Page-Wise Personalized Recommendations in an Industrial e-Commerce Setting

Liying Zheng[1], Yingji Pan[1] and Yuri M. Brovman[1]

[1]*Ebay, Inc.*

Abstract

Providing personalized recommendations based on the dynamic sequential behaviors of users plays an important role in e-commerce platforms since it can considerably improve a user's shopping experience. Previous works apply a unified model pipeline to build recommender systems, without considering the differentiated behavior patterns and intrinsic shopping tendencies on different pages of an e-commerce website. In this paper, we focus on generating a personalized recommender system optimized to both the View Item Page and Homepage by elaborately designing strategies for data formulation and model structure. Our proposed model (PWPRec) consists of a causal transformer encoder together with a fusion module designed for different pages, built on the basis of the classical two-tower structure. This provides the capability to capture a balanced long-short interest or diverse multiple interests of a user during their shopping journey across multiple types of pages. We have conducted experiments both on in-house datasets as well as public datasets to validate the effectiveness of our model, all showing significant improvements on Recall@k metrics compared to the commonly applied sequential models of recent years. Additionally, we built a state-of-the-art deep learning based retrieval system utilizing real-time KNN search as well as near real-time (NRT) user embedding updates to reduce the recommendation delay to a few seconds. Our online A/B test results show a big advantage compared to the previous GRU-based sequential model in production, with a 38.5% increase in purchased items due to model improvements and 107% increase in purchased items due to the engineering innovations.

Keywords

sequential recommendation, multi-interest, attention network, transformer encoder

1. Introduction

Recommender systems play a fundamental role in e-commerce marketplaces, offering personalized recommendation products based on a user's specific interests which will largely improve a user's shopping experience. In this work, we focus on "user context based" recommender systems that generate recommendations using a user's historical interactions as the main context. There are several different landing pages which display recommendations to the user on an e-commerce platform and in this work we focus on two pages: View Item Page (VIP) and Homepage (HP). For the VIP, users usually have a specific shopping mission when they navigate a detailed item page, thus they tend to spend more time comparing similar products and trying to find the most appropriate one. Figure 1 depicts an example of a VIP with a user context based recommendations module based on users recent views. For the HP, usually at the beginning of a user's shopping session, users tend to wander through the whole page without a specific shopping mission. They could be attracted by discounted or hot-sale products, or diversified categories they have been consistently interested in, thus we plan to design a new module generating multiple item sets capturing user's multiple interests.

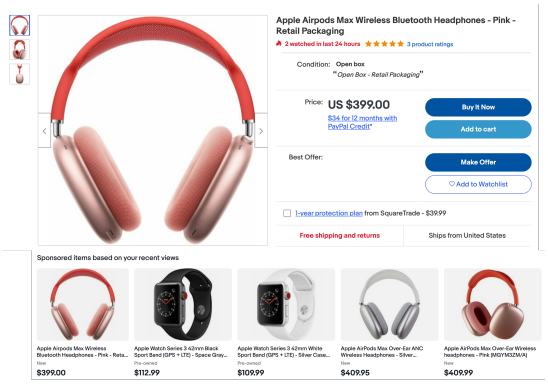

Figure 1: Screenshot of eBay View Item Page recommendation module with one item set of personalized items.

Incorporating different user shopping behavior patterns on the VIP and HP mentioned above, we have developed a page-wise personalized recommendation model (PWPRec) in order to capture a user's different shopping goals and interests. Specifically, here are the main contributions of the paper:

1. We present a page-wise deep learning model that considers multiple shopping contexts in an industrial setting.

ORSUM@ACM RecSys 2022: 5th Workshop on Online Recommender Systems and User Modeling, jointly with the 16th ACM Conference on Recommender Systems, September 23rd, 2022, Seattle, WA, USA

liyzheng@ebay.com (L. Zheng); yingpan@ebay.com (Y. Pan); ybrovman@ebay.com (Y. M. Brovman)

2. We develop a novel model architecture by combining a causal transformer encoder with a long-short or multi-interest fusion module in order to generate user embedding(s).
3. We deploy our recommender system to our production industrial setting building a state-of-the-art deep learning based retrieval system in the process.

The paper is organized in the following sections. Section 2 covers related approaches in literature to our method. The main model architecture is discussed in Section 3. The datasets and sampling strategies used for our offline experiments are then discussed in Section 4. An overview of our production engineering architecture as well as A/B tests is presented in Section 5. We conclude our work in Section 6.

2. Related Works

Adding personalization in recommender systems is a well studied problem both in academia and in industrial applications. Recently, deep neural networks have been adopted in personalized recommendations, with the ability to build a more generalized model by capturing complex content-based features, which can also serve well in cold-start situations or volatile situations. To generate personalized recommendations, the sequential behaviors of users are effectively exploited by applying different sequential encoder networks. Many works apply Recurrent Neural Networks (RNN) for sequential recommendation and obtain promising results, among those Hidasi et al. [1] proposed an GRU-based network to model the sequential behaviors of users, and adopts the last output as the user embedding (known as GRU4Rec), Hidasi and Karatzoglou [2] proposed a top-k gain ranking loss function used in RNNs for session-based recommendations, Li et al. [3] was also based on RNN network, but proposed a way to balance a user's local interest and global interest (known as NARM). Besides RNN network, the recently well-known self-attention mechanism [4] for sequential modeling has also been commonly applied in recommendations, Kang and McAuley [5] proposed a self-attention based network to capture the sequential behaviors of users, and the encoded value of last item in the sequence is regarded as the ultimate user vector (known as SASRec), Sun et al. [6] adopted the Bidirectional Encoder Representations from Transformers which trained a bidirectional model to predict the masked items in the sequence. Also, there are some methods based on graph neural networks proposed for sequential recommendation, Wu et al. [7] modeled session sequences as graph-structured data to take item transitions into account, Xu et al. [8] proposed a graph contextualized self-attention model for session-based recommendation.

