristics from logged user behavior data for accurate recommendations. Nevertheless, user behavior sequences are viewed as a script with multiple ongoing threads intertwined. We find that only a small set of pivotal behaviors can be evolved into the user's future action. As a result, the future behavior of the user is hard to predict. We conclude this characteristic for sequential behaviors of each user as the *behavior pathway*. Different users have their unique behavior pathways. Among existing sequential models, transformers have shown great capacity in capturing global-dependent characteristics. However, these models mainly provide a dense distribution over all previous behaviors using the self-attention mechanism, making the final predictions overwhelmed by the trivial behaviors not adjusted to each user. In this paper, we build the Recommender Transformer (RETR) with a novel Pathway Attention mechanism. RETR can dynamically plan the behavior pathway specified for each user, and sparingly activate the network through this behavior pathway to effectively capture evolving patterns useful for recommendation. The key design is a learned binary route to prevent the behavior pathway from being overwhelmed by trivial behaviors. Pathway attention is model-agnostic and can be applied to a series of transformer-based models for sequential recommendation. We empirically evaluate RETR on seven intra-domain benchmarks and RETR yields state-of-the-art performance. On another five cross-domain benchmarks, RETR can capture more domain-invariant representations for sequential recommendation.

1 INTRODUCTION

Recommender systems (Hidasi et al., 2015; Lu et al., 2015; Zhao et al., 2021) have been widely adopted in real-world industrial applications such as E-commerce and social media. Benefiting from the increase in computing power and model capacity, some recent efforts formulate recommendation as a time-series forecasting problem, known as *sequential recommendation* (Kang & McAuley, 2018; Sun et al., 2019; Chen et al., 2021). The core idea of this field is to infer upcoming actions based on user's historical behaviors, which are reorganized as time-ordered sequences. This intuitive modeling of recommendation is proved time-sensitive and context-aware to make precise predictions.

Recent advanced sequential recommendation models, such as SASRec (Kang & McAuley, 2018), Bert4Rec (Sun et al., 2019) and S3-Rec (Zhou et al., 2020), have achieved significant improvements. Transformers enable these models to recognize global-range sequential patterns, and to model how future behaviors are anchored in historical ones. The self-attention mechanism does make it possible to explore all previous behaviors of each user, with the whole neural network activated. However, misuse of all user information, regardless of whether it is informative or not, floods models with trivial ones, makes models dense in neuron connections and inefficient in computation, and results in key behaviors losing voice. And this clearly contradicts with the way our brain works.

The human being has many different parts of the brain specialized for various tasks, yet the brain only calls upon the relevant pieces for a given situation (Zaidi, 2010). To some extent, user behavior sequences can be viewed as a script with multiple ongoing threads intertwined. And only key clues suggest what will happen next. In sequential recommendation, we find that only a small part of pivotal behaviors can be evolved into the user's future action. And we conclude this characteristics of sequential behaviors as the *behavior pathway*.

Figure 1: Three typical examples of the behavior pathway for different users: correlated, causal, and drifted. The behavior pathway is outlined by the red boxes.

Different users have their unique behavior pathways, and we have provided three typical examples: (a) *Correlated behavior pathway*: A user's behavior pathway is closely associated with behaviors in a certain period. As shown in the first line of Figure 1, the mouse is clicked many times recently, leading to the final decision to buy a mouse. (b) *Casual behavior pathway*: A user's behavior pathway is interested in a specific item at casual times. In the second line of Figure 1, the backpack is randomly clicked sequentially in a multi-hop manner. (c) *Drifted behavior pathway*: A user's behavior pathway in a particular brand might drift over time. In the third line of Figure 1, the user was initially interested in a keyboard, but suddenly became interested in buying a phone at last. It's challenging to capture these potential behaviors dynamically for each user to make precise recommendations.

Motivated by the Pathways (Dean, 2021), a new way of thinking about AI, which builds a single model that is sparsely activated for all tasks with small pathways through the network called into action as needed, we propose a novel Recommender Transformer (RETR) with a Pathway Attention mechanism. RETR dynamically explores behavior pathways for different users and then captures evolving patterns through these pathways effectively. Specifically, the user-dependent pathway attention, which incorporates a pathway router, determines whether or not a behavior token will be maintained in the behavior pathway. Technically, the pathway router generates a customized binary route for each token based on their information redundancy. RETR has a stacked structure, and successive pathway routers constitute a hierarchical evolution of user behaviors. To enable the pathway router to be end-to-end optimized, we propose an adaptive Gumbel-Softmax sampling strategy to overcome the non-differentiable problem of sampling from a Bernoulli distribution.

To effectively capture the evolving patterns via the behavior pathway, our pathway attention mechanism makes RETR mainly attend to the obtained pathway. We force the model to focus on the most informative behaviors by using the query routed through the behavior pathway. We cut off the interaction from the off-pathway behaviors of the query. Compared with using all previous behaviors, our pathway attention mechanism is obviously more effective and can avoid the most informative tokens being overwhelmed by trivial behaviors. Besides, our pathway attention mechanism is model-agnostic and can be easily applied to the existing transformer-based models. To validate the effectiveness of our approach, we conduct experiments on seven intra-domain competitive datasets for sequential recommendations and RETR achieves state-of-the-art performance; Furthermore, our RETR also achieves consistent performance improvements under the cross-domain setting, indicating RETR can capture more domain-universal representation for sequential recommendation. Our main contributions can be summarized as follows:

- We first propose the concept of behavior pathway for sequential recommendation, and find the key to the recommender is to dynamically capture the behavior pathway for each user.
- We propose the novel Recommender Transformer (RETR) with a novel pathway attention mechanism, which can generate the behavior pathway hierarchically and capture the evolving patterns dynamically through the pathway.
- We validate the effectiveness of RETR on seven intra-domain benchmarks and five cross-domain benchmarks, both achieving state-of-the-art performance. RETR can capture more domain-invariant representations and our pathway attention can be applied together with a rich family of transformer-based models to yield consistent performance improvements.

2 RELATED WORK

Traditional recommendation approaches. Capturing evolving behavior characteristics is crucial for many online applications, such as advertising, social media and E-commerce, and it is the key challenge for sequential recommendation (Adomavicius & Tuzhilin, 2005; Kang & McAuley, 2018; Cui et al., 2018; Yan et al., 2019; Fang et al., 2020; Pi et al., 2020; Bian et al., 2021; Zhou et al., 2022; Liu et al., 2022). Traditional recommendation approach, such as the collaborative filtering (CF) (Herlocker et al., 1999) based on matrix approximation (Koren, 2008; Koren et al., 2009), always assumes that the user's behavior is static. However, in practice, user behaviors often change over time due to various reasons, making the CF deteriorate in a real-world application.

Sequential recommendation approaches. To overcome this challenge, some methods, such as FPMC (He & McAuley, 2016) and HRM (Wang et al., 2015), use Markov chains to capture sequential patterns by learning user-specific transition matrices. Higher-order Markov Chains assume the next action is related to several previous actions. Benefit from this strong inductive bias, MC-based methods (He & McAuley, 2016; He et al., 2016) show superior performance in capturing short-term patterns. At the same time, there is a potential state space explosion problem when these approaches are faced with different possible sequences (Wu et al., 2017). In recent years, many works have been using the deep neural network for sequential recommendation. The GRU4Rec (Hidasi et al., 2015) and the RepeatNet (Ren et al., 2019) adopt the recurrent network to capture dynamic patterns from the user behaviors dependent on sequence positions. The RNN-based models achieve competitive performance in capturing short-term behavior patterns but cannot capture long-term behavior patterns effectively. The CNN-based model, such as Caser (Tang & Wang, 2018), applies convolutional operations to extract transitions while tending to overlook the intrinsic relationship across user behaviors. The GNN-based methods, such as SRGNN (Wu et al., 2019), GCSAN (Xu et al., 2019), Jodie (Kumar et al., 2019) and TGN (Rossi et al., 2020) model behavior sequences as graph-structured data and incorporate an attention mechanism for a session-based recommendation. In addition, DIN (Zhou et al., 2018) uses the gate mechanism to weight different user behaviors. However, concatenating all behaviors makes these models overlook the sequential characteristics. Recently, the MLP-based model like FMLP-Rec (Zhou et al., 2022) uses the MLP as the backbone for sequential recommendation. However, these methods are still overwhelmed by the trivial behaviors.

Transformer-based models for Sequential Recommendation. SASRec (Kang & McAuley, 2018), BertRec (Sun et al., 2019), S3-Rec (Zhou et al., 2020), TGSRec (Fan et al., 2021b), LightSANs (Fan et al., 2021a) and SSE-PT (Wu et al., 2020) introduce the transformer architecture into sequential recommendation, which might lead to the over-parameterized architecture of Transformer-based methods. These models capture the evolving patterns by the self-attention mechanism, interacting with all previous behaviors. However, dense interactions will make the model not adapt to different users and overwhelm behavior pathways. Some methods like Locker (He et al., 2021) and Rec-denoiser (Chen et al., 2022) propose the sparse attention mechanism with learned mask, while they may overlook the ability of capturing the behavior pathway in the token level. To tackle this challenge, our paper builds the Recommender Transformer (RETR) with a new Pathway Attention mechanism that is dynamically activated for the behavior pathway of all users. Distinct from the previous routing architecture like Switch Transformer (Fedus et al., 2021) using the MoE (Shazeer et al., 2017) structure for natural language tasks or TRAR (Zhou et al., 2021) using the learned sparse attention for visual question answering, our RETR is designed explicitly for sequential recommendation. Our RETR uses the pathway router to adaptively route the sequential behavior of each user rather than routing the experts of feed-forward networks in switch transformer.

