PSSL: Self-supervised Learning for Personalized Search with Contrastive Sampling

Yujia Zhou[2], Zhicheng Dou[1*], Yutao Zhu[3], and Ji-Rong Wen[4,5]
[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]School of Information, Renmin University of China
[3]Université de Montréal, Montréal, Québec, Canada
[4]Beijing Key Laboratory of Big Data Management and Analysis Methods
[5]Key Laboratory of Data Engineering and Knowledge Engineering, MOE
{zhouyujia,*dou}@ruc.edu.cn

ABSTRACT

Personalized search plays a crucial role in improving user search experience owing to its ability to build user profiles based on historical behaviors. Previous studies have made great progress in extracting personal signals from the query log and learning user representations. However, neural personalized search is extremely dependent on sufficient data to train the user model. Data sparsity is an inevitable challenge for existing methods to learn high-quality user representations. Moreover, the overemphasis on final ranking quality leads to rough data representations and impairs the generalizability of the model. To tackle these issues, we propose a **P**ersonalized **S**earch framework with **S**elf-supervised **L**earning (PSSL) to enhance data representations. Specifically, we adopt a contrastive sampling method to extract paired self-supervised information from sequences of user behaviors in query logs. Four auxiliary tasks are designed to pre-train the sentence encoder and the sequence encoder used in the ranking model. They are optimized by contrastive loss which aims to close the distance between similar user sequences, queries, and documents. Experimental results on two datasets demonstrate that our proposed model PSSL achieves state-of-the-art performance compared with existing baselines.

CCS CONCEPTS

• **Information systems → Personalization**.

KEYWORDS

Personalized search; Self-supervised learning; Contrastive learning

ACM Reference Format:
Yujia Zhou, Zhicheng Dou, Yutao Zhu, Ji-Rong Wen. 2021. PSSL: Self-supervised Learning for Personalized Search with Contrastive Sampling. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482379

1 INTRODUCTION

Search engines have been widely employed across the world as a common tool for retrieving information. For the same query, these platforms usually return the same document list for all users. This strategy is difficult to meet the needs of all users due to the diversity of user interests. To cope with this problem, personalized search has been proposed to re-rank the search results to meet user's individual needs [11, 30]. One essential problem in personalized search is how to model the user preferences with respect to his query log. Some early studies [13, 31, 35, 36] tried to extract personalized features from user's click-through data to predict user interests. Recently, deep learning based methods [17, 40, 41, 44, 45] have been proposed to build user profiles in semantic space. They applied neural networks to learn effective user representations and brought significant improvements in user satisfaction.

Although existing neural methods have made great progress in improving user search experience, there are two weaknesses that limit their ability to build user profiles. **First**, these models rely on large amounts of training data to learn user representations more accurately. However, some users only have limited click-through data to train the user representations. Data sparsity is a major challenge for neural personalized search models. **Second**, existing approaches optimize the model with only the personalized ranking task, which will make the model overemphasize the final ranking quality. The characteristics of query logs, such as the correlations between user behaviors, are not well captured in data representations. Such rough data representations will damage the generalizability of the model. In fact, better data representations can further improve the quality of search results. As can be seen from language models proposed recently [10, 22, 29, 46], the pre-trained data representations benefit the performance of various downstream tasks. This inspires us to rethink the process of model optimization in personalized search. We attempt to learn pre-trained data representations adapting to personalized search, and fine-tune them on the ranking task.

Before the ranking task, how to incorporate the characteristics of personalized search logs in data representations has become a new challenge. Recently, self-supervised learning has achieved great success in various information retrieval tasks, such as sequential recommendation [38, 42] and ad-hoc document ranking [6, 18]. This is a new paradigm for unsupervised representation learning, which aims to learn intrinsic data correlation from the raw data without any supervision signal. The basic process of self-supervised learning is to first construct training samples, and then devise auxiliary tasks to pre-train the model. Its advantages perfectly fit our needs in dealing with the aforementioned two problems: data sparsity and rough data representations. Due to its powerful ability of extracting self-supervised signals from raw data, we intend to apply it to capture behavioral characteristics in query logs, thereby enhancing data representations for personalized search.