Most prior work generates a single embedding to represent a user, this is reasonable for recommendation pages or placements with specified target items. But in some occasions such as Homepage recommendations we may like to provide users with more diversified set of recommendations reflecting the multiple interests of a user. It can be observed that the problem of how to capture the multiple interests of a user a popular topic in recent years. Weston et al. [9] introduced a highly scalable method for learning a non-linear latent factorization to model the multiple interests of a user. Li et al. [10] proposed a multi-interest extractor layer based on capsule network with the dynamic routing mechanism. Cen et al. [11] explored a self-attentive method for multi-interest extraction, and utilized an aggregation module to balance accuracy and diversity.

In terms of engineering system architecture, there are several works which describe large scale embedding based retrieval systems. Pal et al. [12] describes an industrial embeddings based retrieval system which uses HNSW model [13] for the approximate nearest neighbor (ANN) component. There are several production systems that utilize a two tower model for search and retrieval, including in the social media space [14] as well as in e-commerce space [15, 16]. We will now discuss the details of our model architecture.

3. The PWPRec Model

In our application scenario, we find that a user's distribution of recently viewed items differs for different pages. For VIP, users usually have a definite shopping purpose and are thus more likely to click on items related to their most recently viewed items. However, for Homepage, users have a less focused shopping purpose and may click on different categories of items. Thus we build our model in consideration of different pages and placements, which can better capture and understand the different behavior intentions of users.

3.1. Page-Wise Sequential Behavior Analysis

Before introducing our detailed model, we first present an analysis of a user's shopping behavior as a function of time. In our sequential modeling approach, every training example is composed of a positive target clicked item, several negative items a user did not click on in the impression, and a series of user historical items.

We build up a histogram of the overlap between the category of the target item and historical items for all users in the dataset. Figure 2 demonstrates the difference between the VIP and HP distributions. The horizontal axis represents the number of hours between the target

clicked item and a historical item, while the vertical axis represents the category overlap between the target and historical items. It can be seen from the graph that for the View Item Page (orange in Figure 2), about 80% of users are also viewing the same category in the first hour before, 5% are viewing same category in the second hour before, indicating that users are focused on the same category of most recent items. While for Homepage, the curve is more gradual with only 30% overlap in the first hour before, indicating that on homepage users show interest in categories they interacted with in a longer period, thus target item category may correlate to more diverse historical categories.

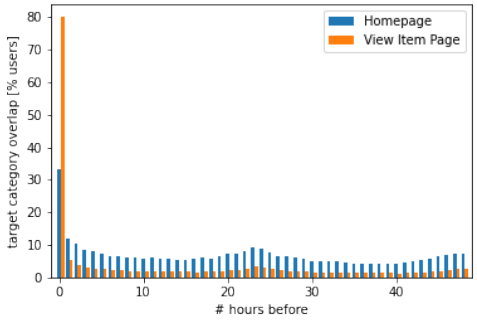

Figure 2: User historical category overlap histogram on different pages.

Based on the above analysis, we decide to adopt different data formulation strategies and different model structures for different pages.

- For the **View Item Page**, considering users usually have specific shopping missions and less interests in other categories, we organize training data in a *"session-based way"* with most recent past behaviors in a shorter period. The ultimate output will be a singular item set showing the user's most recent interest.
- For the **Homepage**, users may show interest in a diverse set of categories that they interacted with, even several days before. In this case, we organize the training data in a *"user-based way"* incorporating more days and more past behaviors. The ultimate output will be multiple item sets showing the user's multiple interests through the long shopping journey.

3.2. Model structure

Our proposed approach for personalized recommendations is based on a two-tower deep learning model structure to generate user embedding(s) and item embeddings at the same time. The overall architecture of PWPRec is shown in Figure 3. Following our previous work [17] we keep using the same structure for the item tower, and focus on optimizing the user tower. The original user tower adopted the recurrent neural network as the base encoder of user's historical events and an average fusion strategy to generate the final user embedding. Here we optimize the user encoder network by two architectural modules: 1) a sequential encoder to better capture the ordered historical events, and 2) a fusion network to better adapt to pages with different historical item distributions. In the next sections, we will delve deep into these modules in detail.

3.3. Causal Transformer Encoder

The transformer network and self-attention mechanism described in [4] are widely applied in NLP related tasks, and achieved state-of-the-art performance. Here we adopt the idea of transformer and self-attention to function as the sequential encoder, and we have made some modifications in order to capture the order information which is of vital importance to recommendation scenarios.