3 METHOD

Suppose that we have a set of users and items, denoted by $\mathcal{U}$ and $\mathcal{I}$ respectively. In the task of sequential recommendation, chronologically-ordered behaviors of a user $u \in \mathcal{U}$ could be represented by a user-interacted item sequence: $\{i_1, \cdots, i_n\}$. Formally, given a user u with her or his behavior sequence $\{i_1, \cdots, i_n\}$, the goal of sequential recommendation is to predict the next item the user u would interact with at the $(n+1)$-th step, denoted as $p\left(i_{n+1} \mid i_{1:n}\right)$.

As aforementioned, we highlight the key to sequential recommendation as the exploration of user-tailored behavior pathways, through which evolving characteristics could be learned. Motivated by

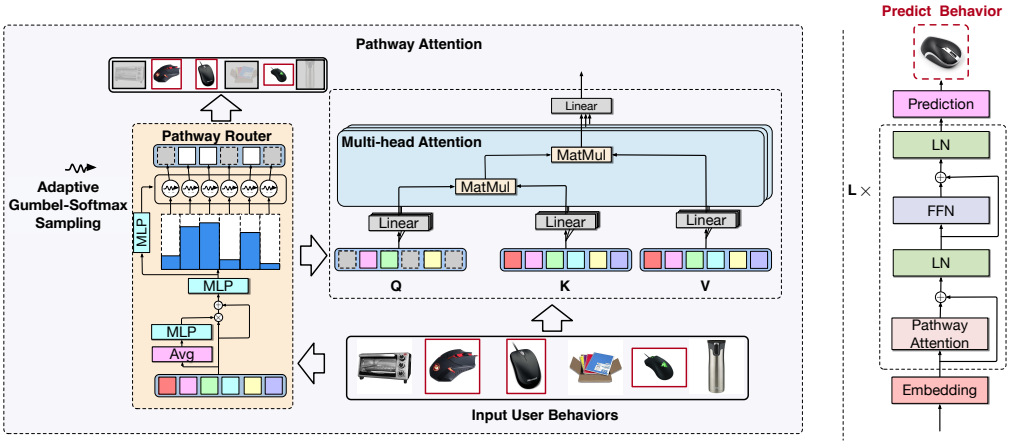

Figure 2: The architecture of Recommender Transformer (RETR) on the right subfigure. Pathway Attention (left) explores the behavior pathway by the pathway router (orange module) and captures the evolving sequential characteristics of the use behaviors by the multi-head attention.

this, we propose a novel *Recommender Transformer* (RETR) with a new *Pathway Attention*, the core subassembly of which is a pathway router. Besides the modification of architecture, we additionally introduce a hierarchical update strategy for the behavior pathway in the feed-forward procedure.

3.1 Recommender Transformer

Considering the limitation of overwhelming attention in Transformers (Bhattacharjee & Das, 2017) for sequential recommendation, we renovate the vanilla architecture to the Recommender Transformer (RETR) with a Pathway Attention mechanism, as shown in Figure 2.

Model inputs. To obtain the model inputs, we follow the sliding window practice and transform the user's behavior sequence into a fixed-length-N sequence $s = (s_1, s_2, \ldots, s_N)$. Then we produce an item embedding matrix $\mathbf{E}_{\mathcal{I}} \in \mathbb{R}^{|\mathcal{I}| \times d}$, where d is the embedding dimensionality. We perform a look-up operation from $\mathbf{E}_{\mathcal{I}}$ to retrieve the input embedding matrix $\mathbf{E}_s \in \mathbb{R}^{N \times d}$ for sequence s. Besides, we also add a learnable position embedding $\mathbf{P}_s \in \mathbb{R}^{N \times d}$ for sequence s. Finally, we can generate the input embedding of each behavior sequence s as $\mathbf{X}_s = \mathbf{E}_s + \mathbf{P}_s \in \mathbb{R}^{N \times d}$.

Overall architecture. Recommender Transformer is characterized by stacking the Pathway Attention blocks and feed-forward layers alternately, containing L blocks. This stacking structure is conducive to learning behavior representations hierarchically. The overall equations of block l are formalized as:

$$\begin{aligned} \widehat{\mathbf{Z}}^l, \mathcal{R}^l &= \text{Path-MSA}\left(\mathbf{Z}^{l-1}, \mathcal{R}^{l-1}\right) \\ \widehat{\mathbf{Z}}^l &= \text{LN}\left(\widehat{\mathbf{Z}}^l + \mathbf{Z}^{l-1}\right) \\ \mathbf{Z}^l &= \text{LN}\left(\text{FFN}\left(\widehat{\mathbf{Z}}^l\right) + \widehat{\mathbf{Z}}^l\right), \end{aligned} \tag{1}$$

where $\mathbf{Z}^l \in \mathbb{R}^{N \times d}, l \in \{1, \cdots, L\}$ denotes the output of the l-th block. The initial input $\mathbf{Z}^0 = \mathbf{X}_s \in \mathbb{R}^{N \times d}$ represents the raw behavior embedding. $\mathcal{R}^{l-1} \in \mathbb{R}^{N \times 1}$ is the previous route from the $(l-1)$-th block and we initialize all elements in the route $\mathcal{R}^0$ to 1. Path-MSA$(\cdot)$ is to conduct the pathway multi-head self-attention. LN$(\cdot)$ is to conduct layer normalization (Ba et al., 2016) and FFN represents the point-wise feed-forward network (Bhattacharjee & Das, 2017).

3.1.1 Pathway Attention

Note that the single-branch self-attention mechanism (Bhattacharjee & Das, 2017) in vanilla transformer cannot model the behavior pathway dynamically, resulting in key behaviors being overwhelmed by those non-pivotal or trivial ones. To solve this problem, we propose the Pathway Attention mechanism, as shown in Figure 2, which can dynamically attend to the behavior pathway of pivotal behavior tokens.

Pathway router. The pathway attention employs a sequence-adaptive pathway router to custom-tailor behavior pathway routes for users. The router generates a binary route $\mathcal{R}^l \in \{0, 1\}^N$ to determine

whether a behavior token would be part of the behavior pathway or not. Each router takes the pre-order route $\mathcal{R}^{l-1}$ and user behavior tokens $\mathbf{Z}^{l-1} \in \mathbb{R}^{N\times d}$ of the $(l-1)$-th block as its inputs. All elements in the route are initialized by 1 and are updated progressively in training.

Foremost, to suppress the potential disturbance to the model caused by the local drifted interest (Figure 1), it is crucial to incorporate the *global* information in the route generation. We apply the average pooling to all the preserved behavior tokens routed by $\mathcal{R}^{l-1}$, and produce the global sequential representation via a multilayer perceptron (MLP) module. Then, we combine this global representation with the inputs and employ a residual connection to maintain the original input information. Finally, we feed them to another MLP layer to predict the probabilities of keeping or dropping the behavior tokens. All MLP layers are column-wise and operate on the embedding dimensionality. The above procedure can be formulated as follows:

$$\begin{aligned} \mathbf{Z}^l_{\text{emb}} &= \mathbf{Z}^{l-1} + \mathbf{Z}^{l-1} \odot \text{MLP}\left(\frac{\sum_{i=1}^{N} \mathcal{R}_i^{l-1}\mathbf{Z}_i^{l-1}}{\sum_{i=1}^{N} \mathcal{R}_i^{l-1}}\right) \\ \boldsymbol{\pi} &= \text{Softmax}\left(\text{MLP}(\mathbf{Z}^l_{\text{emb}})\right) \in \mathbb{R}^{N\times 2}, \end{aligned} \tag{2}$$

where $\odot$ is the Hadamard product. For $t \in \{1, 2, \cdots, N\}$, we let $\boldsymbol{\pi}_t = [1-\alpha_t, \alpha_t]$, where the logit α_t denotes the probability that the t-th behavior token is kept alive for the behavior pathway.