To support personalized search, we need two different categories of representation learning tasks: sentence encoding and sequence encoding. The former aims to learn the representations of queries and documents, while the latter focuses on behavior sequence modeling to obtain the user representations based on their query logs. In fact, the users' query logs contain a lot of paired self-supervised information. For instance, if a user issues two similar queries to find the same document, these two queries reveal the same intention. Moreover, two users who submit the same query to retrieve the same document should show some similarities. To model such paired search patterns in the query logs, we adopt a contrastive sampling method to generate self-supervised signals for pre-training tasks. Under this strategy, we can extract contrastive pairs that show similar meanings from query logs. They help pre-train the parameters of encoders with contrastive learning objectives [7, 14].

Specifically, we propose a **P**ersonalized **S**earch framework with **S**elf-supervised **L**earning (PSSL), which is a neural model with two-stage training. **At the first stage**, we use self-contrastive sampling and user-contrastive sampling to generate self-supervised signals. The former constructs the contrastive pairs from the query log of a single user, while the latter extracts pairs from different users. Based on the sampling from these two angles, four types of pairs, namely query pairs, document pairs, sequence augmentation pairs, and user pairs, are constructed for self-supervised learning. They correspond to four self-supervised tasks to pre-train the sentence encoder and the sequence encoder. As such, pre-trained encoders adapt data representations to personalized search scenarios. **At the second stage**, two encoders are applied to enhance the personalization, which will be fine-tuned with respect to the ranking quality. To verify the validity of our model, we conduct experiments on search logs from two real-world search engines. Experimental results demonstrate that our model PSSL achieves state-of-the-art performance compared with existing search models.

Our main contributions can be summarized as follows. (1) We propose a two-stage training framework (i.e., pre-training and ranking) for personalized search to strengthen data representations. To the best of our knowledge, this is the first time that the use of pre-training tasks for personalized search is investigated. (2) We use self-supervised learning to capture correlations between user behaviors at the pre-training stage, so as to learn better representations of user behaviors. (3) We adopt a contrastive sampling method to generate training pairs for self-supervised learning. Based on a contrastive learning framework, four self-supervised tasks are devised to pre-train the encoders used in the ranking task.

2 RELATED WORK

2.1 Personalized Search

Personalized search has been a research hotspot because it can improve the ranking quality of search engines effectively [4]. Traditionally, click-based features are widely studied due to its easy availability and reliability. Dou et al. [11] proposed P-click and G-click models to count the number of historical clicks on the same query from individual and group behaviors respectively, and fused the results of personalized ranking and original ranking to get the final results. Teevan et al. [31] also collected these click-based features to identify personal navigation for personalizing the results. Topic-based features extract the topic information of the document, thereby modeling which topics users are interested in. The ODP was a proper tool for representing the topic of documents and was widely used for user modeling [2, 26, 37]. In order to tackle its incomplete categories and huge labor cost, researches developed the latent topic space to learn the vectors of documents automatically [5, 13, 34, 36]. Later studies [3, 33, 37] used the learning to rank method to combine these features.

In recent years, deep learning has been applied to user modeling for personalized search, which is effective in exploring the potential interests of users [12, 19, 23, 40, 43–45]. An adaptive ranking model was devised in [28] for building dynamic user profiles. Li et al. [16] focused on in-session contextual ranking with semantic features. Recently, various network structures have appeared in personalized search. Recurrent Neural Network was used to model the sequential information of user interests [12]. Generative Adversarial Network was applied to sample high quality training data [17]. Reinforcement Learning was used to model the dynamic change of user search process [41]. In addition to building user profiles, Zhou et al. [44] and Yao et al. [40] proposed to use the context of history to learn the embedding of the current query. However, although all these existing approaches paid attention to the models for user modeling and personalized ranking, none of them has investigated the use of pre-training tasks for personalized search. In this paper, we propose a pre-training framework for personalized search, which is a new paradigm for enhancing data representations.