3.3.1. Relative Positional Embedding

We first tried the fixed position embedding originated from the vanilla self-attention in [4], it did not work well. This may due to fixed embedding cannot capture the relative positional information well, which is quite essential in an e-commerce setting. In our case, a learnable embedding with relative position value works the best. The relative position value is calculated as below:

$$pos(item_i) = T - i \tag{1}$$

where T is the position of the target item and i is the position of items prior to the target item. The relative position is then encoded into embedding $\mathbf{P}_{emb}$, and the final input to the transformer encoder is calculated as:

$$\mathbf{IN}_{vector} = \mathbf{ITEM}_{emb} + \mathbf{P}_{emb} \tag{2}$$

where $\mathbf{ITEM}_{emb}$ is the original item embedding, and the final input vector $\mathbf{IN}_{vector}$ is a vector addition of the item embedding with the positional embedding $\mathbf{P}_{emb}$.

3.3.2. Causal Attention Mask

The vanilla transformer encoder attends to any positions in a sequence by self-attention and multi-head mechanism, with each head output being formulated as:

$$\begin{aligned} \mathbf{head}_i &= Attention(\mathbf{QW}_\mathbf{i}^\mathbf{Q}, \mathbf{KW}_\mathbf{i}^\mathbf{K}, \mathbf{VW}_\mathbf{i}^\mathbf{V}) \\ Attention(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= softmax(\frac{\mathbf{QK}^\mathbf{T}}{\sqrt{d}})\mathbf{V} \end{aligned} \tag{3}$$

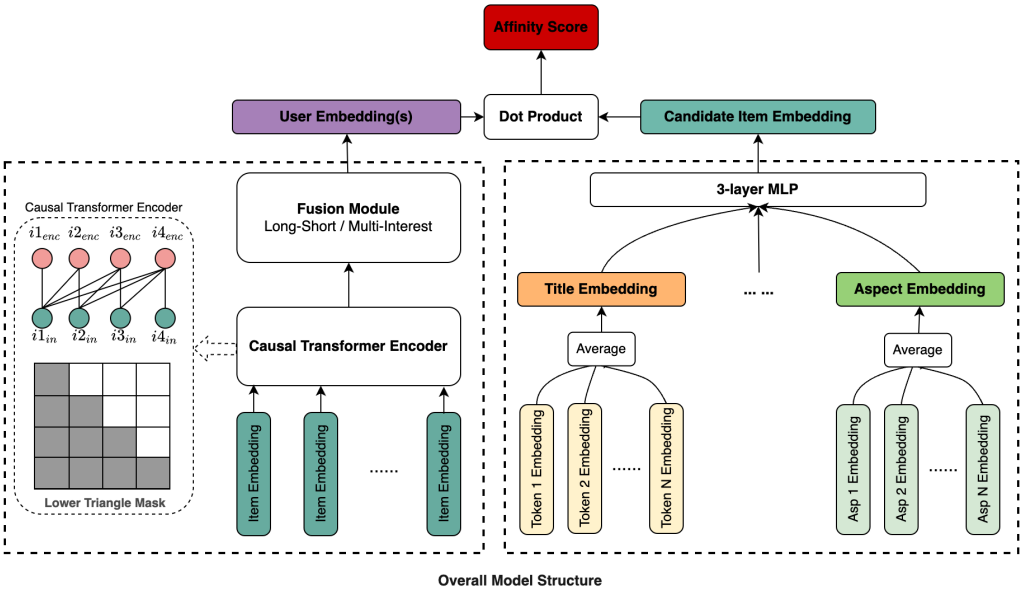

Figure 3: Two tower model architecture for user embedding(s) and item embedding. The causal transformer encoder is explained in the left part. The fusion module serving different pages will be explained in subsequent subsections, with long-short fusion generating a comprehensive user embedding or multi-interest fusion generating multiple user embeddings.

where $\mathbf{Q}$, $\mathbf{K}$, $\mathbf{V}$ are the packed matrices of queries, keys and values, d is the dimension of queries and keys and $\mathbf{W_i^Q}$, $\mathbf{W_i^K}$, $\mathbf{W_i^V}$ are the parameter matrices, as described in self-attention mechanism [4].

For the sequential recommendation scenario, a causal mask [18] needs to be performed to guarantee that post-clicked items cannot be seen when predicting previous items, otherwise this may lead to data leakage. Therefore, we apply a lower triangle attention mask matrix (as shown in the left part of Figure 3) to guarantee the causality between items, and in this way the self-attention can be formulated as :

$$\begin{aligned} Attention(\mathbf{Q},\mathbf{K},\mathbf{V}) &= softmax(Mask(\frac{\mathbf{QK^T}}{\sqrt{d}}))\mathbf{V} \\ Mask &= Tril(Ones(\mathbf{M} \in \mathscr{R}^{L*L})), \end{aligned} \tag{4}$$

where L is the sequence length, Ones represents all-ones matrix and Tril represents the lower triangular part of matrix, and the mask operation will fill future values with $-inf$.

3.4. Fusion Module

In our industrial recommendation scenario, we design two fusion networks to handle the different recommendation targets, one for generating a comprehensive single interest applied for the VIP, named as **Long-Short Fusion**; the other for generating multiple interests for the HP, named as **Multi-Interest Fusion**.

