Learning a Fine-Grained Review-based Transformer Model for Personalized Product Search

Keping Bi
University of Massachusetts Amherst
Amherst, MA, USA
kbi@cs.umass.edu

Qingyao Ai
University of Utah
Salt Lake City, UT, USA
aiqy@cs.utah.edu

W. Bruce Croft
University of Massachusetts Amherst
Amherst, MA, USA
croft@cs.umass.edu

ABSTRACT

Product search has been a crucial entry point to serve people shopping online. Most existing personalized product models follow the paradigm of representing and matching user intents and items in the semantic space, where finer-grained matching is totally discarded and the ranking of an item cannot be explained further than just user/item level similarity. In addition, while some models in existing studies have created dynamic user representations based on search context, their representations for items are static across all search sessions. This makes every piece of information about the item always equally important in representing the item during matching with various user intents. Aware of the above limitations, we propose a review-based transformer model (RTM) for personalized product search, which encodes the sequence of query, user reviews, and item reviews with a transformer architecture. RTM conducts review-level matching between the user and item, where each review has a dynamic effect according to the context in the sequence. This makes it possible to identify useful reviews to explain the scoring. Experimental results show that RTM significantly outperforms state-of-the-art personalized product search baselines.

KEYWORDS

Product Search; Personalization; Transformer Models; Fine-grained Matching; Review-based Matching

ACM Reference Format:
Keping Bi, Qingyao Ai, and W. Bruce Croft. 2021. Learning a Fine-Grained Review-based Transformer Model for Personalized Product Search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462911

1 INTRODUCTION

In product search, users' purchase behaviors usually depend on their individual preferences in addition to product relevance. Aware of this point, recent studies [3, 4, 6, 18, 41, 45] have explored to incorporate personalization in the product retrieval models and produced significant improvements in the search quality. A typical paradigm of existing personalized product search models is to represent the user intents and items explicitly with embedding vectors and match them in the latent space with dot product or cosine similarity to yield the item score [2–4, 6, 7, 18]. Under this paradigm, user's search intents are usually represented by a function of the query vector and the user vector, which can be a convex combination [3], a simple addition [2], several neural layers [18], or a transformer encoder [6]. Item representations are usually learned by predicting words in the item reviews [2–4, 6, 7, 18], and user vectors, which represents the user's personalized preferences, are represented similarly [3] or as a weighted combination of items vectors with attention mechanisms [2, 6, 18].

Despite its popularity, the existing product search paradigm has several limitations in practice. First, in the existing product search framework, the item scoring is based on matching at the user/item level [2, 3, 6, 18] instead of the finer-grained level, e.g., user/item reviews. Thus how a specific user preference mentioned in the user reviews matches an item property indicated in the item reviews could not be captured sufficiently. While a top-retrieved item is close to the user intent or some of the users' historical purchases in the latent space, why the model considers them close is not clear. Second, despite their efforts on constructing dynamic user representations under different contexts [2, 6, 18], existing product search studies always represent items statically regardless of the context [2–4, 6, 7, 18]. During the representation learning, all the associated reviews are considered to have equal importance. However, the same aspect of an item may play different roles in representing the item when matching with various user preferences towards the aspect. For example, for a user who has tooth whitening needs, reviews that comment on the toothpaste's whitening function should be more important to represent the item than the long-lasting breath freshening property. However, in existing product search models, a review complaining about the shipment and package handling of the toothpaste may be considered equally important as other reviews since the item representations are built as static vectors without considering user intents.

Given these limitations, in this paper, we propose to match user intents and items at the level of finer-grained information (e.g., their associated reviews) instead of explicitly representing them with static vectors. Specifically, we score an item based on the sequence of the query, user reviews, and item reviews with a transformer [37] architecture, where every unit, i.e., query or user/item review, could interact with each other during matching. We refer to our model as the review-based transformer model, abbreviated as RTM.

RTM has a couple of advantages over existing Transformer-based [6] or other neural model based product search approaches [2, 3, 18].

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07…$15.00
https://doi.org/10.1145/3404835.3462911

First, RTM conducts user-item matching at the review level so that the reason an item is ranked to the top can be explained by some useful reviews that draw more attention from other units in the sequence. Second, in RTM, the importance of each user and item review during matching is dynamically adapted in different sequences, where both users and items carry dynamic information under different context. When encoded by a multi-layer RTM, the review representation is dynamically changed according to its interactions with other units in the sequence. Also, RTM represents user and item based only on their reviews without the need for their identifiers, so it can easily generalize to the users and items that have associated reviews but have not appeared in the training set. Last but not least, RTM can conduct more flexible personalization than most existing personalized product search models [3, 4, 7, 18, 43]. Personalization in RTM can vary from no to full effect depending on the contexts since the user reviews could have zero accumulative attention weights and so does the query. Our experimental results confirm our model's advantages by showing that RTM significantly outperforms the state-of-art baselines. To better understand the model and explain the search results, we also analyze the influence of different settings on RTM and conduct case studies to show how RTM identifies important information during matching.

Our contributions in this paper can be summarized as follows: 1) we propose a review-based transformer model (RTM) for personalized product search that is superior to existing models in terms of finer-grained matching, dynamic user/item representation, generalization ability, and the influence of personalization; 2) our experimental results show that our RTM achieves significantly better performance than state-of-the-art personalized product search techniques; 3) we analyze the model property and conduct case studies to understand the model behaviors better.

2 RELATED WORK

Our work is closely related to product search, personalized web search, and transformer-based retrieval models.

Product Search. Since product information is more structured, earlier research uses facets such as brands, prices, and categories for product search [22, 36]. However, these approaches cannot handle free-form user queries. To support search based on keyword queries, Duan and Zhai [15], Duan et al. [16] extended language-model-based techniques by assuming that queries are generated from the mixture of one language model of the background corpus and the other one of products conditioned on their specifications. Word mismatch problems still exist in these approaches. Van Gysel et al. [35] noticed this problem and proposed representing and matching queries and products in the latent semantic space.

Aware of the importance of personalization in product search, Ai et al. [3] proposed a hierarchical embedding model where they use a convex combination of the query and user vector to predict purchased items. Guo et al. [18] represent long and short-term user preferences with an attention mechanism applied to users' recent purchases and their global vectors. Recently, from the analysis of commercial search logs, Ai et al. [2] observed that personalization does not always have a positive effect. They further proposed a zero-attention model (ZAM) that can control the influence of personalization. However, the maximal effect personalization can have is equal to the query. Bi et al. [6] found this limitation and proposed a transformer model to encode the query and historically purchased items where personalization can have none to full effect.

There are also studies on other aspects such as popularity, other information sources (e.g., images), diversity, and labels for training in product search. Long et al. [23] incorporated popularity with relevance for product ranking. Di et al. [13] and Guo et al. [19] investigated on using images as a complementary signal. Ai et al. [4] proposed an explainable product search model with dynamic relations such as brand, category, also-viewed, also-bought, etc. Efforts have also been made to improve result diversity to satisfy different user intents behind the same query [27, 42]. In terms of training signals, Wu et al. [39] jointly modeled clicks and purchases in a learning-to-rank framework and [20] compared the effect of different labels such as click-rate, add-to-cart ratios, and order rates. More recently, Zhang et al. [44] integrated the graph-based feature with neural retrieval models for product search. Xiao et al. [41] studied personalized product search under streaming scenarios. Ahuja et al. [1] learned language-agnostic representations for queries and items that can support search with multiple languages. There are also studies on interactive product search such as in a multi-page search setting [8] and in conversational systems [7, 43].