2.2 Pre-training for Information Retrieval

Self-supervised learning is the mainstream way of pre-training, which uses auxiliary tasks to mine its own supervised information from large-scale unsupervised data. With this constructed supervision information, the network can be pre-trained to learn valuable representations for downstream tasks. Some language models, such as ELMo [22] and BERT [10], have made impressive progress in natural language processing. They trained the representations of words or sentences on self-supervised tasks and improved the performance of downstream tasks. In the field of information retrieval, some self-supervised learning frameworks were applied to recommendation. Zhou et al. [42] used mutual information maximization to learn the correlations among attributes, items, and sequences. Xie et al. [38] proposed a contrastive learning framework to learn high-quality user representations. For search tasks, Chang et al. [6] designed several paragraph-level pre-training tasks to enhance the representations of queries and documents. Ma et al. [18] presented the model PROP, which constructed fake query-document pairs based on query likelihood model on large-scale corpus. The success of these studies shows that self-supervised learning is able to enhance data representations. In this paper, we attempt to enhance the query, document, and user representations in personalized search with a self-supervised learning framework.

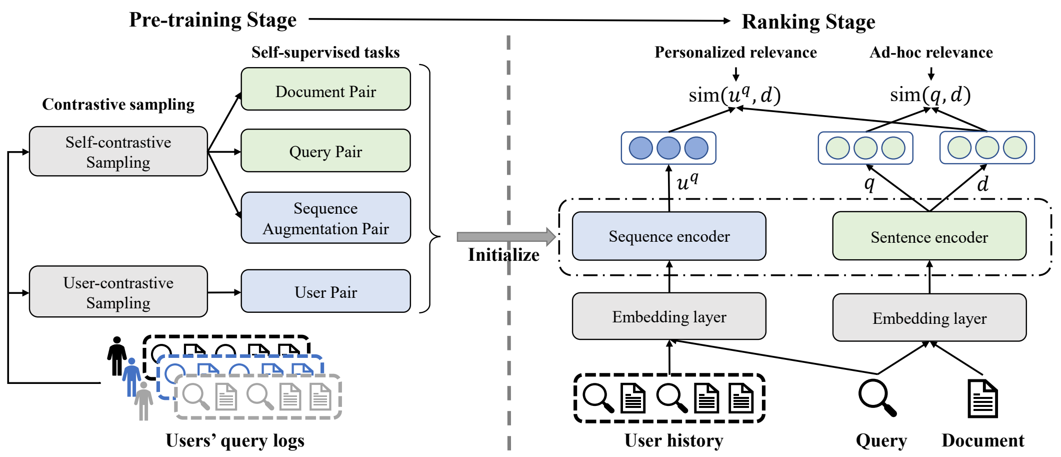


Figure 1: The architecture of our model PSSL, which is a two-stage training model. The first stage is pre-training with four self-supervised tasks that are generated by two angles of contrastive sampling. At the second stage, the sequence encoder and the sentence encoder are initialized with pre-trained parameters. The document ranking score is computed based on them.

3 PSSL: OUR PROPOSED MODEL

Personalized search has become an effective technique to improve user search experience by modeling user interests. As we stated in Section 1, existing personalization models suffer severely from data sparsity and rough data representations. In this paper, we propose a self-supervised learning framework to enhance data representations for personalized search. The architecture of our model PSSL is shown in Figure 1. First, we devise two angles of contrastive sampling to extract self-supervised signals from query logs, and use four tasks for pre-training. And then, the encoders used in the ranking model are initialized with pre-trained parameters. They contribute to high-quality data representations and better personalization.

3.1 Problem Definition

Suppose that there is a set of users, denoted as U. For each user u in the set, the query log H_u records the user's historical behaviors, including issuing a query and clicking on a document. We represent the user's query log as a sequence, i.e., $H_u = \{q_1, d_{1,1}, \cdots, q_{t-1}, d_{t-1,1}\}$, where t is the current timestamp, and $d_{i,j}$ refers to the j_{th} clicked document under the query q_i. Given the current query q, the candidate documents retrieved by the search engine are $\{d_1, d_2, \cdots\}$. Our personalized task is a re-ranking process, which needs to score each of the candidate documents based on the current query and the user's query log. We denote the score of the candidate document d as score$(d|q, H_u)$, which consists of two parts:

$$\text{score}(d|q, H_u) = \phi(\text{Pscore}(d|H_u, q), \text{Ascore}(d|q)), \tag{1}$$

where Pscore$(\cdot)$ computes personalized relevance regarding the user history, while Ascore$(\cdot)$ represents ad-hoc relevance between the current query and the document. The function $\phi(\cdot)$ is a multi-layer perceptron with $tanh(\cdot)$ as the activation function.