Adaptive Gumbel-Softmax sampling from $\boldsymbol{\pi}$ for router. Our goal is to generate the binary route from $\boldsymbol{\pi}$. However, sampling from $\boldsymbol{\pi}$ directly is non-differentiable, and it will impede the gradient-based training. Gumbel-Softmax (Jang et al., 2016) is an effective way to approximate the original non-differentiable sample from a discrete distribution with a differentiable sample from a Gumbel-Softmax distribution. Thus, we adapt the Gumbel-Softmax technique to achieve such a sampling procedure. Instead of directly sampling a keep-or-drop decision $\widehat{\mathcal{R}}_t^l$ for the t-th behavior token from the distribution $\boldsymbol{\pi}_t$, we generate it as:

$$\widehat{\mathcal{R}}_t^l = \underset{j\in\{0,1\}}{\arg\max} \left(\log \pi_t(j) + G_t(j)\right), \tag{3}$$

where $G_t = -\log(-\log U_t)$ is a standard Gumbel distribution, and U_t is sampled i.i.d. from a uniform distribution $\text{Uniform}(0, 1)$. To remove the non-differentiable argmax operation in equation 3, the standard Gumbel-Softmax uses the reparameterization trick (Jang et al., 2016) as a differentiable approximation to relax the one-hot $\widehat{\mathcal{R}}_t^l \in \{0, 1\}$ to $v_t \in \mathbb{R}^2$:

$$v_t(j) = \frac{\exp((\log \pi_t(j) + G_t(j))/\tau)}{\sum_{i\in\{0,1\}} \exp((\log \pi_t(i) + G_t(i))/\tau)}, j \in \{0, 1\}, \tag{4}$$

where τ is the temperature parameter of the Softmax. However, it remains a well-known challenge to tune the temperature in Gumbel-Softmax since a low temperature will cause a high variance in gradient magnitude and a high temperature will lead to an over-smoothing probability. Furthermore, a fixed temperature is not adaptive across different datasets or behaviors of each user, which incurs huge tweaking cost. Motivated by these difficulties, our propose an adaptive variant of Gumbel-Softmax that introduces the token-specific weight mechanism into the Gumbel-Softmax:

$$\begin{aligned} \boldsymbol{\omega} &= \text{ReLU}\left(\text{MLP}\left(\mathbf{Z}^l_{\text{emb}}\right)\right) \in \mathbb{R}^{N\times 1}, \\ v_t(j) &= \frac{\exp(\omega_t \log \pi_t(j) + G_t(j))}{\sum_{i\in\{0,1\}} \exp(\omega_t \log \pi_t(i) + G_t(i))}, j \in \{0, 1\}, \end{aligned} \tag{5}$$

where ω_t is the weight specific for each token t of each sequence. For different user behaviors, we use an MLP module to dynamically introduce the weight from the inputs $\mathbf{Z}^l_{\text{emb}}$, avoiding the high variance in the gradient and mitigating the over-smoothing phenomenon. Our adaptive Gumbel-Softmax can make RETR dynamically adapt to diverse datasets and user behaviors without tuning the temperature.

Hierarchical update strategy for router. The preliminary route $\widehat{\mathcal{R}}^l$, sampled from $\boldsymbol{\pi}$, is not a final decision. In our design, once a token fails to be routed in a certain block, it would permanently lose the privilege to be part of the behavior pathway in the following feed-forward procedure. This constitutes a more efficient hierarchical pathway router strategy. Thus finally we formulate the route $\mathcal{R}^l \in \mathbb{R}^{N\times 1}$ as the Hadamard product of $\widehat{\mathcal{R}}^l$ and the pre-order route $\mathcal{R}^{l-1}$ in the $(l-1)$-th block:

$$\mathcal{R}^l = \widehat{\mathcal{R}}^l \odot \mathcal{R}^{l-1}. \tag{6}$$

Multi-head pathway attention. The standard multi-head self-attention mechanism retrieves sequential characteristics by exploiting all behavior tokens, making the behavior pathway overwhelmed by the trivial behaviors. In the proposed pathway attention, the pathway router would be firstly applied to the input behavior tokens to route information. The pathway router would not pare down the number of tokens, but only the interactions between the off-pathway and on-pathway tokens, as these off-pathway tokens may also convey contextual information.

Specifically, for the query $\mathbf{Q}$, key $\mathbf{K}$, and value $\mathbf{V}$ in the pathway attention: the query is routed by the pathway router through token-wise multiplication between $\mathcal{R}_t^l$ and $\mathbf{Z}_t^{l-1}$, to prevent the pathway from being overwhelmed and to force the pathway attention to attend to the behavior pathway. The key and value are the original input behavior tokens, to ensure that the contextual information from off-pathway behavior tokens can be captured as well:

$$\begin{aligned} \mathbf{Q}_m, \mathbf{K}_m, \mathbf{V}_m &= (\mathcal{R}^l \mathbf{Z}^{l-1}) \mathbf{W}^l_{\mathbf{Q}_m}, \mathbf{Z}^{l-1} \mathbf{W}^l_{\mathbf{K}_m}, \mathbf{Z}^{l-1} \mathbf{W}^l_{\mathbf{V}_m} \\ \widehat{\mathbf{Z}}^l_m &= \text{Softmax}\left(\frac{\mathbf{Q}_m \mathbf{K}_m^{\mathrm{T}}}{\sqrt{d/h}} \right) \mathbf{V}^l_m, \end{aligned} \tag{7}$$

where $m \in \{1, 2, \cdots, h\}$ is the head index in the multi-head self-attention; $\mathbf{W}^l_{\mathbf{Q}_m}, \mathbf{W}^l_{\mathbf{K}_m}, \mathbf{W}^l_{\mathbf{V}_m} \in \mathbb{R}^{d \times \frac{d}{h}}$ are transformation matrices learned from data. Finally, the outputs $\left\{ \widehat{\mathbf{Z}}^l_m \in \mathbb{R}^{N \times \frac{d}{h}} \right\}_{1 \le m \le h}$ of multiple heads are concatenated into $\widehat{\mathbf{Z}}^l \in \mathbb{R}^{N \times d}$. We use $\widehat{\mathbf{Z}}^l, \mathcal{R}^l = \text{Path-MSA}\left(\mathbf{Z}^{l-1}, \mathcal{R}^{l-1}\right)$ to summarize the above pathway attention. Its output is further transformed by equation 1 to form the final output of the l-th block $\mathbf{Z}^l \in \mathbb{R}^{N \times d}$.

In the prediction of the $(t+1)$-th behavior, only the first t observable behaviors should be taken into account. To avoid a future information leak and ensure causality, we apply a causal pathway attention in that a look-ahead mask is employed and all links between $\mathbf{Q}_j$ and $\mathbf{K}_i$ $(j > i)$ are removed.

Model-agnostic pathway mechanism. It is worth noting that our pathway attention is a lightweight module readily pluggable into any transformer-based model by replacing the self-attention mechanism with our *pathway attention* while remaining the architecture unchanged. To verify the effectiveness of our model-agnostic pathway mechanism, we apply our pathway attention to mainstream sequential recommendation transformers: BERTRec (Sun et al., 2019), SASRec (Kang & McAuley, 2018), SMRec (Chen et al., 2021), S3-Rec (Zhou et al., 2020), TGSRec (Fan et al., 2021b), and LightSANs (Fan et al., 2021a), which further enhances the performance and generalization of the these models.

3.2 Prediction Layer and Training Objective

Prediction layer. In the final layer of RETR, we calculate the user's preference score for the item k in the step $(t+1)$ in the context of user behavior history as $p\left(i_{t+1} = k \mid i_{1:t}\right) = e_k \cdot \mathbf{Z}_t^L$, where e_k is the representation of item k from item embedding matrix $\mathbf{E}_{\mathcal{I}}$, and $\mathbf{Z}_t^L$ is the output of the L-th block in RETR at step t, with L being the number of RETR blocks.

Training objective. We adopt the pairwise ranking loss to optimize the RETR model parameters as:

$$\mathcal{L} = -\sum_{u \in \mathcal{U}} \sum_{t=1}^{n} \log \sigma\big(p(i_{t+1} \mid i_{1:t}) - p(i^-_{t+1} \mid i_{1:t})\big), \tag{8}$$

where we pair each ground-truth item i_{t+1} with a randomly sampled negative item i^-_{t+1}. In each epoch, we randomly generate one negative item for each time step in each sequence. This pairwise ranking loss is widely adopted in previous literature of sequential recommendation (Kang & McAuley, 2018; Zhou et al., 2022).

4 Experiments

We extensively evaluate the proposed Recommender Transformer (RETR) on seven intra-domain real-world benchmarks and five cross-domain benchmarks. Due to page limitation, we also include further ablation study results and visualization examples in Appendix B and Appendix C respectively.

Intra-domain setting. We evaluate RETR on seven intra-domain datasets: Netflix, MSD, Taobao, Yelp, Tmall, Steam, and MovieLens1M. All methods are trained from scratch on these datasets. The

statistics of the seven datasets are summarized in Table 4 of Appendix A and the description for these datasets can be found therein. All datasets are widely used for sequential recommendation task. It is notable that Netflix, MSD, Taobao and Steam are large-scale datasets.

Cross-domain setting. We evaluate the ability of RETR to capture the *domain-invariant* representation for sequential recommendation under the cross-domain setting. Specifically, we follow the training strategy in UniSRec (Hou et al., 2022) to pre-train our RETR on multiple datasets "Grocery and Gourmet Food", "Home and Kitchen", "CDs and Vinyl", "Kindle Store" and "Movies and TV", and then finetune the pre-trained RETR respectively on different target datasets "Prime Pantry", "Industrial and Scientific", "Musical Instruments", "Arts, Crafts and Sewing" and "Office Products". These datasets are all sub-categories in the Amazon review datasets (Ni et al., 2019). The detailed descriptions of source and target datasets are deferred to Appendix A.