Most existing work either focuses on non-personalized product search or conducts personalized product search with static user/item representations. In contrast, we propose an adaptive personalization model for product search and conduct item scoring in a novel paradigm, i.e., dynamic matching user intents and items at the review level instead of explicitly represent them in the semantic space and match them directly.

Personalized Web Search. Personalization has also been studied in the context of Web search, where results are customized to each user in order to maximize their satisfaction [17]. Existing approaches usually infer users' personal needs from their locations, search histories, clicked documents, etc., and then re-rank results accordingly [5, 9, 10, 31, 33]. While users could behave differently for the same query [14, 38], personalization does not always benefit search quality. Based on the analysis of user behavior patterns in the large-scale logs on Live Search, Teevan et al. [34] observed that the effectiveness of personalization in Web search depends on multiple factors such as result entropy, result quality, search tasks, and so on.

We focus on product search in this paper, where personalization is more appealing than in Web search. While relevance is usually the primary criterion in Web search, user purchases depend on both item relevance and user preferences in product search.

Transformer-based Retrieval Models. After the pre-trained contextual language models, i.e., BERT [12], grounded on the transformer architecture, achieved compelling performance on a wide range of natural language processing tasks, more studies have explored leveraging BERT in information retrieval tasks as well. Nogueira and Cho [26] show BERT's effectiveness on passage ranking, and Dai and Callan [11] demonstrate that BERT can leverage language structures better and enhance retrieval performance on queries in natural languages. Wu et al. [40] proposed a passage cumulative gain model that applies a sequential encoding layer on top of the BERT output of a query-passage pair to score a document. Qu et al. [28] refine the original BERT model with an additional

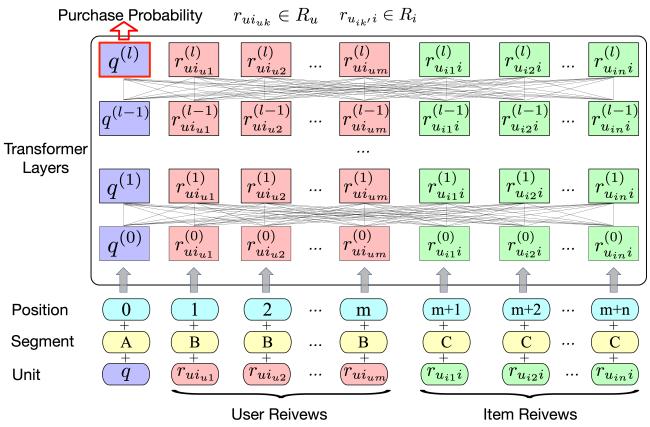


Figure 1: Our Review-based Transformer Model (RTM).

attention layer on each question-utterance pair to attentively select useful history to identify the answer span in conversational question answering.

Our model is based on transformers instead of BERT. In other words, we leverage the transformer architecture to dynamically match the query-user pair and the item at the review level but we do not represent queries and reviews with pre-trained BERT.

3 REVIEW-BASED TRANSFORMER MODEL

This section introduces each component of our review-based transformer model (RTM) and how RTM conducts personalized item scoring. Then we show how RTM is optimized and provide interpretations of RTM from the perspective of model design.

3.1 Personalized Item Scoring

In contrast to previous studies that have explicit representations for both users and items [2, 3, 18] or just for items [2, 6, 18], our model does not represent the user or item with a single vector. Instead, we consider the user and item's historical reviews as the basic unit carrying their information and learn to score the item given the query-user pair using the interactions between their basic information units. In this way, the matching of a query-user pair and an item can be conducted at a finer grain and could potentially capture more connections between the user and the item, which we will illustrate later in Section 3.4 in detail.

Let q be a query submitted by a user u and i be an item in S_i, which is the collection of all the items. $R_u = (r_{ui_{u1}}, r_{ui_{u2}}, \cdots, r_{ui_{um}})$ and $R_i = (r_{u_{i1}i}, r_{u_{i2}i}, \cdots, r_{u_{in}i})$ denote the sequence of m and n reviews associated with u and i respectively in a chronological order, where $r_{ui_{uk}} (1 \le k \le m)$ is the review u wrote for her k-th purchased item i_{uk} and $r_{u_{ik'}i} (1 \le k' \le n)$ is the review associated with $u_{ik'}$, which is the k'-th user who purchased i. m and n are the length of R_u and R_i respectively. As shown in Figure 1, we feed the sequence of (q, R_u, R_i) to an l–layer transformer encoder to let the query and user's purchase history interact with the item's associated reviews.

Input Embeddings. Inspired by the architecture of BERT [12], our model also has three types of embeddings associated with each input unit (query or review) to the transformer encoder. They are the *unit embedding* that is computed from the words in the unit, which we will introduce later in Section 3.2, the *position embeddings* [37] that indicate the position of a unit in the sequence, and the *segment embeddings* that differentiate whether the unit is the query q, a review from u, or a review about i, denoted as A, B, and C respectively, as shown in Figure 1.

Since a recent review from the user may indicate her current intention better than her long-ago reviews and an item's recent review may reveal the item's current properties better than an old review about the item, position embeddings could be beneficial by indicating the temporal order of each review. While the time of reviews could be significant in some categories, it is also possible that the information of items such as the music on a CD is static, and user preferences behind particular purchase needs stay similar for a long time. Therefore, we make the position embeddings an optional choice in our model.

The segment embedding of a unit is also optional in RTM since the model could infer which segment the current unit belongs to with their position embeddings. The reviews about i always have later positions than reviews of u, and q is always at position 0. When position embeddings are not necessary in some cases, segment embeddings are still needed to differentiate the input units. We will show the effects of position and segment embeddings on our model later in Section 5.2.

Formally, the input vector of each unit is the sum of the unit embedding and its associated optional position embedding and segment embedding:

$$\begin{aligned} \mathbf{q}^{(0)} &= \mathbf{q} + \mathcal{I}_{pos} E(0) + \mathcal{I}_{seg} E(A) \\ \mathbf{r_{ui_{uk}}}^{(0)} &= \mathbf{r_{ui_{uk}}} + \mathcal{I}_{pos} E(k) + \mathcal{I}_{seg} E(B), 1 \le k \le m \\ \mathbf{r_{u_{ik'}i}}^{(0)} &= \mathbf{r_{u_{ik'}i}} + \mathcal{I}_{pos} E(m+k') + \mathcal{I}_{seg} E(C), 1 \le k' \le n \end{aligned} \tag{1}$$

where $E(\cdot)$ is the embedding of position $0, 1, \cdots, m+n$ or segment A, B, C; $\mathcal{I}_{pos} \in \{0, 1\}$ and $\mathcal{I}_{seg} \in \{0, 1\}$ indicate whether position and segment embeddings are used in the computation; $\mathbf{q}$, $\mathbf{r_{ui_{uk}}}$, and $\mathbf{r_{u_{ik'}i}}$ are the vector representation of $q, r_{ui_{uk}}$, and $r_{u_{ik'}i}$ respectively, which we will introduce in Section 3.2.