3.2 The Architecture of Ranking Model

As we stated in Section 1, the representation learning tasks in personalized search mainly conclude two dimensions. One is sentence-level encoding, which intends to learn the semantic representations of queries and documents. The other is sequence-level encoding, which focuses on learning user representations from their historical behavior sequences. Based on this consideration, we propose a sentence encoder and a sequence encoder to learn data representations.

The architecture of our ranking model is shown in the right half of Figure 1. First, after the embedding layer, the current query and the document are fed into the sentence encoder to learn the representations. And then, given the user history, the sequence encoder is applied to learn the query-aware user representation based on the current intent. Finally, by matching the document with the current query and the user representation, we can calculate the ranking score of the document and obtain the personalized ranking results. The details of each step are introduced as follows.

3.2.1 Embedding Layer. To learn the embedding of each query and document, we initialize a word embedding matrix with d-dimension, $M \in \mathbb{R}^{|W|*d}$, where $|W|$ is the vocabulary size. Given a query or a document, we first split it into words and then convert them into word embeddings. The embeddings of the user history are generated by converting each behavior in the sequence into vectors. For convenience, all the characters that appear in the following sections represent the vectors after the word embedding layer.

3.2.2 Sentence Encoder. Previous studies have pointed out that the queries issued by users are usually short and ambiguous [8, 27]. Moreover, documents retrieved by search engines often contain lots of worthless terms. Therefore, learning accurate semantic representations of queries and documents is integral to search tasks. We design a sentence encoder to tackle this problem. Given the current query q and the candidate document d, their sentence-level representations are computed as:

$$\bar{q} = \text{SenE}(q), \quad \bar{d} = \text{SenE}(d),$$

where the function SenE$(\cdot)$ is the sentence encoder. The detailed implementation will be introduced in Section 3.4.

3.2.3 Sequence Encoder. User representation learning is one of the most crucial tasks in personalized search. A high-quality user

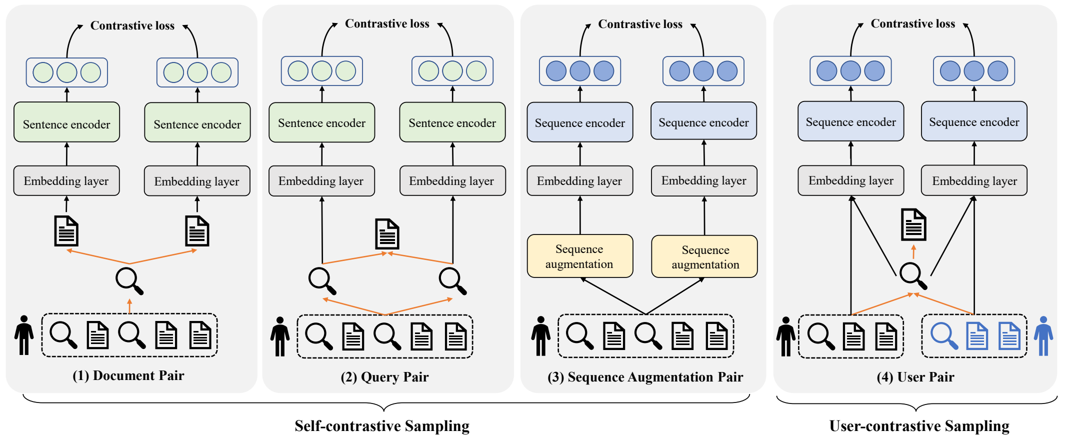


Figure 2: The overview of self-supervised tasks. Under two angles of contrastive sampling, we generate four types of self-supervised signals from query logs, including document pair, query pair, sequence augmentation pair, and user pair. Contrastive loss is used to optimize the parameters of two encoders. The red lines show specific search patterns in query logs.

representation can accurately reflect the user's preferences, thereby affecting the ranking of the final document list. In general, the user representation is learned from his query log, which can be regarded as a behavior sequence. This promotes us to devise a sequence encoder to model user interests based on historical behaviors.