Following previous works (Kang & McAuley, 2018), we group the interaction records by users or sessions for all datasets and sort them by the timestamps in ascending order. We split the historical sequence for each user into three parts: (1) the most recent behavior for testing, (2) the second most recent behavior for validation, and (3) all remaining behaviors for training. During testing, the input sequences contain training behaviors and validation behaviors. We filter less popular items and inactive users with fewer than five interaction records. Note that we also provide the rolling prediction results following the protocol from previous works (Chen et al., 2021) in Appendix C.

Evaluation metrics. Following the previous literature (Zhou et al., 2022; Kang & McAuley, 2018), we apply top-k Hit Ratio (HR@k), top-k Normalized Discounted Cumulative Gain (NDCG@k) and Mean Reciprocal Rank (MRR) for evaluation. We report HR@10, NDCG@10 and MRR of the results. Besides, following the standard strategy in SASRec (Kang & McAuley, 2018), we pair the ground-truth item with 100 randomly sampled negative items that the user has not interacted with. All metrics are calculated according to the ranking of the items and we report the average score.

Baseline methods. We compare our RETR with several state-of-the-art sequence recommendation models. Specifically, we compare our RETR with state-of-the-art transformer-based sequential recommendation models: Rec-denoiser (Chen et al., 2022), Locker (He et al., 2021), SASRec (Kang & McAuley, 2018), BertRec (Sun et al., 2019), SMRec (Chen et al., 2021). S3-Rec (Zhou et al., 2020), TGSRec (Fan et al., 2021b) and LightSANs (Fan et al., 2021a). These methods adopt the attention mechanism to make precise recommendations. Note that Rec-denoiser (Chen et al., 2022) and Locker (He et al., 2021) are novel transformer-based models with learnable sparse attention. Besides, we also compare our RETR with state-of-the-art graph-based sequential recommendation methods: Jodie (Kumar et al., 2019) and TGN (Rossi et al., 2020). We further compare our approach with some cross-domain recommendation models RecGURU (Li et al., 2022) and UniSRec (Hou et al., 2022). All baseline methods are configured using default parameters of the original paper or optimal parameters which can produce their best results through a grid search.

Implementation details. Our model is supervised by the pairwise rank loss in equation 8, using the ADAM (Kingma & Ba, 2015) optimizer with an initial learning rate of 0.001. Batch size is set to 512. The maximum number of training epochs for all methods is set to 300. All hyperparameters are tuned on the validation set. The training process is early stopped within 10 epochs. Our RETR has $L = 2$ layers, and each layer has $h = 4$ heads (the ablation study of multi-head attention can be found in Appendix B) and dimension d is set to be 256. The maximum sequence length N is set to 200 for MovieLens1M and 100 for the other intra-domain and cross-domain datasets.

In the cross-domain setting, we further pre-train the proposed approach and baseline methods from multiple datasets following the training strategy in UniSRec (Hou et al., 2022) for 300 epochs using the default parameters from UniSRec (Hou et al., 2022). All models are pretrained without using the item ID embeddings. In the phase of finetuning on the target domain, we use the item ID embeddings and fix the backbone for all competing pretrained models. All experiments are repeated three times, implemented in PyTorch (Paszke et al., 2019), and conducted on a single NVIDIA 3090 GPU.

4.1 Intra-domain Results

The results of different methods on seven intra-domain datasets are shown in Table 1. We can easily find that transformer-based models, SASRec (Kang & McAuley, 2018), BertRec (Sun et al., 2019), SMRec (Chen et al., 2021), S3-Rec (Zhou et al., 2020), TGSRec (Fan et al., 2021b) and

Table 1: Performance comparison to state-of-the-art models under the intra-domain setting. Rec-denoiser (Chen et al., 2022) uses BertRec as the backbone.

Datasets	Meric	BERT4Rec	SASRec	SMRec	S3-Rec	SINE	TGSRec	LightSAN	Locker	Rec-denoiser	Jodie	TGN	RETR
Netflix	HR@10	0.4792	0.4622	0.4848	0.4917	0.4902	0.4887	0.4852	0.4897	0.4913	0.4813	0.4802	**0.5184**
	NDCG@10	0.3330	0.3202	0.3492	0.3571	0.3601	0.3512	0.3441	0.3557	0.3582	0.3368	0.3318	**0.3795**
	MRR	0.2652	0.2519	0.2725	0.2819	0.2796	0.2778	0.2785	0.2873	0.2886	0.2687	0.2612	**0.3175**
MSD	HR@10	0.4819	0.4766	0.5083	0.5315	0.5264	0.5137	0.4994	0.5495	0.5581	0.4825	0.4782	**0.5963**
	NDCG@10	0.4891	0.4831	0.5112	0.5381	0.5304	0.5279	0.5163	0.5495	0.5471	0.4872	0.4832	**0.6012**
	MRR	0.3120	0.3079	0.3302	0.3494	0.3667	0.3612	0.3451	0.3686	0.3723	0.3224	0.3102	**0.4025**
Taobao	HR@10	0.1261	0.1182	0.1272	0.1336	0.1580	0.1537	0.1590	0.1584	0.1603	0.1447	0.1421	**0.1803**
	NDCG@10	0.0425	0.0391	0.0531	0.0627	0.0873	0.0745	0.0794	0.0922	0.0952	0.0582	0.0571	**0.1218**
	MRR	0.0489	0.0436	0.0721	0.0788	0.0934	0.0802	0.0841	0.0928	0.0967	0.0628	0.0603	**0.1149**
Yelp	HR@10	0.7597	0.7373	0.7548	0.7597	0.7564	0.7533	0.7552	0.7503	0.7520	0.7492	0.7473	**0.7775**
	NDCG@10	0.4778	0.4642	0.4789	0.4937	0.4902	0.4887	0.4863	0.4935	0.4973	0.4792	0.4784	**0.5169**
	MRR	0.4026	0.3927	0.4023	0.4107	0.4093	0.4072	0.4086	0.4189	0.4214	0.3997	0.3985	**0.4378**
MovieLens	HR@10	0.8269	0.8233	0.8302	0.8352	0.8311	0.8303	0.8294	0.8349	0.8368	0.8277	0.8259	**0.8513**
	NDCG@10	0.5965	0.5936	0.6079	0.6172	0.6134	0.6081	0.6119	0.6003	0.6128	0.6009	0.5998	**0.6397**
	MRR	0.5614	0.5573	0.5703	0.5812	0.5801	0.5734	0.5791	0.5787	0.5812	0.5651	0.5627	**0.5978**
Tmall	HR@10	0.6196	0.6275	0.6476	0.6687	0.6512	0.6506	0.6399	0.6703	0.6729	0.6384	0.6362	**0.7214**
	NDCG@10	0.5025	0.5049	0.5192	0.5423	0.5411	0.5372	0.5415	0.5792	0.5830	0.5307	0.5198	**0.6197**
	MRR	0.4026	0.4804	0.4934	0.5194	0.5147	0.5121	0.5119	0.5373	0.5426	0.5003	0.4997	**0.5903**
Steam	HR@10	0.8656	0.8729	0.8792	0.8813	0.8765	0.8773	0.8832	0.8831	0.8892	0.8780	0.8731	**0.9079**
	NDCG@10	0.6283	0.6306	0.6408	0.6573	0.6502	0.6491	0.6519	0.6497	0.6523	0.6451	0.6399	**0.6835**
	MRR	0.5883	0.5925	0.6011	0.6135	0.5972	0.6003	0.6104	0.6114	0.6159	0.5873	0.5798	**0.6383**

LightSANs (Fan et al., 2021a), achieve competitive performance on most datasets, indicating that the transformer-based models have a better capacity to capture sequential behaviors of complex characteristics. These models can capture the interaction information between all previous user behaviors via the attention mechanism. Besides, the graph-based models like Jodie (Kumar et al., 2019) and TGN (Rossi et al., 2020) also achieve competitive performance. Rec-denoiser (Chen et al., 2022) and Locker (He et al., 2021) introduce novel sparse attention mechanisms, thereby achieving better performance compared with other baselines. This validates the effectiveness of the learned mask attention. While these baselines are strong competitors, our RETR can achieve state-of-the-art performance by a large margin on most datasets compared with the Rec-denoiser and Locker.

Results on Yelp, MovieLens1M and Tmall. Our RETR achieves competitive performance on Yelp and Tmall. These datasets are sparse, containing less action information. Thus they have lots of noisy logged information. By effectively capturing the behavior pathway, RETR is not affected by this trivial behavior information and captures the most informative behavior representation to achieve better performance. Note that under the Tmall benchmark, RETR gains **7%** HR@10, **12%** NDCG@10 and **14%** MRR against the strongest baseline SMRec (Chen et al., 2021). Besides, for the MoveLens1M benchmark, RETR also achieves the best performance among all competing baselines.

Results on large-scale datasets. Our RETR can consistently achieve state-of-the-art results on large-scale datasets (Netflix, MSD, Taobao, and Steam). These datasets are challenging and difficult to capture pivotal behavior pathway useful for precise recommendation from the rich but noisy user's behaviors. Especially for the Taobao dataset, RETR gains relative improvements of **12%** HR@10, **37%** NDCG@10 and **20%** MRR against the strongest baseline SINE (Tan et al., 2021). It provides evidence that RETR can achieve competitive performance in both small- and large-scale datasets. The substantial performance gains of our RETR indicate that focusing more on the behavior pathway enables RETR to capture sequential characteristics more efficiently and effectively than the vanilla self-attention mechanism, which considers all previous user behaviors and is easily overwhelmed.