Transformer Layers. The input sequence of vectors is then passing through transformer layers where the units interact with each other. As in [37], each transformer layer has two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network.

Let K and V denote vectors of the sequence $(q, r_{ui_{u1}}, \cdots, r_{ui_{um}}, r_{u_{i1}i}, \cdots, r_{u_{in}i})$ and let Q be the vector of any unit in the sequence. For attention head j, the output vector is computed as a weighted sum of values V according to the attention their corresponding keys K obtain with respect to Q, i.e.,

$$\text{Attn}_j(Q, K, V) = \text{softmax}(\frac{Q_j K_j^T}{\sqrt{d/h}}) V_j \tag{2}$$

where d is the dimension size of a unit vector and h is the number of attention heads. $Q_j = QW_j^Q, K_j = KW_j^K, V_j = VW_j^V$ and W_j, K_j, V_j are project matrices for Attn_j. Attn is applied for h attention heads, and each output is concatenated and mapped to the final yield of the multiple-head attention (MultiHeadAttn).

The position-wise feed-forward network (FFN) applies the same transformation to each position separately and identically. Then

there is also a residual connection for each sub-layer, which allows the input to go through the layer directly, followed by layer normalization. So the output vector $x^{(t)}$ in the t-th transformer layer ($1 \le t \le l$) at an arbitrary position from 0 to $m+n$ can be computed as:

$$\begin{aligned} x^{(t)} &= \text{LayerNorm}(y + \text{FFN}(y)) \\ y &= \text{LayerNorm}(x^{(t-1)} + \text{MultiHeadAttn}(x^{(t-1)}, K, V)) \end{aligned} \tag{3}$$

where $x^{(t-1)}$ is the output of the $(t-1)$-th layer, which can be obtained with Eq. 3 when $t > 1$ and Eq. 1 when $t = 1$. For more details about the transformers, please refer to [37].

Final Score. At last, we use the output query vector at the final layer, i.e., $\mathbf{q}^{(l)}$, which involves all the interactions between every unit in the sequence with query q to score item i. Specifically, the score of i given q and u is computed with function f:

$$f(q, u, i) = \mathbf{q}^{(l)} W_o \tag{4}$$

where $W_o \in \mathbb{R}^{d \times 1}$.

RTM can be degraded to a non-personalized model when there are no user reviews available. Also, RTM can score an item based on its descriptions when it does not have associated reviews.

3.2 Query/Review Representation

It is important to compute query and review representations on the fly so that the model can handle arbitrary queries and reviews during inference time. Previous studies [3, 35] have explored methods to construct query embeddings directly from query words, such as averaging word embeddings or applying recurrent neural networks on the query word sequence. A state-of-the-art technique is to encode a query with a non-linear projection ϕ on the average query word vectors:

$$\mathbf{q} = \phi(\{w_q | w_q \in q\}) = \tanh(W_{\phi_q} \cdot \frac{\sum_{w_q \in q} \mathbf{w_q}}{|q|} + b_{\phi_q}) \tag{5}$$

where $W_{\phi_q} \in \mathbb{R}^{d \times d}$ and $b_{\phi_q} \in \mathbb{R}^d$ are learned during training, $|q|$ is the length of query q, and $\mathbf{w_q} \in \mathbb{R}^d$ is the embedding of word w_q in q. We use the same projection function ϕ with two different parameters W_{ϕ_r} and b_{ϕ_r} to collect the initial input representation of review r.

In this way, word embeddings are shared across reviews toward the ranking optimization goal; significant words in the reviews can be emphasized with more weights in the matrix W_{ϕ_r}. Thus the interaction between reviews can capture their keyword matching.

We also considered representing reviews with another popular method, i.e., paragraph vectors [21], by predicting words in the review with the review vector. However, paragraph vectors need to be trained beforehand and thus are difficult to generalize to unseen reviews. Moreover, in paragraph vectors, word embeddings can only be updated by predicting the words with these review vectors, which is an unsupervised signal regardless of user purchases. Our experiments also show that projected average embeddings yield better results than paragraph vectors, so we exclude paragraph vectors from this paper.

Another choice to represent reviews is using pre-trained BERT [12] to encode the word sequence with transformers directly. However, we focus on modeling the interaction between the basic units, i.e., reviews, for both users and items, rather than capturing the semantic meaning carried in each review. Since reviews can be long and noisy, it is not for sure better to capture the reviews' complex semantic structures than to identify some keywords in the reviews using a reasonable and straightforward way. Also, encoding every review with BERT will introduce tremendous computation costs, which prevents us from using it.

3.3 Model Optimization

Similar to previous studies [2, 3], we optimize our model by maximizing the log likelihood of the observed (query,user,purchased item) triples, which can be written as:

$$\begin{aligned} \mathcal{L} &= \sum_{(q,u,i)} \mathcal{L}(q,u,i) = \sum_{(q,u,i)} \Big(\log P(i|q,u) + \log P(q,u) \Big) \\ &\approx \sum_{(q,u,i)} \log \frac{\exp(f(q,u,i))}{\sum_{i' \in S_i} \exp(f(q,u,i'))} \end{aligned} \tag{6}$$

where $f(q,u,\cdot)$ is computed with Eq. 4, and $\log P(q,u)$ is ignored because it is predefined as a uniform distribution. Due to the large number of items in the candidate set S_i, we adopt the negative sampling strategy [21, 25] to estimate Eq. 6 and randomly select k_{neg} negative samples from S_i according to a uniform distribution. In addition, different L2 regularization settings could not improve the performance in our experiments, which indicates that overfitting is not a problem for our experiments. Hence, we do not include the regularization terms in Eq. 6.

3.4 Model Interpretation

Existing product search models [2, 3, 6, 19] consider reviews associated with a user or item as a whole and do not differentiate their influences when matching users and items. In contrast, by conducting the attention mechanism on each unit in the sequence of the query, user reviews, and item reviews, RTM can explicitly capture the interactions between the finer-grained units, i.e., query and user/item reviews. These fine-grained interactions reflect several essential aspects of the model:

Explainable Item Matching. The attention scores of each item review concerning the query indicate which review plays a crucial role in matching this item with the purchase intent behind the query-user pair. We can rank the reviews with their attention weights from large to small for each retrieved item and display the ranking to the user. In contrast to showing item reviews according to their recency as most e-commerce platforms do, a system based on RTM could help users understand why an item is retrieved by showing the most potentially helpful reviews, which may further facilitate their purchase decisions.

Dynamic Review Representation. RTM can offer more potent learning abilities with multiple transformer layers, which could be beneficial when more interactions between reviews are needed. In a multiple-layer RTM, the review embeddings associated with the user or the item are dynamically changed based on their interactions with the other units in the sequence (q, R_u, R_i). In this case, more interactions happen between units when attending to (q, R_u, R_i) with $r_{ui_{uk}} \in R_u$ and $r_{u_{ik'}i} \in R_i$.

The final representation of a user review $r_{ui_{uk}}$ is learned from its interaction with R_u and R_i. On the one hand, the self-attention

mechanism of attending to R_u with $r_{ui_{uk}}$ updates the embedding of $r_{ui_{uk}}$ by interacting with all the reviews from user u, where similarities and dissimilarities between reviews are taken into account. On the other hand, attending to R_i with $r_{ui_{uk}}$ indicates that how the specific preference u shows for i_{uk} is satisfied by the descriptions of i from other users' perspectives. Reviews in R_i specify the other users' opinion towards item i, which could reflect i's advantages and disadvantages user u may care about.