Previous studies have shown that a dynamic user profile in response to the current query performs better than the static version [12, 28]. Inspired by this observation, we feed the user history and the current query together into the sequence encoder to learn the query-aware user representation. Formally, given the concatenated sequence of user's history H_u and current query q, the user representation is defined as:

$$u^q = \mathrm{SeqE}([H_u, q]),$$

where the function $\mathrm{SeqE}(\cdot)$ is the sequence encoder, which will be also described in Section 3.4. The user representation u^q will contribute to matching with candidate documents in the following.

3.2.4 Re-ranking. As shown in Eq. (1), the final score of candidate documents consists of two parts. For personalized relevance, we compute the similarity between the user representation u^q and the document representation $\bar{d}$:

$$\mathrm{Pscore}(d|H_u, q) = \mathrm{Sim}(u^q, \bar{d}),$$

where the function $\mathrm{Sim}(\cdot)$ is implemented by cosine similarity in this work. For ad-hoc relevance, we first take the similarity between the query representation and the document representation into account. Moreover, following the previous work [3, 12], we extract additional features to reveal the relevance between the query and the document, including click-based features, topic-based features, and several neural matching features. These features $f_{q,d}$ are fed into MLP to represent the relevance. Finally, the ad-hoc relevance consists of two parts that are combined with another MLP layer:

$$\mathrm{Ascore}(d|q) = \phi\left(\mathrm{Sim}\left(\bar{q}, \bar{d}\right), \phi(f_{q,d})\right),$$

where $\phi(\cdot)$ represents a multilayer perceptron. Two MLP layers here have different parameters that are learned during the training.

By re-ranking the results based on the final score, we obtain the personalized search results with respect to the user's individual interests. However, due to the shortcomings as we discussed in Section 1, the training of sentence encoder and sequence encoder is limited in data representations. To handle this problem, we apply the self-supervised learning framework to pre-train the two encoders with four auxiliary tasks.

3.3 Self-supervised Learning with Contrastive Sampling

Self-supervised learning provides us with sufficient training samples for learning data representations. It can mine the correlations between user behaviors to enhance the generalizability of the model. In this section, we will introduce how to extract self-supervised signals from query logs to pre-train the two encoders.

To design auxiliary tasks that are helpful for personalization, we design the learning objectives considering the characteristic of personalized search. Since that the personalization relies heavily on query logs, we attempt to mine specific search patterns in query logs to construct self-supervised signals. In fact, there are plenty of paired self-supervised information hidden in the query logs. For example, if a user issued “Online translation” and “Google Translate”, and clicked the same URL “https://translate.google.cn/” in the past, we can infer that these two queries reflect the same intent for this user. Therefore, bringing the representations of these two queries closer helps infer the query intent. Such specific search patterns can be transferred to documents and user's behavior sequences for constructing data pairs. Inspired by the contrastive learning framework [7, 14], we propose two ways of data sampling to generate paired data, i.e., self-contrastive sampling and user-contrastive sampling. The overview of self-supervised tasks is shown in Figure 2. Under two angles of contrastive sampling, four self-supervised tasks are devised to pre-train the encoders.

3.3.1 Self-contrastive Sampling. Self-contrastive sampling aims to extract data pairs with similar meanings from a single user's historical behavior sequence. Based on this method, we propose three auxiliary tasks for pre-training.

Document Pair. In the search process, a common scenario is that a user clicks on multiple documents in the returned results under a single query. This indicates that these clicked documents are in line with the user's query intent, and the information contained in these documents has a certain similarity. Based on this consideration, we design the task to close the representations of clicked documents under the same query.