3.4.1. Long-Short Fusion Strategy

For generating one single interest, we would like to adopt a network architecture which combines a user's short-term and long-term interests. The short-term takes the last position output of the transformer encoder, indicating the most recent preferences; while the long-term takes outputs of all the positions into consideration, indicating their global preferences. We involve the attention mechanism to calculate a weighted average of all the outputs to form a long-term interest. which can be interpreted as:

$$\begin{aligned} \mathbf{U}_{long} &= \sum_{k=1}^{L} Attention(\mathbf{ik}_{enc}, \mathbf{iL}_{enc}) * \mathbf{ik}_{enc} \\ Attention(\mathbf{ik}_{enc}, \mathbf{iL}_{enc}) &= \mathbf{v}^T\sigma(\mathbf{A_1} \times \mathbf{ik}_{enc} + \mathbf{A_2} \times \mathbf{iL}_{enc}) \end{aligned} \tag{5}$$

where the attention function is additive attention [19], $\mathbf{iL}_{enc}$ represents the last position item embedding, $\mathbf{ik}_{enc}$ represents item embeddings of all the positions, $\mathbf{A_1}$ transforms $\mathbf{ik}_{enc}$ into a latent space, $\mathbf{A_2}$ plays the same role for $\mathbf{iL}_{enc}$, and σ is the sigmoid function.

After learning a long-term and a short-term embedding, the last important step is to integrate them appropriately. Here we chose the gated way to learn contribution coefficients of long-term and short-term embeddings, which is illustrated in Figure 4, and can be calculated as:

$$\begin{aligned} \mathbf{U}_{emb} &= (1 - gate) \times \mathbf{U}_{long} + gate \times \mathbf{U}_{short} \\ gate &= \sigma(\mathbf{G_1} \times \mathbf{U}_{short} + \mathbf{G_2} \times \mathbf{U}_{long}) \end{aligned} \tag{6}$$

where $\mathbf{U}_{short} = \mathbf{iL}_{enc}$, $\mathbf{U}_{long}$ is in Equation (5) and σ is sigmoid activation function. In the gate equation, $\mathbf{G_1}$ and $\mathbf{G_2}$ both transform $\mathbf{U}_{short}$ and $\mathbf{U}_{long}$ into latent spaces, respectively.

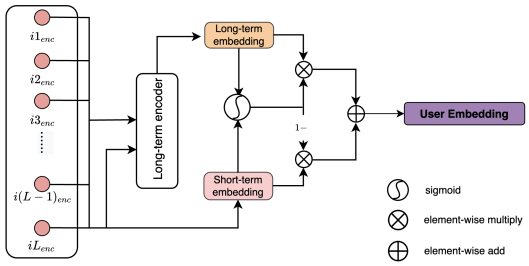

Figure 4: Long-short Fusion Module.

3.4.2. Multi-Interest Fusion Strategy

The multi-interest fusion module is utilized to capture multiple interests from a user's shopping journey. A multi-head self-attentive network is applied to transform the sequential item encoders of a user into multiple user representations. We follow the self-attentive method originated from Lin et al. [20], which was then applied in a recommendation system in Cen et al. [11] to function as the multi-interest extractor. In our work, we found that when this multi-interest fusion module was combined with the transformer sequential encoder, the model performance was significantly improved.

The multi-interest fusion network is illustrated in Figure 5. Suppose we have a sequence of items $i_1, i_2, ..., i_L$, and after the causal transformer encoder, the items can be represented as $\mathbf{I} = \{\mathbf{i1_{enc}}, \mathbf{i2_{enc}}, ..., \mathbf{iL_{enc}}\}$, with sequence length L. A multi-head self-attentive layer is adopted to calculate the attention weights $\mathbf{A}$ of input item sequences, with each head representing one interest. The mutiple user embeddings $\mathbf{U}$ for the current user can be calculated as :

$$\begin{aligned} \mathbf{A} &= softmax((tanh(\mathbf{IW_1})\mathbf{W_2})^T) \\ \mathbf{U} &= \mathbf{AI} \end{aligned} \tag{7}$$

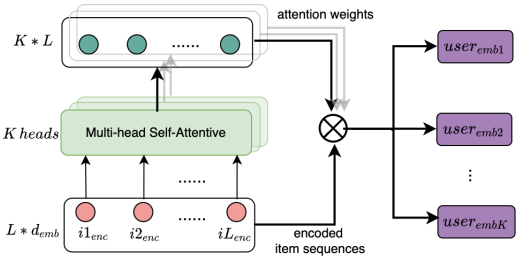

Figure 5: Multi-Interest Fusion Module.

where $\mathbf{I} \in \mathscr{R}^{L*d_{emb}}$ is the sequential items embeddings, $\mathbf{W_1} \in \mathscr{R}^{d_{emb}*d_{hidden}}$ is a trainable parameter matrix which transforms input item encoded vectors from dimension d_{emb} to d_{hidden} (usually hidden is several times larger than emb to increase model capacity), $\mathbf{W_2} \in \mathscr{R}^{d_{hidden}*K}$ is another trainable parameter matrix which maps d_{hidden} to the number of embeddings K (K is the number of user interests to be generated). The attention weights matrix is $\mathbf{A} \in \mathscr{R}^{K*L}$. The final multiple user embeddings is $\mathbf{U} \in \mathscr{R}^{K*d_{emb}}$.

4. Offline Datasets & Experiments

In this section, we describe the dataset we utilized to train and validate our PWPRec model. We have adopted different data formulation strategies for the View Item Page and Homepage respectively. We also conducted experiments on both our eBay dataset and public dataset to validate the effectiveness of our model.