Similarly, the final vector of an item review $r_{u_{ik'}i}$ is also dynamically changed according to its interaction with R_u and R_i. The interactions between $r_{u_{ik'}i}$ and R_i readjust the representation of $r_{u_{ik'}i}$ by considering its relation with the other reviews in R_i. In addition, attending to the reviews in R_u with $r_{u_{ik'}i}$ indicates how $r_{u_{ik'}i}$ matches the preferences expressed in the user's each historical review. This information and how each user preference is satisfied by item reviews carried in $r_{ui_{uk}}$ could be both beneficial to score item i.

Dynamic Personalization. RTM has the flexibility to make predictions with variable (none to full) emphasis on personalization, similar to [6], by learning different accumulative weights for the user's reviews. In RTM, $q^{(l)}$ in Eq. 4 is computed from attending to the sequence of (q, R_u, R_i) in the $(l-1)$-th layer with q. Personalization can take full effect when the attention weight assigned to q is 0 and no effect when the reviews in R_u have 0 accumulative attention weights.

In addition, in contrast to [2, 6] where personalization degree for a user depends only on her query, RTM is more flexible since it could perform various degrees of personalization for different items given the same query-user pair since attention weights on R_u could vary when R_i is different even with the same q. This strategy allows the user profile to have differentiated effects when it contains much or no useful information while matching various items.

4 EXPERIMENTAL SETUP

In this section, we first show how we construct the datasets and conduct evaluation, then we describe the baseline models and training settings for different models.

4.1 Datasets and Evaluation

The Amazon product search dataset [24] [1] is the only available dataset for product search that have user reviews. As in previous work [3, 4, 6, 7, 18, 35, 43], we use it for our experiments. Specifically, we use the 5-core data [24] where each user and each item has at least 5 associated reviews. Our experiments are based on three categories of different scales, which are *Clothing, Shoes & Jewelry*, *Sports & Outdoors*, and *CDs & Vinyl*. The statistics are shown in Table 1.

Query Construction. Following the same paradigm used in [3, 4, 6, 7, 18, 35, 43], we construct queries for each purchased item with the product category information. This strategy is based on the finding that directed product search is users' search for a producer's name, a brand, or a set of terms describing product category [30]. Precisely, we extract the multi-level category information from the meta-data, concatenate the words in the categories, and remove stopwords and duplicate words to form a query string. Since an item could belong to multiple categories, there may be multiple extracted queries for the item. Each query is considered as the initial query issued by the user and leading to purchasing the item. The queries are general and do not reveal specific details of the purchase items.

[1] http://jmcauley.ucsd.edu/data/amazon/

Table 1: Statistics of the Amazon datasets.

Dataset	Sports & Outdoors	Clothing, Shoes & Jewelry	CDs & Vinyl
#Users	35,598	39,387	75,258
#Items	18,357	23,033	64,443
#Reivews	296,337	278,677	1,097,591
#Queries	1,538	2,021	695
#Vocab	32,386	21,366	202,959
ReviewLen	89.18±106.99	62.22±60.16	174.56 ±177.05
QueryLen	7.07 ± 1.74	7.14±1.97	5.77±1.65
#Query-user pairs			
Train	269,850	467,651	1,524,168
Valid/Test	776/910	4,106/4,025	10,930/10,077
#Purchased items per query-user pair			
Train	1.16±0.55	1.30±0.82	2.95±8.53
Valid/Test	1.01±0.08/1.02±0.24	1.00±0.08/1.00±0.05	1.12±0.61/1.11±0.56

Training/Validation/Test Splits. As in [6], we split each dataset into training, validation, and test sets according to the following steps. First, we randomly put 70% queries in the training set, and the rest 30% are shared by the validation and test sets so that none of the test queries have been seen during training. Then, we partition each user's' purchases to training, validation, and test set according to the ratio 0.8:0.1:0.1 in chronological order. If none of the queries associated with the purchased item is in the test set, the purchase will be moved back to the training set. In contrast to randomly partitioning data into training and test set as in previous work [3, 4, 7, 43], our dataset is closer to a real scenario, where all the purchases in the test set happen after the purchases in the training set. This also makes retrieval on our test set harder since future information can be used to predict past purchases in the previous datasets. Also, our training and test sets have less similar distributions compared with previous datasets, which makes model prediction more difficult as well.

Evaluation Metrics. Following the typical way of collecting candidates with an efficient initial ranker and using neural models for re-ranking in document retrieval, we re-rank the candidate items retrieved by BM25 [29] with each method and obtain the ranking lists. [2] Then we use Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain at 20 (NDCG@20), and Recall at 20 (R@20) as evaluation metrics. MRR shows the first position where any purchased item is retrieved; NDCG@20 focuses on the top 20 items' ranking performance where higher positions have more credits, and R@20 indicates how many target items are retrieved in the top 20 results in total. Fisher random test [32] with $p < 0.05$ is used to measure significant differences between results.

4.2 Baselines

We compare our RTM with eight representative baselines:

BM25: The BM25 [29] model is based on word matching between queries and item reviews, which also provides the initial ranking lists for the other models.

[2] In the experiments where each method ranks all the items in the collections, we have similar observations, so we do not include this setting for space concerns.

POP: The Popularity (POP) model ranks items according to their frequency of being purchased in the training set.

LSE: The Latent Semantic Entity model (LSE) [35] is an embedding-based non-personalized model that learns the vectors of words and products by predicting the products with n-grams in their reviews. It then scores the products with the cosine similarity between their vectors and query vectors.

QEM: The Query Embedding Model (QEM) [2], is also a non-personalized model that conducts item generation based on the query embedding, and item embeddings are learned by predicting words in their associated reviews.

HEM: The Hierarchical Embedding Model (HEM) [3] has the item generation model and language models of users and items. It balances the effect of personalization by applying a convex combination of user and query representation.

AEM: The Attention-based Embedding Model (AEM) [2, 18] constructs query-dependent user embeddings by attending to users' historical purchased items with the query. [3]

ZAM: The Zero Attention Model (ZAM) [2] extends AEM with a zero vector and conducts differentiated personalization by allowing the query to attend to the zero vector.

TEM: The Transformer-based Embedding Model (TEM) [6] is a state-of-the-art model that encodes query and historically purchased items with a transformer and does item generation based on the encoded query-user information.

BM25, POP, LSE, and QEM are non-personalized retrieval models, and all the rest are personalized product search models.