For the user u, suppose that he clicks two documents d_i and d_j under the query q. After word embedding, we apply the sentence encoder to learn their document representations $\bar{d_i}$ and $\bar{d_j}$:

$$\bar{d_i} = \text{SenE}(d_i), \quad \bar{d_j} = \text{SenE}(d_j), \quad u \longrightarrow q \longrightarrow (d_i, d_j),$$

where the path $u \longrightarrow q \longrightarrow (d_i, d_j)$ means the user u issued the query q and clicked the documents d_i and d_j. The loss function is referred to the contrastive learning loss that maximizes the similarity of positive document pair. Following [7], considering a mini-batch of N pairs of documents, we treat $(\bar{d_i}, \bar{d_j})$ as the positive pair and treat other $2(N-1)$ documents within the same mini-batch as negative samples D^-. Formally, the loss function of this task $\mathcal{L}_{DP}$ for the document pair (d_i, d_j) can be defined as:

$$\mathcal{L}_{DP}(d_i, d_j) = -\log \frac{\exp\left(\text{Sim}(\bar{d_i}, \bar{d_j})\right)}{\exp\left(\text{Sim}(\bar{d_i}, \bar{d_j})\right) + \sum_{d^- \in D^-} \exp\left(\text{Sim}(\bar{d_i}, d^-)\right)},$$

where the function $\text{Sim}(\cdot)$ is implemented by cosine similarity in this paper. It can also be replaced by inner product.

Query Pair. As we discussed above, users may issue two similar queries to find the same document. This inspires us to mine such a search pattern and to close the representations of paired queries.

Suppose that for the user u, he clicked the document d under the query q_i and q_j respectively. These two queries can form a positive pair. The sentence encoder is applied to their embeddings to learn the query representations:

$$\bar{q_i} = \text{SenE}(q_i), \quad \bar{q_j} = \text{SenE}(q_j), \quad u \longrightarrow (q_i, q_j) \longrightarrow d.$$

We also use the contrastive learning framework to train the model. For N pairs of queries in a mini-batch, $(\bar{q_i}, \bar{q_j})$ is regarded as the positive pair, while the others are treated as negative queries Q^-. The query pair loss function $\mathcal{L}_{QP}$ for (q_i, q_j) is:

$$\mathcal{L}_{QP}(q_i, q_j) = -\log \frac{\exp\left(\text{Sim}(\bar{q_i}, \bar{q_j})\right)}{\exp\left(\text{Sim}(\bar{q_i}, \bar{q_j})\right) + \sum_{q^- \in Q^-} \exp\left(\text{Sim}(\bar{q_i}, q^-)\right)}.$$

The above two pre-training tasks concentrate on learning the parameters of sentence encoder to enhance data representations. To pre-train the sequence encoder, we design two tasks associated with user representations based on their behavior sequences.

Sequence Augmentation Pair. In personalized search, the main task is to model the user interests from his historical behavior sequence. In order to achieve better personalization, we hope that the sequence encoder can highlight the behaviors that best reflect the user's personality. To implement this idea, we apply the sequence augmentation to construct different views of the user's behavior sequence. The representations of augmented sequence pair should be closer than augmented sequences from other users.

Specifically, there are three sequence augmentation strategies for user's behavior sequence. (1) Behavior deleting. This way randomly deletes some behaviors from the sequence to enhance the generalizability of the model. We believe that the remaining behaviors in the sequence can also represent the user to a certain extent. (2) Behavior reorder. This strategy randomly swaps the positions of some behaviors. Although the order of behavior has a certain impact on the modeling of user interests, the user's long-stable preferences will not change and should be highlighted by the sequence encoder. (3) Session deleting. This method randomly deletes some sessions from the sequence. Users sometimes issue a series of queries in one session for a single information need. The remaining behaviors provide a local view of the user representation.

For the user history H_u, we apply two random augmentation strategies on the sequence, and get two augmented sequences $H_{u,i}$ and $H_{u,j}$. After the embedding layer, we use the sequence encoder to learn the representations of $H_{u,i}$ and $H_{u,j}$:

$$s_i = \text{SeqE}(H_{u,i}), \quad s_j = \text{SeqE}(H_{u,j}),$$

where s_i and s_j are representations of augmented sequences for the user. They respectively represent some of the user preferences, and form the positive pair for training. The augmented sequences from other users' query logs in the same mini-batch are regarded as the negative samples S^-. Similarly, the loss function of sequence augmentation pair $\mathcal{L}_{SAP}$ for the user history H_u is:

$$\mathcal{L}_{SAP}(H_u) = -\log \frac{\exp\left(\text{Sim}(s_i, s_j)\right)}{\exp\left(\text{Sim}(s_i, s_j)\right) + \sum_{s^- \in S^-} \exp\left(\text{Sim}(s_i, s^-)\right)}.$$

3.3.2 *User-contrastive Sampling.*

In addition to self-supervised sampling, we believe that the behavior sequences of different users can also form data pairs. The reason is that users with similar search behaviors tend to have similar interests. Therefore, we attempt to extract self-supervised signals from different users' query logs.