4.1. Dataset for View Item Page

For the View Item Page, users tended to click more items related to recently viewed items, thus we organize the data in a *session-based way*. Here we choose two session-based datasets for our experiments, one is collected from our eBay in-house data, the other is the public YooChoose dataset [21] which is also commonly adopted by research papers.

- **eBay (session-based) dataset**
  This dataset is derived from our real world eBay production traffic containing view item page events within a session. All items are enriched with necessary metadata like titles, aspects and categories as well.
- **YooChoose dataset**
  This dataset is provided by YooChoose in RecSys Challenge 2015, with each session encapsulating the click events that a user performed from a

retailer. In this dataset, only item id and category is provided to generate an item embedding.

In order to better validate the effectiveness of sequential encoders, we filter out very short sessions with sequence length of less than 4. The data statistics of the two datasets are shown in Table 1.

Statistics	eBay(session-based)	YooChoose
# of training sessions	18 million	1.9 million
# of validation sessions	2 million	470k
# of items	72 million	53k
Average sequence length	15	8

Table 1
View Item Page Data Statistics.

4.2. Dataset for Homepage

For the Homepage, we organize the data in a *user-based way* within a longer time window and thus much longer user's sequential length is obtained. Here we choose two user-based datasets in our experiments, one is collected from our eBay in-house data, the other is the public Taobao dataset [22].

- **eBay (user-based) dataset**
  This dataset is also derived from our real world eBay production traffic containing clicked items on Homepage as the target label, and all the items that a user have viewed within 30 days before the clicked item as the sequential historical events.
- **Taobao dataset**
  This dataset contains the sequential behavior of users collected from Taobao, which consists of 1 million users shopping behaviors within 10 days. We follow the same training/validation data splitting methods as in [11].

The statistics of the above datasets are shown in Table 2. Also, we build up a histogram of the overlap between the category of the target item and historical items for the Taobao dataset in Figure 6, which shows a gradual curve more similar to the eBay HP than VIP. So we adopted the Taobao dataset to validate the effectiveness of the multiple interests model targeted for Homepage.

4.3. Model Training & Validation

For model training and validation, different negative sampling strategies and loss calculations are adopted for View Item Page and Homepage.

Statistics	eBay(user-based)	Taobao
# of training users	40 million	0.8 million
# of validation users	2 million	97k
Average sequence length	102	87

Table 2
Homepage Data Statistics.

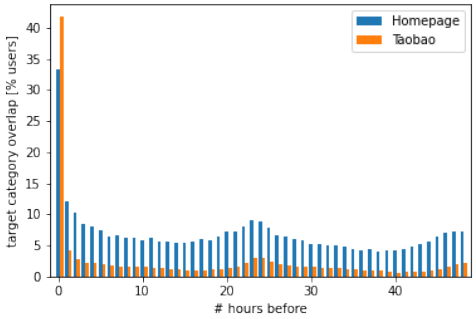

Figure 6: User historical category overlap histogram on Taobao and eBay Homepage

4.3.1. View Item Page

View item page data samples are grouped by session, as the session lengths are usually shorter than the *user-based way*, we adopt global negative sampling to choose negative items in a larger candidate pool. For the training phase, each training data sample has one positive item and 10 negative items; while for validation phase, we select 1000 negative items to make the evaluation more generalized to the whole candidate item set. We use cross entropy loss to train the model, whose target is to maximize the softmax probability of the positive item:

$$P(pos|U) = \frac{e^{\gamma(\mathbf{v}_{pos},\mathbf{v}_u)}}{\sum_{i\in pos\cup neg} e^{\gamma(\mathbf{v}_i,\mathbf{v}_u)}}, \qquad Loss = -logP(pos|U) \tag{8}$$

where $\mathbf{v}_u \in \mathbb{R}^d$ is a d-dimensional vector for the embedding of user U, $\mathbf{v}_{pos} \in \mathbb{R}^d$ is a d-dimensional vector for the embedding of positive item, γ is the affinity function between user and item (we adopt the inner product result as the affinity score), and $pos \cup neg$ is the union set of the target positive and sampled negative items.

4.3.2. Homepage

As for the Homepage, data samples are grouped in the *user-based way* with longer sequential behaviors within a 30 days time window, batch negative sampling is adopted

to select 1000 samples both in training and validation phase. Here the loss calculation logic for training and validation process is tackled differently for accelerating the convergence of multiple user embeddings model structure.

- **Training phase.** As we have the positive item for the target label information, we can use the positive item embedding to choose one final user embedding from multiple embeddings as the one to calculate the training loss.

$$\mathbf{v}_u = \mathbf{V}_u[argmax(\mathbf{V}_u \mathbf{v}_{pos}^T)] \tag{9}$$

  where $\mathbf{v}_u \in \mathbb{R}^d$ is the final user embedding we select to calculate the loss in equation (8), $\mathbf{V}_u$ is the multiple embeddings genenerated for the user, and $\mathbf{v}_{pos} \in \mathbb{R}^d$ is the positive item embedding.
- **Validation phase.** Different from the training phase, label information like positive item cannot be used in metrics calculation, otherwise this would result in label leakage. Here we applied a simplified trick to fastern the procedure, which is selecting one user embedding having the maximum summarized affinity score with the candidate item set as the final user embedding for loss and metrics calculation.

$$\mathbf{v}_u = \mathbf{V}_u[argmax(\sum_{i \in items} \mathbf{V}_u \mathbf{v}_i^T)] \tag{10}$$

  where $\mathbf{v}_u \in \mathbb{R}^d$ is the final user embedding we select to calculate model metrics, $\mathbf{V}_u$ is the multiple embeddings genenerated for the user, and $\mathbf{v}_i \in \mathbb{R}^d$ is the item embedding contained in the candidate item set for validation.