4.3 Implementation Details

All the baselines were trained for 20 epochs with 384 samples in each batch according to the settings in their original papers [2, 3, 6], and they can converge well. We trained our model for 30 epochs with 128 samples in each batch. In the baseline models, each word in the reviews of a target item corresponds to one entry (word, item, user, query) in the batch, while in RTM each target item has one entry (item, user, query) in a batch. The number of negative samples for items or words in all the models is set to 5. We set the embedding size of all the models to 128. Larger embedding sizes do not lead to significant differences, so we only report results with $d = 128$. The sub-sampling rate of words in all the neural baseline models is set to $1e-5$. We sweep the number of attention heads h from {1,2,4,8} for AEM, ZAM, TEM, and our RTM. For TEM and RTM, we vary the number of transformer layers l from 1 to 3 and set the dimension size of the feed-forward sub-layer of the transformer from {128, 256, 512}. We cutoff reviews to 100 words and limit the number of historical reviews for a user and an item, i.e., m and n in Figure 1, to 10 and 30, respectively. We use Adam with $\beta_1 = 0.9, \beta_2 = 0.999$ to optimize RTM. The learning rate is initially set from {0.002, 0.005, 0.01, 0.02} and then warm-up over the first 8,000 steps, following the paradigm in [12, 37]. To make the training of RTM more stable, we initialize the parameters of words with embeddings pre-trained with Word2Vec [25]. For the number of candidates, we use the top 100 results from BM25 for re-ranking on *Sports* and *Clothing* whose recall values are 0.425 and 0.343, respectively. On *CDs*, the *Recall*@100 of BM25 is only 0.108, so we use the top 1000 items for re-ranking, which has a higher recall - 0.370. Our code can be found at https://github.com/kepingbi/ProdSearch.

[3]The attention models described in [2] and [18] are highly similar to each other, so we only implement the one in [2] and named it as AEM.

5 RESULTS AND DISCUSSION

In this section, we first compare the overall performance of RTM and the baseline models. Then we conduct model analysis and case studies to interpret the model behavior.

5.1 Overall Performance

Table 2 shows each system's overall ranking performance on the three datasets. We show four variants of RTM that use both, either, or none of the position embeddings and segment embeddings by setting $\mathcal{I}_{pos}$ and $\mathcal{I}_{seg}$ in Eq. 1 to 0 or 1. We illustrate the effect of position and segment embeddings in Section 5.2. Note that the numbers in Table 2 are small for two reasons: 1) there is only about 1 target purchased item out of 20k~65k items for each query-user pair in the test sets, shown in Table 1. 2) as we mentioned in Section 4.1, the search task on our sequentially split data is more challenging than on the randomly divided partitions in [3, 7, 43].

As in previous studies [2, 3, 18], we observe that non-personalized models perform worse than personalized models. BM25 performs better on *Sports* and *Clothing* than *CDs*, which indicates that term matching plays a more important on *Sports* and *Clothing*. In contrast, POP matters more on *CDs* than *Sports* and *Clothing*. Non-personalized neural models do not always outperform BM25, especially on *Sports* and *Clothing*, probably because semantic matching brings limited benefits when most candidate results from BM25 have exact term matching.

Among the personalized retrieval baselines, TEM achieves the best performance, which is consistent with [6] and confirms the benefit of using transformers. ZAM performs better than AEM most of the time, indicating that dynamic personalization is helpful. Similar to [2, 18], we also observe that HEM could not outperform the attention-based models, which indicates that building a dynamic user profile helps improve the result quality compared with using static user representation across the search sessions.

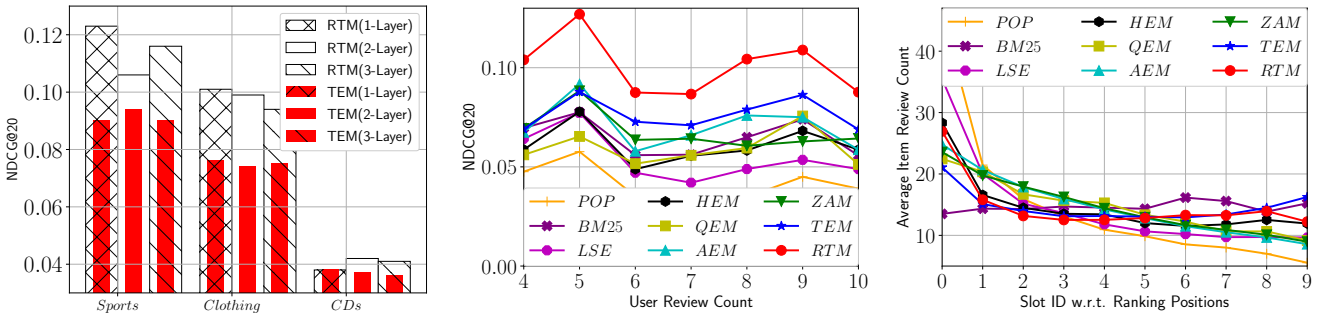
On all the categories, RTM achieves the best performance. It has significant improvements upon the best baseline – TEM – in most metrics on all the datasets. It confirms that by modeling the dynamic matching between user and item, RTM has significant advantages over models with static item profiles, although some of them also have dynamic user profiles (AEM, ZEM, and TEM). In addition, by capturing the finer-grained matching at the review level, RTM can differentiate different items better than matching them at the item level in TEM.

5.2 Model Analysis

Position and Segment Embeddings. As shown in Table 2, RTM without position and segment embeddings has the worse performance most of the time, showing the necessity to differentiate the reviews from the user and the item. On *Sports* and *Clothing*, position embeddings are always helpful, which indicates that the user's recent purchases have different influences from the long-ago purchases. The latest reviews reveal more accurate information about the products. For example, users' preferences on clothes styles

Table 2: Comparison between the baselines and our proposed RTM. '*' marks the best baseline performance. '†' indicates significant improvements over all the baselines in Fisher Random test [32] with $p < 0.05$.

Dataset		Sports & Outdoors			Clothing, Shoes & Jewelry			CDs & Vinyl		
Model		*MRR*	*NDCG@20*	*R@20*	*MRR*	*NDCG@20*	*R@20*	*MRR*	*NDCG@20*	*R@20*
Non-personalized	BM25	0.049	0.051	0.173	0.044	0.051	0.160	0.011	0.015	0.042
	POP	0.033	0.055	0.158	0.030	0.044	0.112	0.015	0.018	0.039
	LSE	0.026	0.047	0.148	0.041	0.058	0.135	0.017	0.021	0.044
	QEM	0.049	0.070	0.172	0.039	0.057	0.144	0.010	0.014	0.037
Personalized	HEM	0.044	0.071	0.197	0.043	0.061	0.144	0.021	0.030	0.073
	AEM	0.047	0.076	0.209	0.048	0.069	0.162	0.023	0.033	0.084
	ZAM	0.052	0.083	0.220	0.047	0.069	0.166	0.025	0.036	0.086
	TEM	0.060*	0.094*	0.238*	0.052*	0.076*	0.182*	0.026*	0.038*	**0.095***
	RTM ($\mathcal{I}_{pos}$=0, $\mathcal{I}_{seg}$=0)	0.065	0.092	0.208	0.061†	0.087†	0.190†	0.027	0.036	0.084
	RTM ($\mathcal{I}_{pos}$=0, $\mathcal{I}_{seg}$=1)	0.058	0.093	**0.242**	0.068†	0.099†	**0.224**†	**0.030**†	**0.042**†	**0.095**
	RTM ($\mathcal{I}_{pos}$=1, $\mathcal{I}_{seg}$=0)	0.082†	0.110†	0.234	**0.071**†	**0.101**†	0.219†	**0.030**†	0.039	0.085
	RTM ($\mathcal{I}_{pos}$=1, $\mathcal{I}_{seg}$=1)	**0.096**†	**0.123**†	0.237	0.069†	0.099†	0.218†	0.028	0.040	0.094



(a) Effect of Transformer Layer Numbers **(b) Effect of User Review Counts** **(c) Item Review Count w.r.t. Ranking Positions**

Figure 2: Model analysis in terms of different aspects. Figure 2b and 2c correspond to Clothing, Shoes & Jewelry.

may change according to the seasonal trend; a swimming earplug product has been updated lately, and recent customer reviews complained the new version is not as comfortable as before. In these cases, position embeddings that capture reviews' recency can help identify the current user preference and item status and potentially improve the search quality.