4.2 Cross-domain Results

To verify the ability of RETR to capture domain-invariant representations, we evaluate pre-trained RETR on five target datasets under the cross-domain setting. The multi-domain pre-training version of RETR, denoted as X-RETR, can be effectively transferred to new recommendation domains. We also provide the results for multi-domain pre-training version of Rec-denoiser, SASRec, SMRec, and LightSAN, denoted as X-Rec-denoiser, UniSRec, X-SMRec, and X-LightSAN respectively. Technically, we follow the pretraining strategy of UniSRec (Hou et al., 2022) to train all models. As shown in Table 2, the X-RETR already achieves competitive cross-domain performance, outperforming the state-of-the-art cross-domain method UniSRec by a large margin on most target datasets. Compared with other multi-domain pre-trained backbones, X-RETR achieves the highest performance and em-

powers better transferability among different backbones. Specially, X-RETR gains **12**% HR@10 and **22.5**% NDCG@10 compared with X-SMRec, on the Scientific benchmark. These results indicate that RETR can extract domain-invariant representations for sequential recommendation, indicating that a stronger backbone is crucial in parallel with transfer-learning method for enhancing the transferability. RETR can be regarded as a general backbone to capture more domain-invariant representations.

Table 2: Performance comparison of different competitive methods under the cross-domain setting. X-* indicates the model pre-trained on multiple datasets and finetuned on a target dataset ("X" stands for "cross-domain") following the training procedure of UniSRec (Hou et al., 2022).

Target Datasets	Meric	X-SMRec	X-LightSAN	X-Locker	X-Rec-denoiser	UniSRec	RecGURU	X-RETR
Scientific	HR@10	0.1304	0.1315	0.1312	0.1329	0.1235	0.1023	**0.1459**
	NDCG@10	0.0706	0.0725	0.0744	0.0752	0.0634	0.0572	**0.0865**
Pantry	HR@10	0.0713	0.0725	0.0741	0.0734	0.0693	0.0469	**0.0847**
	NDCG@10	0.0327	0.0321	0.0346	0.0338	0.0311	0.0209	**0.0425**
Instruments	HR@10	0.1293	0.1278	0.1316	0.1301	0.1267	0.1113	**0.1353**
	NDCG@10	0.0792	0.0778	0.0830	0.0813	0.0748	0.0681	**0.0893**
Arts	HR@10	0.1281	0.1275	0.1283	0.1293	0.1239	0.1084	**0.1378**
	NDCG@10	0.0749	0.0738	0.0749	0.0763	0.0712	0.0651	**0.0853**
Office	HR@10	0.1319	0.1321	0.1336	0.1343	0.1280	0.1145	**0.1426**
	NDCG@10	0.0842	0.0858	0.0857	0.0869	0.0831	0.0768	**0.0998**

4.3 Ablation Study

Here we provide a ablation study of RETR. For more ablation results, please refer to Appendix B, including the ablation study of different hyperparameters and effectiveness of each model component. Further visual examples are provided in Appendix C.

Pathway attention towards different transformer-based models. As described before, our RETR yields state-of-the-art performance on all datasets. We further apply our pathway mechanism towards different transformer-based models like BERTRec (Sun et al., 2019), SMRec (Chen et al., 2021), S3-Rec (Zhou et al., 2020), TGSRec (Fan et al., 2021b), and LightSANs (Fan et al., 2021a). In Table 3, we observe that our pathway attention can improve the performance of all baseline transformer-based models substantially. RETR can be further enhanced using advanced backbones alternative to the vanilla transformers and achieve the best results among all competing methods. These results provide strong evidences that our proposed pathway attention is model-agnostic to transformer-based methods and not limited to a particular architectural choice.

Table 3: Ablation study of model-agnostic pathway attention on MovieLens. Results in each column are obtained without/with pathway attention. (↑: positive improvement using pathway attention.)

Meric	BERT4Rec	SMRec	S3-Rec	TGSRec	LightSAN	RETR
HR@10	0.8269 / 0.8506 ↑	0.8302 / 0.8585 ↑	0.8352 / **0.8594** ↑	0.8303 / 0.8497 ↑	0.8294 / 0.8529 ↑	0.8513
NDCG@10	0.5965 / 0.6389 ↑	0.6079 / 0.6486 ↑	0.6172 / **0.6487** ↑	0.6081 / 0.6325 ↑	0.6119 / 0.6475 ↑	0.6397
MRR	0.5614 / 0.5998 ↑	0.5703 / 0.6031 ↑	0.5812 / **0.6052** ↑	0.5734 / 0.5973 ↑	0.5791 / 0.5993 ↑	0.5978

5 Conclusion

A sequential recommender is designed to make accurate recommendations based on users' historical behaviors. However, the users' behaviors are dynamic and come in a continually evolving manner. A user's current decision may only call upon the interest from the certain relevant behaviors of the past. We conclude these sequential characteristics as the behavior pathway. We propose the Recommender Transformer (RETR) with a novel pathway attention mechanism to tackle these challenges. The pathway attention mechanism develops a pathway router to dynamically allocate the behavior pathway for each user and capture the evolving patterns. RETR can capture more domain-invariant representations and the pathway attention is model-agnostic and can be easily applied to a series of transformer-based methods. RETR achieves state-of-the-art performance on seven intra-domain datasets and five cross-domain benchmarks for sequential recommendation.

REFERENCES

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *TKDE*, 2005.

Nabiha Asghar. Yelp dataset challenge: Review rating prediction. *arXiv preprint arXiv:1605.05362*, 2016.

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

Prateep Bhattacharjee and Sukhendu Das. Temporal coherency based criteria for predicting video frames using deep multi-stage generative adversarial networks. In *NeurIPS*, pp. 4268–4277, 2017.

Shuqing Bian, Wayne Xin Zhao, Kun Zhou, Jing Cai, Yancheng He, Cunxiang Yin, and Ji-Rong Wen. Contrastive curriculum learning for sequential user behavior modeling via data augmentation. In *CIKM*, 2021.

Chao Chen, Haoyu Geng, Nianzu Yang, Junchi Yan, Daiyue Xue, Jianping Yu, and Xiaokang Yang. Learning self-modulating attention in continuous time space with applications to sequential recommendation. In *ICML*, 2021.

Huiyuan Chen, Yusan Lin, Menghai Pan, Lan Wang, Chin-Chia Michael Yeh, Xiaoting Li, Yan Zheng, Fei Wang, and Hao Yang. Denoising self-attentive sequential recommendation. In *Proceedings of the 16th ACM Conference on Recommender Systems*, 2022.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers., arxiv. *preprint*, 2019.

Qiang Cui, Shu Wu, Qiang Liu, Wen Zhong, and Liang Wang. Mv-rnn: A multi-view recurrent neural network for sequential recommendation. *TKDE*, 2018.

Jeff Dean. Introducing pathways: A nextgeneration ai architecture. *Google Blog*, 2021.

Xinyan Fan, Zheng Liu, Jianxun Lian, Wayne Xin Zhao, Xing Xie, and Ji-Rong Wen. Lighter and better: low-rank decomposed self-attention networks for next-item recommendation. In *SIGIR*, 2021a.

Ziwei Fan, Zhiwei Liu, Jiawei Zhang, Yun Xiong, Lei Zheng, and Philip S Yu. Continuous-time sequential recommendation with temporal graph collaborative transformer. In *CIKM*, 2021b.

Hui Fang, Danning Zhang, Yiheng Shu, and Guibing Guo. Deep learning for sequential recommendation: Algorithms, influential factors, and evaluations. *TOIS*, 2020.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *arXiv preprint arXiv:2101.03961*, 2021.

Ruining He and Julian McAuley. Fusing similarity models with markov chains for sparse sequential recommendation. In *ICDM*, 2016.

Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley. Vista: A visually, socially, and temporally-aware model for artistic recommendation. In *RecSys*, 2016.

Zhankui He, Handong Zhao, Zhe Lin, Zhaowen Wang, Ajinkya Kale, and Julian McAuley. Locker: Locally constrained self-attentive sequential recommendation. In *CIKM*, 2021.

Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR*, 1999.

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

Yupeng Hou, Shanlei Mu, Wayne Xin Zhao, Yaliang Li, Bolin Ding, and Ji-Rong Wen. Towards universal sequence representation learning for recommender systems. In *KDD*, 2022.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *ICDM*, 2018.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *KDD*, 2008.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 2009.

Srijan Kumar, Xikun Zhang, and Jure Leskovec. Predicting dynamic embedding trajectory in temporal interaction networks. In *KDD*, 2019.

Chenglin Li, Mingjun Zhao, Huanming Zhang, Chenyun Yu, Lei Cheng, Guoqiang Shu, Beibei Kong, and Di Niu. Recguru: Adversarial learning of generalized user representations for cross-domain recommendation. In *WSDM*, 2022.

Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *NeurIPS*, 2019.