4.4. Offline Experimental Results

The primary evaluation metric we use is Recall@k at several $k = 1, 5, 10, 20$. For P impressions, the metric is defined as:

$$Recall@k = \frac{1}{P} \sum_{i=1}^{P} \frac{\text{\# relevant items @ k}}{\text{\# total relevant items}} \tag{11}$$

We then explain the experimental results conducted on different pages with different datasets.

4.4.1. View Item Page Experiments

The experimental results can be found in Table 3. We select three other models for comparison: GRU4Rec [1], NARM [3], SASRec [5]. Our baseline model , described in [17], is very similar to GRU4Rec but has the following enhancements: 1) changes the loss function to cross entropy with inverse temperature, 2) adopt a attention-based weighted sum mechanism to generate the ultimate embedding. We call our baseline model **GRU4Rec-Enhanced**. For our model PWPRec proposed in this paper, we add the suffix (LS) to represent Long-Short fusion strategy.

We see from Table 3 that on both of the datasets we have depicted in Section 4.1, our model **PWPRec(LS)** achieved the best performance. Our model gains 10+% increase on Recall@1 compared to the baseline model GRU4Rec-Enhanced. We notice that on the YooChoose dataset, the recall values are lower, possibly because of the smaller size of training set as well as the lack of item features, like titles or aspects. However, our model has a bigger advantage on this dataset even for recall with larger Ks, which implies that even in a situation where less features are available, our model can perform better and have better generalization capabilities.

4.4.2. Homepage Experiments

The experimental results can be found in Table 4. For our model PWPRec proposed in this paper, we add the suffix (MI) to represent Multi-Interest fusion strategy. Here we chose the model ComiRec described in [11] as the baseline, and also the well-known multi-interest model MIND [10] for comparison. We see from Table 4 that our model **PWPRec(MI)**, with the transformer encoder and multi-interest fusion layer, outperforms the other two models by 20+% on Recall@1 metrics, and a high 10+% increase for recall with larger K. Similar to previous experiments, our model gains a more significant improvement on the public datasets which lacks item feature information.

5. Production Engineering Architecture

Details of the continuous improvements we have made to the engineering architecture of this system can be seen in our eBay Tech Blog post [23]. Most of the modeling innovations described in Sections 3.2 and 4 were A/B tested against the baseline version of the system described in our previous work [17]. In our previous approach, most of the model calculations were performed offline with daily batch jobs to generate user/item embeddings and perform KNN for every user embedding searching over the space of item embeddings. There is a clear disadvantage, namely the delay between offline calculation of predictions (performed daily) and displaying the recommendations to the user could lead to stale outdated recommendations and a degraded user experience. To overcome this issue and reduce this delay to a few seconds, we built a state-of-the-art deep learning based

Table 3
Experimental results for View Item Page

Dataset	Model	Recall@1	Recall@5	Recall@10	Recall@20
	GRU4Rec-Enhanced	0.4212	0.6341	0.7038	0.7643
	NARM	0.4249(+0.88%)	0.6378(+0.58%)	0.7101(+0.90%)	0.7743(+1.31%)
	SASRec	0.4745(+12.65%)	0.6509(+2.65%)	0.7084(+0.65%)	0.767(+0.35%)
	ours-PWPRec(LS)	**0.4761(+13.03%)**	**0.6611(+4.26%)**	**0.7239(+2.86%)**	**0.7777(+1.75%)**
	GRU4Rec-Enhanced	0.1222	0.127	0.1372	0.2116
	NARM	0.1186(-2.95%)	0.123(-3.15%)	0.1331(-2.99%)	0.2079(-1.75%)
	SASRec	0.1366(+11.78%)	0.1429(+12.52%)	0.1526(+11.22%)	0.2281(+7.80%)
	ours-PWPRec(LS)	**0.1407(+15.14%)**	**0.1459(+14.88%)**	**0.1568(+14.29%)**	**0.2306(+8.98%)**

Table 4
Experimental results for Homepage

Dataset	Model	Recall@1	Recall@5	Recall@10	Recall@20
	ComiRec	0.4835	0.7139	0.7522	0.7916
	MIND	0.3832(-20.74%)	0.5863(-17.87%)	0.6231(-17.16%)	0.687(-13.21%)
	ours-PWPRec(MI)	**0.5898(+21.99%)**	**0.8221(15.16%)**	**0.8512(13.16%)**	**0.8738(+10.38%)**
	ComiRec	0.1098	0.1970	0.2461	0.3007
	MIND	0.0862(-21.49%)	0.1533(-22.18%)	0.1926(-21.74%)	0.2514(-16.39%)
	ours-PWPRec(MI)	**0.1373(+25.05%)**	**0.2419(+22.79%)**	**0.2914(+18.41%)**	**0.3427(+13.97%)**