In contrast, on *CDs*, incorporating position embeddings does not lead to better evaluation results than using segment embeddings alone. This shows that the order of user and item reviews do not need to be differentiated as long as we know which of them corresponds to the user and item respectively. This observation is consistent with our intuition that the content in a CD and sound quality are usually static regardless of the review order. We can also infer that long-ago purchases play similar roles as recent purchases in terms of representing user preferences on *CDs*.

Using both position and segment embeddings does not always lead to the best evaluation results. The possible reasons are that the low and high positions can indicate which sections of the input correspond to the user and item respectively, which makes segment embeddings not necessary sometimes. When the chronological order of reviews is not crucial in some categories, the sequence of reviews does not matter so that position embeddings could introduce noise to the model and harm the performance, as we mentioned in Section 3.1.

Number of Layers. RTM achieves the best performance with 2 layers on *CDs* and 1 layer on *Sports* and *Clothing*. This indicates that the dynamic review representation introduced in Section 3.4 is beneficial on larger datasets. As shown in Table 1, *CDs* has more average reviews per user/item than the other two datasets, and it also has a larger vocabulary, which makes the contexts of each review more varied. We will further analyze the behavior of RTM with single and multiple layers in Section 5.3 by case studies.

User Review Count. Figure 2b shows the performance of query-user pairs with different counts of user reviews in the training set of *Clothing*. The other two datasets show similar trends. The minimum number is 4 since each user has at least five reviews and 10% of them has been put to the validation or test set, as mentioned in Section 4.1. According to Section 4.3, the max number is 10 since we use at most 10 historical reviews for users. The corresponding numbers of the query-user pairs with user review count from 4 to 10 are 1297, 655, 453, 417, 243, 180, and 780, respectively. We can see that RTM has consistent improvements over other models on query-user pairs with different review counts and RTM can achieve compelling performance with a decent small number of user reviews.

Table 3: A case of query, user, and purchased item for the single-layer RTM case study. The attention weights are average from all the 8 attention heads.

Query: **"clothing shoe jewelry men big tall active athletic sock"** (Attention weight w.r.t. $q^{(0)}$:0.031)					
Attn	ID	User Reviews	Attn	ID	Item Reviews
0.027	ur3	Fruit of the Loom **T-Shirt**. This shirt is a great value for the price. **It is snug and fits me perfectly**. There is enough room to wear the shirt with an under-shirt as well, giving me warmth.	0.042	ir9	As expected bought these socks for my **husband** because he is always wearing holes in his socks, and I am looking to Carhartt to provide a sock that might be able to better withstand his abuses. So far they are **sufficiently cushiony**, **they stay up on his legs** and get the job done. Time will tell if they are as durable as I am hoping.
0.010	ur2	Casio Men's MQ24-1E black resin **watch**. I love it because it is small, easy to fasten, lightweight, and inexpensive. I had to cut the band for a smaller wrist, but glad I bought the watch.	0.018	ir4	Can't go wrong with Carhartt. These are great socks. There are great for everyday work. They're comfortable and durable. It's a great product Not much more to say.

Item Review Count w.r.t. Ranking Positions. Figure 2c compares different methods in terms of their tendency to rank items with more reviews to higher positions, i.e., their preferences on popularity. We only show *Clothing* since the other two datasets have similar trends. For the top 100 items ranked by each method, we group them into 10 slots and show the items' average review counts in each slot. For example, Slots 0 and 1 correspond to items that are ranked from 1 to 10 and 11 to 20. We observe that BM25 and POP have the least and most tendency to value popular items respectively, which is consistent with their principles of only emphasizing relevance or popularity. Among other methods, LSE emphasizes popularity the most; HEM and RTM put more popular items to top 10 and less popular items to positions from 11 to 50 than ZAM, AEM, and QEM; TEM ranks the fewest popular items to top 10 and has similar value to RTM at Slot 1. Overall, the fact that RTM achieves better performance than baselines suggests that RTM could better balance popularity and relevance by the review-level matching.

5.3 Case Study

We sample two cases in the test set of *Clothing* and *CDs* from our best model to illustrate the three aspects of RTM introduced in Section 3.4 and show how RTM identifies useful information from review-level user-item interactions. We show one example in *Clothing* to represent the case of a single-layer RTM, and the other example in *CDs* to illustrate how a multi-layer RTM performs.

Case Analysis for Single-layer RTM. In Figure 3, we show how RTM allocates attention to the sequence of the query, user's historical reviews, and reviews of an ideal item for the query "clothing shoe jewelry men big tall active athletic sock" with respect to $q^{(0)}$. There are 8 attention heads in total and they capture different aspects of the sequence. $h2$ and $h5$ focus more on how to allocate attention to each user reviews, while the other heads concentrate on differentiating important item reviews. Overall, the item reviews have the most portion of accumulative attention weights from the query, which is reasonable since item reviews are more important to differentiate candidate items. The accumulative attention weights on the user side are positive, which shows that user's historical reviews do help and personalization is needed.

To compare the reviews that take the most and least effect on matching the item with the query-user pair, we show the review text of $ur3$, $ur2$, $ir9$, and $ir4$ (the 3rd, 2nd user review and the 9th, 4th item review) and their attention weights in Table 3. The query "clothing shoe jewelry men big tall active athletic sock" attends to $ur3$ – a previous review on a T-shirt – the most, and $ur2$ – a historical review on a Casio watch – the least. This makes sense since

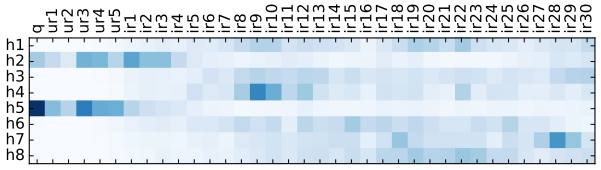


Figure 3: Attention weights for the case shown in Table 3.

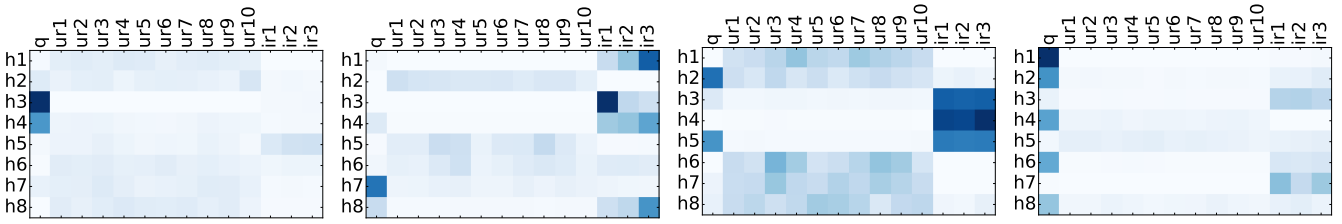
socks share more common properties with T-shirts than watches, such as the material and the fitness. For the item reviews, RTM allocates the most weight to $ir9$ which includes a lot of useful information for "men big tall active athletic sock", such as that the socks are for male, they are cushiony, and they stay up on legs. On the contrary, $ir4$, which receives the least attention, does not reveal whether the socks are for men or women, and the descriptions such as "comfortable" and "'durable" can also be applied to other products. The attention weights can help explain why RTM ranks this item to the top, and showing reviews with large attention weights to the user could help them better understand why the item is a good candidate and facilitate their purchase decisions.