Yuli Liu, Christian Walder, and Lexing Xie. Determinantal point process likelihoods for sequential recommendation. *arXiv preprint arXiv:2204.11562*, 2022.

Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. Recommender system application developments: a survey. *Decision Support Systems*, 2015.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *EMNLP-IJCNLP*, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 2019.

Qi Pi, Guorui Zhou, Yujing Zhang, Zhe Wang, Lejian Ren, Ying Fan, Xiaoqiang Zhu, and Kun Gai. Search-based user interest modeling with lifelong sequential behavior data for click-through rate prediction. In *CIKM*, 2020.

Pengjie Ren, Zhumin Chen, Jing Li, Zhaochun Ren, Jun Ma, and Maarten De Rijke. Repeatnet: A repeat aware neural recommendation machine for session-based recommendation. In *AAAI*, 2019.

Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*, 2019.

Qiaoyu Tan, Jianwei Zhang, Jiangchao Yao, Ninghao Liu, Jingren Zhou, Hongxia Yang, and Xia Hu. Sparse-interest network for sequential recommendation. In *WSDM*, 2021.

Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *WSDM*, 2018.

Pengfei Wang, Jiafeng Guo, Yanyan Lan, Jun Xu, Shengxian Wan, and Xueqi Cheng. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*, 2015.

Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing. Recurrent recommender networks. In *WSDM*, 2017.

Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. Sse-pt: Sequential recommendation via personalized transformer. In *Fourteenth ACM Conference on Recommender Systems*, pp. 328–337, 2020.

Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *AAAI*, 2019.

Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. Graph contextualized self-attention network for session-based recommendation. In *IJCAI*, 2019.

An Yan, Shuo Cheng, Wang-Cheng Kang, Mengting Wan, and Julian McAuley. Cosrec: 2d convolutional neural networks for sequential recommendation. In *CIKM*, 2019.

Zeenat F Zaidi. Gender differences in human brain: a review. *The open anatomy journal*, 2010.

Wayne Xin Zhao, Shanlei Mu, Yupeng Hou, Zihan Lin, Yushuo Chen, Xingyu Pan, Kaiyuan Li, Yujie Lu, Hui Wang, Changxin Tian, et al. Recbole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In *CIKM*, 2021.

Guorui Zhou, Xiaoqiang Zhu, Chenru Song, Ying Fan, Han Zhu, Xiao Ma, Yanghui Yan, Junqi Jin, Han Li, and Kun Gai. Deep interest network for click-through rate prediction. In *KDD*, 2018.

Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, 2020.

Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced mlp is all you need for sequential recommendation. In *WWW*, 2022.

Yiyi Zhou, Tianhe Ren, Chaoyang Zhu, Xiaoshuai Sun, Jianzhuang Liu, Xinghao Ding, Mingliang Xu, and Rongrong Ji. Trar: Routing the attention spans in transformer for visual question answering. In *ICCV*, 2021.

A DESCRIPTIONS OF THE DATASETS

Intra-domain datasets. Here are descriptions of the seven datasets: (1) **Netflix**: Netflix dataset is a large-scale movie rating dataset released by Netflix. (2) **MSD**: The Million Song Dataset (MSD) is a large-scale, metadata-rich and open-source dataset on Kaggle. (3) **Taobao**: Taobao dataset (Tan et al., 2021) contains user behaviors in Taobao's recommender system. In experiments, we only use the click behaviors. (4) **Yelp** (Asghar, 2016): Yelp is a dataset for business recommendation. We only use the transaction records after January 1st, 2019. (5) **Tmall**: Tmall contains users' shopping logs on Tmall online shopping platform, which is from the IJCAI-15 competition. (6) **Steam** (Kang & McAuley, 2018): Steam dataset is collected from a large online video game distribution platform. This dataset includes 2,567,538 users, 15,474 games and 7,793,069 English reviews from October 2010 to January 2018. (7) **MovieLens1M**: this is a widely used benchmark dataset for evaluating collaborative filtering algorithms. The version we use is MovieLens-1M, which includes 1 million user ratings.

Table 4: Statistics of the intra-domain datasets.

Dataset	Users	Items	Actions
Netflix	463,435	17,769	57,000,000
MSD	571,355	41,140	34,000,000
Taobao	987,994	4,162,024	100,150,807
Yelp	30,431	20,033	316,354
MovieLens1M	6,040	3,416	1,000,000
Tmall	66,909	37,367	427,797
Steam	334,730	13,047	3,700,000

Cross-domain datasets. We choose five categories from Amazon review datasets (Hou et al., 2022): Grocery and Gourmet Food, Home and Kitchen, CDs and Vinyl, Kindle Store, and Movies and TV, as the source domain datasets for pre-training. For the target datasets, we choose another five categories from Amazon review datasets (Hou et al., 2022): Prime Pantry, Industrial and Scientific, Musical Instruments, Arts, Crafts and Sewing, and Office Products, as target domain datasets to evaluate the proposed approach under the cross-domain setting. The detail statistics are shown in Table 5.

Table 5: Statistics of the cross-domain datasets.

Source Dataset	Users	Items	Actions	Target Dataset	Users	Items	Actions
Food	115,349	39,670	1,027,4137	Scientific	8,442	4,385	59,427
CDs	94,010	64,439	1,118,563	Pantry	13,101	4,898	126,9626
Kindle	138,436	98,111	2,204,596	Instruments	24,962	9,964	208,926
Movies	281,700	59.203	3,226,731	Arts	45,486	21,019	395,150
Home	731,913	185,552	6,451,926	Office	87,436	25,986	684,837

B FURTHER ABLATION STUDY

Number of heads and maximum sequence length. In the left column of Table 6, we adjust the number of heads for RETR on Yelp. We find that the performance first increases rapidly with the growth of the head number and achieves the best performance at $h = 4$. We perform a similar grid search on other datasets. In the right column of Table 6, we adjust the maximum sequence length N for RETR on Yelp. As shown in Table 6, we find that the performance of our RETR first increases rapidly with the growth of the block number and achieves the best performance at $N = 100$. We perform a similar grid search on other datasets.

Table 6: Ablation study of the head number (**Left**) and the maximum sequence length for RETR (**Right**). Experiments are conducted on the Yelp Dataset.

Model (# h)	MRR	Model (# maximum sequence length)	MRR
RETR ($h = 1$)	0.4317	RETR ($N = 25$)	0.4237
RETR ($h = 2$)	0.4345	RETR ($N = 50$)	0.4294
RETR ($h = 4$)	**0.4378**	**RETR** ($N = 100$)	**0.4378**
RETR ($h = 8$)	0.4369	RETR ($N = 200$)	0.4362

Effectiveness of each model component and number of blocks. In the left column of Table 7, we analyze the efficacy of each component in RETR on the Yelp dataset and have the following observations. First, we remove the pathway router module and randomly choose whether it can be maintained or dropped for each input behavior token. Removing the pathway router decreases the prediction performance a lot (MRR: $0.4354 \rightarrow 0.3887$), showing the necessity of learning behavior pathway effectively based on a data-dependent module. Second, discarding the hierarchical update strategy for the behavior pathway also decreases the prediction performance, suggesting that this strategy is crucial for RETR to get a more accurate behavior pathway. In the right column of Table 7, we adjust the number of blocks for RETR on Yelp. We find that the performance first increases rapidly with the growth of the block number and achieves the best performance at $L = 2$. We perform a similar grid search on other datasets.

Table 7: Ablation study of (**Left**) the effectiveness of each model component and (**Right**) the number of blocks for each RETR block. Experiments are conducted on the Yelp Dataset.

Model	MRR	Model (# number of blocks)	MRR
RETR	**0.4378**	RETR ($L = 1$)	0.4197
RETR w/o Pathway Router	0.3887	**RETR** ($L = 2$)	**0.4378**
RETR w/o hierarchical update	0.4234	RETR ($L = 3$)	0.4342
SASRec	0.3927	RETR ($L = 4$)	0.4340

Evaluation on efficiency. The efficiency is compared between SASRec (Kang & McAuley, 2018), SINE (Tan et al., 2021) and SMRec (Chen et al., 2021) on the Yelp dataset. The computation cost is measured with gigabit floating-point operations (GFLOPs) on the self-attention module with position encoding. Meanwhile, the model scale measured with parameters is also presented. As shown in Table 8, our RETR has almost the same number of parameters or GFLOPs, compared with SASRec, indicating that our pathway router is a light-weight module. Our pathway attention does not bring more costs. It's worth noticing that the parameter scales and GLOPs of other competing transformers (apart from SASRec) are larger than RETR, but our RETR achieves higher performance. This result shows that our RETR is more efficient and effective than other competing attention-based models.

Table 8: Ablation study of (**Left**) the effectiveness of different temperatures; Comparison Parameters and GFLOPs (**Right**). All ablation study experiments are conducted on the Yelp Dataset.