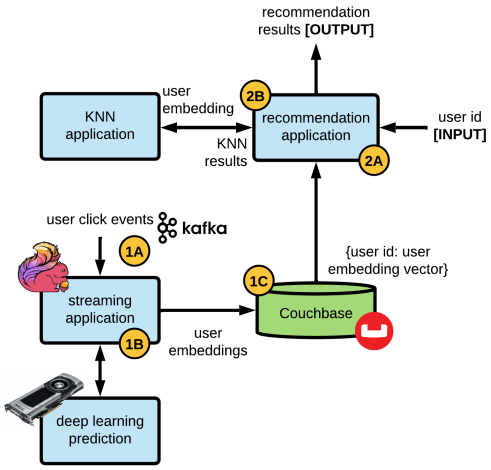

Figure 7: Production engineering architecture featuring real-time KNN search as well as NRT user embedding updates.

retrieval system utilizing real-time KNN search as well as near real-time (NRT) user embedding updates, details displayed in Figure 7.

To enable fast real-time KNN search for vector embeddings, we have built an in-house KNN microservice based on HNSW [13] method, where a user embedding is sent as an input, an ANN search is performed in the item embedding space, and then item recommendations are returned. In order to generate a user embedding in real time, we capture user click activity on the site using Apache Kafka message events and process them using a Apache Flink application. The events are enriched with metadata and processed through a deep learning model prediction microservice to generate the actual embedding vector, which is subsequently stored in Couchbase. Putting all of these together, we generate the full NRT flow for personalized recommendations:

1. *Step 1A* - A user clicks on previous View Item Pages and these click events are collected using the Kafka messaging platform.
2. *Step 1B* - The Flink application aggregates the last several events and generates a user embedding by calling the model prediction microservice.
3. *Step 1C* - The user embedding is stored in Couchbase with {key:value} = {user id: user embedding vector}.
4. *Step 2A* - As the user lands on a View Item Page, the backend recommendations application gets the user embedding from Couchbase.
5. *Step 2B* - A request is made to the KNN microservice, personalized recommendations are returned and rendered back to the user.

As a result of this system architecture, the delay between generating personalized recommendations based on the user's session data and displaying them is reduced to a few seconds. This system is in production serving high volume traffic to a diverse set of users. Next we will

discuss our online A/B testing results which support our offline model evaluations.

5.1. Online Evaluation

In order to understand how our models perform online, we deployed them to serve real world users and production traffic. We compare results respectively for View Item Page and Homepage, and the NRT architecture as well.

5.1.1. View Item Page A/B Test

We performed A/B testing on the View Item Page on the desktop platform comparing our PWPRec(LS) model to our previous baseline model [17] named as GRU4Rec-Enhanced. Our model outperformed the previous baseline with a 38.53% increase on purchases. This implies our model with transformer encoder better captures the sequential behavior of a user, and the Long-Short fusion mechanism is also a good choice to automatically balance the weights captured from long interests and short interests, better than the previous weighted sum fusion way.

5.1.2. NRT A/B Test

It was interesting to see the impact of the reduced delay between recommendation generation and serving on the operational metrics of the system as we deployed the NRT engineering architecture to production. The purchases were improved by 107% compared to the previous offline system. This makes sense from a user experience perspective, as the user shopping journey evolves in real time, the model embedding is updated in real time, the recommendation relevance quality is improved, and operations metrics are better. [17]

5.1.3. Homepage Multi-Interest User Scrapes

We are in the process of serving our multi-interest model online for A/B test. However, we wanted to share some multi-interest user recommendations from production environment to demonstrate the performance of the model. We can see the generated recommendations in Figure 8 which depicts 3 distinct sets of recommended items based on the multiple interests of a user from related browsing history. Based on the user's past viewed items shown on the first line, our model captures three interests for this user, which accurately reveals the intrinsically diverse set of interests of a user throughout their shopping journey.

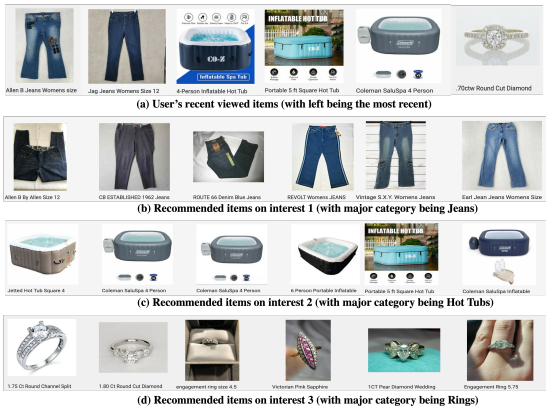

Figure 8: A user scrape of production environment, (a) shows the user's historical interacted items, (b)(c)(d) shows the three interests we have captured for this user, with (b) representing the first interest on Jeans, (c) representing the second interest on Hot Tubs, (d) representing the third interest on Rings.

6. Summary and future work

In this paper, we presented an approach for generating personalized recommendations by considering different user behavior patterns on different pages in an industrial e-commerce setting. Different strategies on data formulation and fusion layer adoption have been elaborately designed to capture a user's sequential behavior on the View Item Page and the Homepage. The overall structure is based on a two-tower model aiming to learn embeddings of items and users in a shared vector space. To model the user's sequential behavior, we adopt a causal transformer encoder together with Long-Short fusion or Multi-Interest fusion determined by page settings on the user tower side. This approach captures the user's long-short interests and multiple interests well. In order to verify the effectiveness of our model, we have conducted experiments on our in-house datasets and commonly adopted public datasets as well. All experiments showed significant improvements over the baseline approaches in comparison. Furthermore, a personalized recommender system with NRT engineering architecture has been launched to production and is now serving recommendations at scale to eBay buyers. This system reacts quickly based on instant user interactions and generates large improvements in the buyer shopping experience. Online A/B tests have also been conducted for our proposed model as well as the NRT architecture, which also show increases on downstream business metrics, such as purchases.