Case Analysis for Multi-layer RTM. For query "CDs vinyl Europe jazz", the attention scores of each unit in the sequence of the query, user reviews, and the reviews of a purchased item with respect to different units are shown in Figure 4. In the first transformer layer, most of the attention is paid to the query itself by $q^{(0)}$, shown in Figure 4a. Figure 4b shows that in the second layer, item reviews draw most attention weights from $q^{(1)}$. These two figures imply that a single layer is not enough to learn the dynamic matching between the query-user pair and the item, probably because the initial representations are not informative enough.

From the average attention weight of each unit with respect to $q^{(1)}$ from the 8 heads, the 3 item reviews have the top 3 attention scores, ranked as $ir3$, $ir1$, $ir2$ (in Figure 4b). These attention weights offer a possible explanation for how this item is scored. We can verify the explanation's rationality by checking the text of the item reviews shown in Table 4. $ir3$ mentions that this CD is Jan's best recording in a long time and Jan Garbarek is a Norwegian jazz saxophonist, which is relevant to the query "CDs vinyl Europe jazz". $ir1$ recommends the CD to people who like modern jazz and implies that it is from a European musician - Jan Garbarek. $ir2$ does not mention jazz at all and considers this album not bad and also not excellent. $ir3$ is more positive than $ir1$ on the album, and both of them are more informative and positive than $ir2$, which indicates the explanation of the item score from RTM is reasonable.

Table 4: A case of query, user, and purchased item for the multi-layer RTM case study. The attention weights are average from all the 8 attention heads.

Query: **"CDs vinyl Europe jazz"** (Attention weight w.r.t $q^{(1)}$: 0.097; w.r.t $q^{(0)}$: 0.215)									
Attention w.r.t.			ID	User Reviews	Attention w.r.t.			ID	Item Reviews
$q^{(1)}$	$r^{(0)}_{ui_{u4}}$	$r^{(0)}_{u_{i3}i}$			$q^{(1)}$	$r^{(0)}_{ui_{u4}}$	$r^{(0)}_{u_{i3}i}$		
0.055	0.065	0.026	ur4	... Before buying this I already owned 12 **Garbarek** albums, must admit though **I'd pretty much heard "all" he had to offer** ...	0.196	0.115	0.131	ir3	... I really liked **Jan** in the seventies. ... **I believe it is Jan's best recording in a long time.** I don't prefer it to his earlier Avante guard or **jazzier** stuff ..., and **Jan**'s always great solos.
0.053	0.065	0.028	ur8	Aural attack To all current **EST** fans - if news of **Svensson**'s death wasn't hard enough to take, the music on this album is what you might call "tough love"...	0.167	0.110	0.130	ir1	Excellent introduction to **Garbarek**'s world for newbies ...If you like modern **jazz** and/or ECM-style musicians, buy it without hesitation! ...
0.052	0.073	0.028	ur3	... there's much more of **Steve Reich**, **Christian Wallumrod** & even **Esbjorn Svensson** here, meaning that whilst every piece is structured such that you can often predict when an established, largely minimalist pattern is going to change ...	0.127	0.109	0.114	ir2	... Why some people adore this **Garbarek**'s release while others simply hate it? ... Not "excellent" but for sure "not bad"... This is - in essence - an IMPROVISATION album. If you are not open to this freestyle, probably you'll not like it, ... Anyway, decide with your own ears and criteria.



(a) Attention with respect to $q^{(0)}$ **(b) Attention with respect to $q^{(1)}$** **(c) Attention with respect to $r^{(0)}_{ui_{u4}}$** **(d) Attention with respect to $r^{(0)}_{u_{i3}i}$**

Figure 4: Attention weights for the case shown in Table 4. $r_{ui_{u4}}$ and $r_{u_{i3}i}$ denote the same review as $ur4$ and $ir3$ respectively.

Among the user reviews, $ur4$, $ur8$, and $ur3$ receive the top attention with respect to $q^{(1)}$ (in Figure 4b). From the review text shown in Table 4, we can see that $ur4$ is written by the user for a previously purchased album from the European jazz musician Jan Garbarek. In $ur8$, EST is short for "Esbjörn Svensson Trio", which was a Swedish jazz piano trio, indicating the album with the review is also related to European jazz. $ur3$ mention Steve Reich (an American composer who also plays jazz), Christian Wallumrod (a Swedish jazz pianist), and Esbjörn Svensson (a Norwegian jazz musician), which are also related to the query. These reviews are useful to indicate the user's preference for European jazz and should draw more attention with respect to the query.

To find out which unit has more effect on $ur4$ and $ir3$ in the first transformer layer, we also show the attention distribution of each unit with respect to $r^{(0)}_{ui_{u4}}$ (ur4) and $r^{(0)}_{u_{i3}i}$ (ir3) in Figure 4c and 4d. For $ur4$, the top 5 units that it attends to are $ir3$, $ir1$, $ir2$, $ur3$, q, and $ur8$, from which we can tell that $ir3$ has the largest contribution in terms of satisfying preferences indicated by $ur4$. As shown in Table 4, $ur4$ indicates that the user is a big fan of Jan Garbarek and has listened to almost all his albums, so the comment in $ir3$ that this album is the best of Garbarek is quite persuasive to the user to purchase the item. In Figure 4d, the units with top attention weights with respect to $ir3$ are q, $ir3$, $ir1$, $ir2$, $ur3$, $ur8$, and $ur4$. This implies that the final representation of $ir3$ depends mostly on the query and all the item reviews, including itself. The attention weights also indicate that $ir3$ considers itself to satisfy the query and the preferences expressed in the user reviews $ur3$, $ur8$, and $ur4$ the most. These observations are consistent with our interpretation on the dynamic review representation of a multi-layer RTM in Section 3.4.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a review-based transformer model (RTM) for personalized product search, which scores an item by encoding the sequence of the query, user reviews, and item reviews with a transformer architecture. RTM conducts review-level matching between a query-user pair and an item instead of the popular paradigm that represents and matches users and items with static representations in the semantic space. Each user and item review's importance is dynamically changed according to other units in the sequence, which enables RTM to perform adaptive personalization and the dynamic utilization of the item information in different search sessions. The empirical results show that RTM not only improves product search quality but also provides useful information to explain why an item is ranked to the top.

As a next step, we are interested in investigating incorporating other information from users and items such as brand, category, the relationship of also-purchased, and also-viewed, etc., with transformers to weigh each information source dynamically across search sessions.

ACKNOWLEDGMENTS

This work was supported in part by the Center for Intelligent Information Retrieval. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

REFERENCES

[1] Aman Ahuja, Nikhil Rao, Sumeet Katariya, Karthik Subbian, and Chandan K Reddy. 2020. Language-Agnostic Representation Learning for Product Search on E-Commerce Platforms. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 7–15.