Model	Parameters (M)	GFLOPs	MRR
RETR	5.0	9.6	**0.4378**
SASRec (2018)	5.0	9.6	0.3927
SINE (2021)	5.1	9.7	0.4011
SMRec (2021)	5.2	9.9	0.4023

Why we need to use this pathway router in sequential recommendation? Previous sequential recommendation methods have proved that the recommender can be benefited a lot from the user's historical behaviors, even though the behavior sequence may be short. However, when meeting with the short behavior sequence, the recommender still needs to deal with various behavior pathways and can be overwhelmed by the trivial behaviors. As shown in Figure 4, we show the last 10 behaviors from a random user in the Steam dataset. Further, in Figure 4, we show that the state-of-the-art MLP-based model, FMLP-Rec (Zhou et al., 2022), is still overwhelmed by the old drifted behaviors (simulation games). To avoid the recommender being overwhelmed by the trivial behaviors, we design the pathway router to capture the pivotal behavior pathway that explains the user's preferences, whenever the behavior sequence is short or long. It's crucial to develop the pathway router to capture the behavior pathway for making precise recommendations. The pathway router is important for sequential recommendation models to avoid being overwhelmed by trivial user behaviors.

Replace the proposed pathway-based method with other sparse attention methods. We replace the proposed pathway-based method with two sparse attention methods: LogSparse (Li et al., 2019) and sparse attention (Child et al., 2019). As shown in Table 9, our RETR using the pathway attention

Figure 3: Illustrations of how RETR, SASRec and FMLP-Rec (Zhou et al., 2022) differs on utilizing the historical behaviors of a random user in Steam Dataset. We provide the visualizations of behavior heatmaps for RETR, SASRec and FMLP-Rec of a random user in Steam dataset.

remarkably outperforms other competing methods with two sparse attention methods on Tmall. These experiment results show that sparse attention methods cannot capture the exact behavior pathway and show worse performance than RETR.

Table 9: Ablation study of sparse attention methods on the Tmall dataset.

Model	NDCG@10	HR@10	MRR
RETR w/ LogSparse	0.4923	0.6015	0.4735
RETR w/ Sparse attention	0.4871	0.5873	0.4620
RETR	**0.6197**	**0.7214**	**0.5903**

Quantitative results on whether the proposed model effectively captures a useful pathway. We give quantitive results to validate that our RETR can effectively capture various behavior pathways. Specifically, we evaluate our RETR using a subset of sequences derived from the obtained behavior pathway on Tmall. Technically, we first train RETR on Tmall. For each user, we take the captured behavior pathway from our RETR as the inputs to retrain a RETR rather than using the whole user's behaviors. As shown in Table 10, we find that using the behavior pathway as the inputs can achieve comparable results as the original RETR which uses complete user behaviors. It provides the evidence that our RETR can aptly capture the useful pathway for each user.

Table 10: Quantitative results on the Tmall dataset.

Model	NDCG@10	HR@10	MRR
RETR w/ pathway inputs	0.6112	0.7142	0.5831
RETR	0.6197	0.7214	0.5903
SASRec	0.5049	0.6275	0.4804
SASRec w/ pathway inputs	0.5778	0.6812	0.5425
SASRec w/ off-pathway inputs	0.4383	0.5697	0.4215

What's the essential difference between RETR and sequential model with attention mechanism? Our RETR has two essential differences compared with sequential model with attention mechanism: (1) Our RETR designs the pathway router to capture the behavior pathway, while the other sequential models have not considered it before. As shown in Figure 4, the previous self-attention mechanism mainly focuses on the recent behaviors, and cannot capture the accurate behavior pathway. Only our RETR can capture the precise behavior pathway. (2) The pathway attention for RETR is the cross-attention between the pathway behavior tokens and off-pathway tokens. Our pathway cross-attention mechanism can avoid the trivial interaction between the off-pathway tokens.

Why use the cross-attention mechanism? We choose the cross-attention mechanism for three main reasons: (1) The cross-attention mechanism can force the pathway attention to attend to the behavior pathway; (2) It can ensure that the contextual information from off-pathway behavior tokens can be captured, using the original input behavior tokens as the key and value; (3) Our pathway cross-attention mechanism avoids the trivial interaction between the off-pathway tokens, while the previous

self-attention mechanism for sequential models can be overwhelmed by the trivial information in the off-pathway behavior tokens. To verify our explanation, we further conduct evaluation experiments on Tmall. Specifically, we train RETR on Tmall, and then use the trained RETR on Tmall to capture the behavior pathway for each user in Tmall. We use the pathway behaviors and off-pathway behaviors as the inputs to train SASRec respectively. As shown in Table 10, we can see that SASRec achieves better performance using the behavior pathway as the inputs compared with the original SASRec using the whole user's behavior as the inputs. On the contrary, the off-pathway inputs hurt SASRec's performance seriously. Finally, our RETR achieves the best performance, indicating that the pathway-offpathway cross-attention is more effective than the pathway self-attention.

C VISUAL EXAMPLES

Setups. We also provide qualitative visualizations for our RETR, and SASRec Kang & McAuley (2018). Technically, we use the GradCAM Selvaraju et al. (2017) to generate behavior heatmaps of the output of the last layer in each model. Three random examples of users' historical behaviors in the Steam dataset are shown in sequential order through subplots (a)–(c) in Figure **??**. We provide attention heatmaps of each example at the last ten time steps. We can observe three main behavior pathway characteristics corresponding to three behavior sequences respectively: (a) *Casual behavior pathway:* RPG games are randomly clicked by the user, while the user has a continuing interest in RPGs. (b) *Correlated behavior pathway:* The user has recently been interested in indie games. (c) *Drifted behavior pathway:* The user has recently been interested in simulation games but chooses an indie game at last.

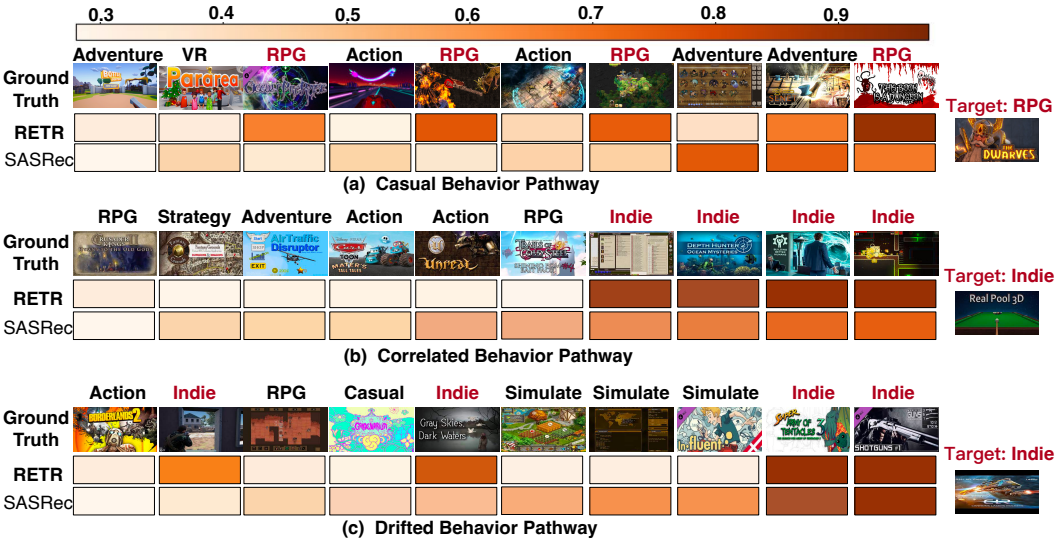

Figure 4: Visualizations of behavior heatmaps for RETR and SASRec of three random users in Steam dataset. They are corresponding to casual, correlated and drifted behavior pathways respectively.

Visualization results. We elaborate the three representative categories of behavior pathway in recommender systems with model-learned attention heatmaps. (1) *Casual behavior pathway:* As shown in Figure 4(a), the RGB game is randomly clicked at casual times. Our RETR can capture all the RPG casual behavior pathways, while the SASRec focuses on the incorrect recent adventure games. The SASRec cannot capture the early clicked RPG game. This phenomenon proves that our RETR can deal with the casual behavior pathway effectively. (2) *Correlated behavior pathway:* For the correlated behavior pathway, we also provide an example which is shown in Figure 4(b). The indie game is clicked many times recently, leading to the final decision to an indie game. Our RETR can effectively capture the correlated behavior pathway. However, the SASRec provides higher attention scores on the recent RPG games. On the contrary, our RETR pays no attention to these wrong results, showing that it has a greater ability to cope with the correlated behavior pathway. (3)

Drifted behavior pathway: As shown in Figure 4(c). The user was initially interested in the indie game, but suddenly became interested in simulation games recently and chose an indie game at last. Our RETR captures the drifted behavior pathway for the indie game and has not concentrated on the old drifted pathway – simulation games, while the SASRec is affected by the trivial behaviors of simulation games. These visualization results strongly show that our RETR can capture various behavior pathways dynamically for each user.

D ROLLING PREDICTION ON HELD-OUT TEST USER SEQUENCES

Setup. We build a rolling prediction setup on held-out test user sequences. Specifically, we split the users into train/val/test sets with the ratio $8:1:1$. For training users, we use the last record for prediction, while all remaining behaviors are for training. For validation and test users, the last 5 records of each user are for the rolling prediction, and we tune hyper-parameters using the validation users. We report the average results of the last 5 records on test users for the model that achieves the best results on the validation users.

Table 11: Rolling prediction performance comparison to state-of-the-art models under the intra-domain setting on Tmall. Rec-denoiser (Chen et al., 2022) uses BertRec as the backbone.