We are actively working to enhance the performance and extend the application scenario of our model as well

as engineering system. One direction of future work is to incorporate more rich user features (e.g. demographic features like buyer age and behavioral features like purchase quantity) as well as item features (e.g. item price, popularity). Another direction is to add a deep learning ranking model after the multiple recommendation sets have been retrieved, in order to further optimize for operational metrics, like engagement or conversion. Last but not least, besides the current View Item Page and Homepage we are serving, we plan to extend our personalized recommneder system to more scenarios like infinite feed as well as checkout success placements, in order to give users more personalized and diverse choice with NRT experiences in their shopping journey.

7. Acknowledgements

We wanted to thank the generous support of Sathish Veeraraghavan, Bing Zhou, Arman Uygur, Marshall Wu, Sriganesh Madhvanath, Santosh Shahane, Leonard Dahlmann, Menghan Wang, and Jeff Kahn for the help with the production system as well as review comments during the manuscript preparation.

References

[1] B. Hidasi, A. Karatzoglou, L. Baltrunas, D. Tikk, Session-based recommendations with recurrent neural networks, arXiv:1511.06939 (2016).

[2] B. Hidasi, A. Karatzoglou, Recurrent neural networks with top-k gains for session-based recommendations, arXiv:1706.03847 (2017).

[3] J. Li, P. Ren, Z. Chen, Z. Ren, J. Ma, Neural atentive session-based recommendation, arXiv:1711.04725 (2017).

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, arXiv preprint arXiv:2102.06156 (2017).

[5] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, arXiv:1808.09781 (2018).

[6] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, P. Jiang, Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer, arXiv:1904.06690 (2019).

[7] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, T. Tan, Session-based recommendation with graph neural networks, arXiv:1811.00855 (2019).

[8] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, X. Zhou, Graph contextualized self-attention network for session-based recommendation (2019). URL: https://www.ijcai.org/proceedings/2019/0547.pdf.

[9] J. Weston, R. J. Weiss, H. Yee, Nonlinear latent factorization by embedding multiple user interests (2013). URL: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/41535.pdf.

[10] C. Li, Z. Liu, M. Wu, Y. Xu, P. Huang, H. Zhao, G. Kang, Q. Chen, W. Li, D. L. Lee, Multi-interest network with dynamic routing for recommendation at tmall, arXiv:1904.08030 (2019).

[11] Y. Cen, J. Zhang, X. Zou, C. Zhou, H. Yang, J. Tang, Controllable multi-interest framework for recommendation, arXiv:2005.09347 (2020).

[12] A. Pal, C. Eksombatchai, Y. Zhou, B. Zhao, C. Rosenberg, J. Leskovec, Pinnersage: Multi-modal user embedding framework for recommendations at pinterest, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2311–2320.

[13] Y. A. Malkov, D. A. Yashunin, Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs, IEEE transactions on pattern analysis and machine intelligence 42 (2018) 824–836.

[14] J.-T. Huang, A. Sharma, S. Sun, L. Xia, D. Zhang, P. Pronin, J. Padmanabhan, G. Ottaviano, L. Yang, Embedding-based retrieval in facebook search, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2020, pp. 2553–2561.

[15] H. Zhang, S. Wang, K. Zhang, Z. Tang, Y. Jiang, Y. Xiao, W. Yan, W.-Y. Yang, Towards personalized and semantic retrieval: An end-to-end solution for e-commerce search via embedding learning, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 2407–2416.

[16] S. Li, F. Lv, T. Jin, G. Lin, K. Yang, X. Zeng, X.-M. Wu, Q. Ma, Embedding-based product retrieval in taobao search, in: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 3181–3189.

[17] T. Wang, Y. M. Brovman, S. Madhvanath, Personalized embedding-based e-commerce recommendations at ebay, arXiv preprint arXiv:2102.06156 (2021).

[18] W.-C. Kang, J. McAuley, Self-attentive sequential recommendation, arXiv preprint arXiv:1808.09781 (2018).

[19] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate., CoRR, abs/1409.0473, 2014 (2020).

[20] Z. Lin, M. Feng, C. Nogueira dos Santos, M. Yu, B. Xiang, B. Zhou, Y. Bengio, A structured self-attentive sentence embedding, arXiv preprint arXiv:1703.03130 (2017).

[21] Yoochoose, Recsys challenge 2015, 2015. URL: https://recsys.acm.org/recsys15/challenge/.
[22] Alimama, User behavior data from taobao for recommendation, 2018. URL: https://tianchi.aliyun.com/dataset/dataDetail?dataId=649&userId=1&lang=en-us.
[23] Y. M. Brovman, Building a deep learning based retrieval system for personalized recommendations, 2022. URL: https://tech.ebayinc.com/engineering/building-a-deep-learning-based-retrieval-system-for-personalized-recommendations/.