[2] Qingyao Ai, Daniel N Hill, SVN Vishwanathan, and W Bruce Croft. 2019. A zero attention model for personalized product search. In *CIKM'19*. 379–388.

[3] Qingyao Ai, Yongfeng Zhang, Keping Bi, Xu Chen, and W Bruce Croft. 2017. Learning a hierarchical embedding model for personalized product search. In *SIGIR'17*. ACM, 645–654.

[4] Qingyao Ai, Yongfeng Zhang, Keping Bi, and W Bruce Croft. 2019. Explainable Product Search with a Dynamic Relation Embedding Model. *ACM Transactions on Information Systems (TOIS)* 38, 1 (2019), 1–29.

[5] Paul N Bennett, Ryen W White, Wei Chu, Susan T Dumais, Peter Bailey, Fedor Borisyuk, and Xiaoyuan Cui. 2012. Modeling the impact of short-and long-term behavior on search personalization. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 185–194.

[6] Keping Bi, Qingyao Ai, and W. Bruce Croft. 2020. A Transformer-based Embedding Model for Personalized Product Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, ACM, New York, NY., 1521–1524. https://doi.org/10.1145/3397271.3401192.

[7] Keping Bi, Qingyao Ai, Yongfeng Zhang, and W Bruce Croft. 2019. Conversational product search based on negative feedback. In *CIKM'19*. 359–368.

[8] Keping Bi, Choon Hui Teo, Yesh Dattatreya, Vijai Mohan, and W Bruce Croft. 2019. A Study of Context Dependencies in Multi-page Product Search. In *CIKM'19*.

[9] Qiannan Cheng, Zhaochun Ren, Yujie Lin, Pengjie Ren, Zhumin Chen, Xiangyuan Liu, and Maarten de Rijke. 2021. Long Short-Term Session Search: Joint Personalized Reranking and Next Query Prediction. (2021).

[10] Paul Alexandru Chirita, Wolfgang Nejdl, Raluca Paiu, and Christian Kohlschütter. 2005. Using ODP metadata to personalize search. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. 178–185.

[11] Zhuyun Dai and Jamie Callan. 2019. Deeper text understanding for IR with contextual neural language modeling. In *SIGIR'19*. 985–988.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[13] Wei Di, Anurag Bhardwaj, Vignesh Jagadeesh, Robinson Piramuthu, and Elizabeth Churchill. 2014. When relevance is not enough: Promoting visual attractiveness for fashion e-commerce. *arXiv preprint arXiv:1406.3561* (2014).

[14] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of the 16th international conference on World Wide Web*. 581–590.

[15] Huizhong Duan and ChengXiang Zhai. 2015. Mining coordinated intent representation for entity search and recommendation. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. 333–342.

[16] Huizhong Duan, ChengXiang Zhai, Jinxing Cheng, and Abhishek Gattani. 2013. A probabilistic mixture model for mining and analyzing product search log. In *CIKM'13*. ACM, 2179–2188.

[17] Susan T Dumais. 2016. Personalized Search: Potential and Pitfalls.. In *CIKM*. 689.

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

[19] Yangyang Guo, Zhiyong Cheng, Liqiang Nie, Xin-Shun Xu, and Mohan Kankanhalli. 2018. Multi-modal preference modeling for product search. In *2018 ACM Multimedia Conference on Multimedia Conference*. ACM, 1865–1873.

[20] Shubhra Kanti Karmaker Santu, Parikshit Sondhi, and ChengXiang Zhai. 2017. On application of learning to rank for e-commerce search. In *SIGIR'17*. ACM, 475–484.

[21] Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *International conference on machine learning*. 1188–1196.

[22] Soon Chong Johnson Lim, Ying Liu, and Wing Bun Lee. 2010. Multi-facet product information search and retrieval using semantically annotated product family ontology. *Information Processing & Management* 46, 4 (2010), 479–493.

[23] Bo Long, Jiang Bian, Anlei Dong, and Yi Chang. 2012. Enhancing product search by best-selling prediction in e-commerce. In *CIKM'12*. ACM, 2479–2482.

[24] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *SIGKDD'15*. ACM, 785–794.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[26] Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage Re-ranking with BERT. *arXiv preprint arXiv:1901.04085* (2019).

[27] Nish Parikh and Neel Sundaresan. 2011. Beyond relevance in marketplace search. In *CIKM'11*. ACM, 2109–2112.

[28] Chen Qu, Liu Yang, Minghui Qiu, Yongfeng Zhang, Cen Chen, W Bruce Croft, and Mohit Iyyer. 2019. Attentive History Selection for Conversational Question Answering. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1391–1400.

[29] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995).

[30] Jennifer Rowley. 2000. Product search in e-shopping: a review and research propositions. *Journal of consumer marketing* 17, 1 (2000), 20–35.

[31] Xuehua Shen, Bin Tan, and ChengXiang Zhai. 2005. Implicit user modeling for personalized search. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. 824–831.

[32] Mark D Smucker, James Allan, and Ben Carterette. 2007. A comparison of statistical significance tests for information retrieval evaluation. In *CIKM'07*. ACM, 623–632.

[33] Jaime Teevan, Susan T Dumais, and Eric Horvitz. 2005. Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*. 449–456.

[34] Jaime Teevan, Susan T Dumais, and Daniel J Liebling. 2008. To personalize or not to personalize: modeling queries with variation in user intent. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. 163–170.

[35] Christophe Van Gysel, Maarten de Rijke, and Evangelos Kanoulas. 2016. Learning latent vector spaces for product search. In *CIKM'16*. ACM, 165–174.

[36] Damir Vandic, Flavius Frasincar, and Uzay Kaymak. 2013. Facet selection algorithms for web product search. In *CIKM'13*. ACM, 2327–2332.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[38] Ryen W White and Steven M Drucker. 2007. Investigating behavioral variability in web search. In *Proceedings of the 16th international conference on World Wide Web*. 21–30.

[39] Liang Wu, Diane Hu, Liangjie Hong, and Huan Liu. 2018. Turning clicks into purchases: Revenue optimization for product search in e-commerce. In *SIGIR'18*. ACM, 365–374.

[40] Zhijing Wu, Jiaxin Mao, Yiqun Liu, Jingtao Zhan, Yukun Zheng, Min Zhang, and Shaoping Ma. 2020. Leveraging Passage-level Cumulative Gain for Document Ranking. In *WWW'20*.

[41] Teng Xiao, Jiaxin Ren, Zaiqiao Meng, Huan Sun, and Shangsong Liang. 2019. Dynamic Bayesian Metric Learning for Personalized Product Search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1693–1702.

[42] Jun Yu, Sunil Mohan, Duangmanee Pew Putthividhya, and Weng-Keen Wong. 2014. Latent dirichlet allocation based diversified retrieval for e-commerce search. In *WSDM'14*. ACM, 463–472.

[43] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *CIKM'18*. ACM, 177–186.

[44] Yuan Zhang, Dong Wang, and Yan Zhang. 2019. Neural IR Meets Graph Embedding: A Ranking Model for Product Search. In *The World Wide Web Conference*. 2390–2400.

[45] Jie Zou and Evangelos Kanoulas. 2019. Learning to ask: Question-based sequential Bayesian product search. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 369–378.