Meric	BERT4Rec	SASRec	SMRec	S3-Rec	SINE	TGSRec	LightSAN	Rec-denoiser	Locker	Jodie	TGN	RETR
HR@10	0.2437	0.2415	0.2639	0.2751	0.2682	0.2617	0.2639	0.2872	0.2830	0.2622	0.2628	**0.3051**
NDCG@10	0.3873	0.3776	0.4039	0.4115	0.4073	0.4036	0.4082	0.4176	0.4144	0.3971	0.3905	**0.4578**
MRR	0.2584	0.2553	0.2671	0.2716	0.2683	0.2675	0.2690	0.4176	0.4144	0.2635	0.2649	**0.2985**

Rolling prediction results. We conduct rolling prediction experiments on the Tmall dataset. Note that we evaluate the models on the users that are not present at training time. This setup is more complicated than the case where test and training users overlap. As shown in Table 11, our RETR achieves state-of-the-art performance among all baselines. Note that Rec-denoiser and Locker also achieve competitive performance with the help of learned sparse attention, but our RETR outperforms these two methods substantially. These results show that our RETR develops an effective pathway-attention mechanism with strong capacity to make better rolling predictions on held-out test user sequences.

E COMPARISON WITH TRAR

TRAR (Zhou et al., 2021) uses an efficient sparse attention mechanism for visual question answering by routing the attention span, while our RETR is designed explicitly for sequential recommendation. Besides the incomparable application domains, the architecture of RETR is substantially different from TRAR, especially in three main aspects:

- TRAR routes the choice of adjacency masks, automatically selecting the attention span. The sparse attention mechanism in TRAR is effective for visual question answering, but it is not designed specifically to be effective for sequential recommendation. As shown in Table 12, we replace the pathway attention in RETR with the sparse attention in TRAR and conduct experiments on Tmall, MovieLens1M, and Yelp. We find that unsurprisingly, TRAR achieves worse performance than SASRec, while our RETR outperforms TRAR and SASRec considerably. These results are sufficient evidence that capturing the behavior pathway is crucial for sequential recommendation and our pathway attention mechanism is more effective for sequential recommendation.
- RETR is aimed at capturing the behavior pathway. It develops the pathway router to capture the behavior pathway dynamically for each user. Technically, the pathway router can embed global information from the whole behavior sequence and keep the original information from the input representation via the residual connection, while the pathway controller in TRAR only captures the global information via attention pooling. As shown in Table 13, we replace the pathway router in RETR with the pathway controller in TRAR. Our RETR

performs worse using the pathway controller. These results show that the pathway controller cannot capture the behavior pathway effectively and achieve worse performance.

- RETR develops a hierarchical update strategy for router as described in Equ 6. As shown in Table 7, discarding the hierarchical update procedure will lead to worse performance, indicating that this strategy is crucial to capture the behavior pathway effectively from the intrinsically hierarchical representations. However, the pathway controller in TRAR cannot update the choice of mask hierarchically.

Table 12: Quantitative results on the Tmall, MovieLens1M, and Yelp dataset.

Dataset	Model	NDCG@10	HR@10	MRR
Tmall	TRAR (Zhou et al., 2021)	0.5007	0.6139	0.4659
	SASRec (Kang & McAuley, 2018)	0.5049	0.6275	0.4804
	RETR	**0.6197**	**0.7214**	**0.5903**
MovieLens1M	TRAR (Zhou et al., 2021)	0.5759	0.8119	0.5384
	SASRec (Kang & McAuley, 2018)	0.5936	0.8233	0.5573
	RETR	**0.6397**	**0.8513**	**0.5952**
Yelp	TRAR (Zhou et al., 2021)	0.4439	0.7186	0.3883
	SASRec (Kang & McAuley, 2018)	0.4642	0.7373	0.3927
	RETR	**0.5169**	**0.7775**	**0.4378**

Table 13: Ablation on the pathway attention mechanism on the Tmall dataset.

Model	NDCG@10	HR@10	MRR
RETR w/ pathway controller	0.5613	0.6526	0.5408
RETR	**0.6197**	**0.7214**	**0.5903**

F Ablation Study for Adaptive Gumbel-Softmax

The temperature parameter τ is a crucial hyperparameter for the standard Gumbel-Softmax. A fixed temperature cannot be adaptive across different datasets or users. It is widely-known to be uneasy to tune the temperature parameter, in that a lower value may lead to high variances in gradients and a higher value may lead to over-smoothing probabilities. To mitigate these technical issues, we propose a novel adaptive Gumbel-Softmax mechanism to eliminate the need of temperature tuning, which can produce token-specific weights automatically adjusted to varying behaviors of each user.

Table 14: Quantitative results on the Tmall dataset. "-" indicates failure case of model training.

Model	NDCG@10	HR@10	MRR
RETR w/ standard Gumbel-Softmax ($\tau = 2$)	0.6049	0.7084	0.5787
RETR w/ standard Gumbel-Softmax ($\tau = 1$)	0.6084	0.7105	0.5803
RETR w/ standard Gumbel-Softmax ($\tau =$ **0.8**)	0.6103	0.7138	0.5822
RETR w/ standard Gumbel-Softmax ($\tau = 0.6$)	0.6095	0.7129	0.5817
RETR w/ standard Gumbel-Softmax ($\tau = 0.4$)	-	-	-
RETR w/ standard Gumbel-Softmax ($\tau = 0.2$)	-	-	-
RETR w/ Gumbel-Softmax (temperature annealing)	0.6093	0.7133	0.5814
RETR w/ adaptive Gumbel-Softmax	**0.6197**	**0.7214**	**0.5903**

As shown in Tables 14 and 15, we find that RETR with the standard Gumbel-Softmax achieves the highest performance in different datasets at different temperatures ($\tau = 0.8$ for Tmall and $\tau = 0.6$ for Yelp). These results show that a fixed temperature is not adaptive across diverse datasets. However, lower temperatures ($\tau = 0.2, 0.4$) cause the failure of model training for RETR due to the high variance in gradients. Higher temperatures ($\tau = 1, 2$) may perform worse because of the

Table 15: Quantitative results on the Yelp dataset. "-" indicates failure case of model training.

Model	NDCG@10	HR@10	MRR
RETR w/ standard Gumbel-Softmax ($\tau = 2$)	0.5113	0.7695	0.4328
RETR w/ standard Gumbel-Softmax ($\tau = 1$)	0.5124	0.7719	0.4345
RETR w/ standard Gumbel-Softmax ($\tau = 0.8$)	0.5136	0.7730	0.4354
RETR w/ standard Gumbel-Softmax ($\tau =$ **0.6**)	0.5152	0.7749	0.4360
RETR w/ standard Gumbel-Softmax ($\tau = 0.4$)	-	-	-
RETR w/ standard Gumbel-Softmax ($\tau = 0.2$)	-	-	-
RETR w/ Gumbel-Softmax (temperature annealing)	0.5134	0.7727	0.4350
RETR w/ adaptive Gumbel-Softmax	**0.5169**	**0.7775**	**0.4378**

over-smoothing probabilities in Gumbel-Softmax. It proves difficult to tune the temperature for the standard Gumbel-Softmax. Previous work like TRAR (Zhou et al., 2021) develops a schedule that starts with a high temperature and gradually anneals it to a small but non-zero value. This schedule can make the training more stable, but it cannot achieve the best performance adaptively across different datasets.

In contrast, RETR with the proposed adaptive Gumbel-Softmax featured by the token-specific weight mechanism achieves the best performance compared with the standard Gumbel-Softmax under different temperatures. These results indicate that the adaptive Gumbel-Softmax is much more effective in capturing the behavior pathway and can be dynamically adapted to different datasets without tuning the temperature. This adaptive mechanism can simultaneously overcome the over-smoothing phenomenon and dynamically avoid the high variances in gradients.

G Visualization for Hierarchical Update

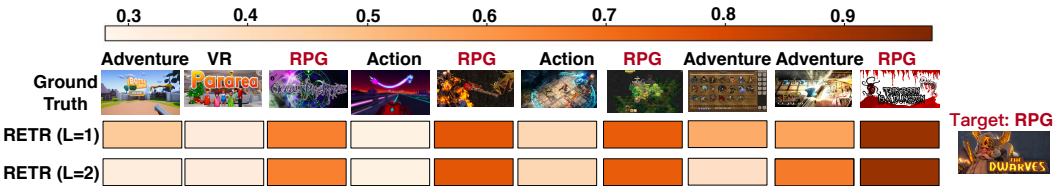

Figure 5: Visualization for hierarchical update of RETR. We provide the visualization of behavior heat maps of RETR for different blocks of a random user in Steam dataset.

We further explore how the behavior pathway updates hierarchically via the visualization in Figure 5. Technically, we use the GradCAM Selvaraju et al. (2017) to generate behavior heat maps of the output of the first and last layer in RETR, respectively. As shown in the Figure 5, we observe that our RETR can update the captured behavior pathway hierarchically, dropping the drifted behavior pathway *Adventure* in the last layer (**row 2**) from the first layer (**row 1**). This visualization shows that our RETR can determine the practical behavior pathway via the hierarchical update procedure.