User Pair. For ambiguous queries, the retrieved documents often contain multiple topics. Different user groups tend to click on documents with different topics. For example, for the query "Apple", some users prefer "Apple company", while others focus on "Apple fruit". We believe that users with the same preferences have a certain similarity in user representations. To improve the effectiveness of training, we only choose the queries with ambiguity (click entropy greater than 1.0) to construct user pairs.

For two users u_i and u_j, if they both issued the query q and clicked on the same document d, these two users can be regarded as the positive pair facing the query q. Note that the sequences of user history H_{u_i} and H_{u_j} only contain the behaviors before the query q. The sequence encoder is used to learn their query-aware user representations u_i^q and u_j^q:

$$u_i^q = \text{SeqE}([H_{u_i}, q]), \quad u_j^q = \text{SeqE}([H_{u_j}, q]), \quad (u_i, u_j) \longrightarrow q \longrightarrow d.$$

Similar to the previous tasks, the set of negative samples U^- consists of representations of other users in the mini-batch. The user pair loss function $\mathcal{L}_{UP}$ for the user u_i and u_j is:

$$\mathcal{L}_{UP}(H_{u_i}, H_{u_j}) = -\log \frac{\exp\left(\text{Sim}(u_i^q, u_j^q)\right)}{\exp\left(\text{Sim}(u_i^q, u_j^q)\right) + \sum_{u^- \in U^-} \exp\left(\text{Sim}(u_i^q, u^-)\right)}.$$

3.4 The Implementation Details of Encoders

In this section, we will introduce the implementation details of the sentence encoder and the sequence encoder. The basic structure is based on the Transformer encoder [32] to model the contextual information following [44].

Implementation of Sentence Encoder. To tackle the query ambiguity and document noise at word level, we attempt to combine contextual information of surrounding words to represent sentences. The transformer encoder is applied to learn the context-aware sentence representations, which is widely used in various fields. Formally, for a query or a document, suppose it consists of n words, i.e., $q = \{w_1, w_2, \cdots, w_n\}$. The sentence encoder is:

$$\mathrm{SenE}(q) = \mathrm{Transformer}_w^{\mathrm{sum}}([w_1, w_2, \cdots, w_n]),$$

where the function $\mathrm{Transformer}_w^{\mathrm{sum}}$ means the sum of outputs of word-level transformer layer. The final output is the sentence representation which considers the contextual information.

Implementation of Sequence Encoder. Existing studies have pointed out that long-term history and short-term user history play different roles in personalized search [12, 36]. Long-term history generally contains user behaviors before the current session, and often reflects the user's long-stable interests. Short-term history refers to the user's past interactions in the current session, which shows the user's recent information needs. Based on this consideration, we use a hierarchical transformer structure to learn user representations following [44].

Given a user history H_u, we divide it into long-term and short-term history H_u^l and H_u^s, which are fed into two transformers to model user preferences from different views. To learn the user representation based on the current query q, we concatenate it with short-term history to model the user's information need of the current session. Moreover, a '[User]' token is added at the end of the sequence to represent the summarized user representation. The sequence encoder consists of a short-term transformer $\mathrm{Transformer}_s(\cdot)$ and a long-term transformer $\mathrm{Transformer}_l(\cdot)$:

$$\mathrm{SeqE}(H_u, q) = \mathrm{Transformer}_l^{\mathrm{last}}\left(\left[H_u^l, u^{q,s}\right]\right),$$
$$u^{q,s} = \mathrm{Transformer}_s^{\mathrm{last}}\left(\left[H_u^s, q, [\mathrm{User}]\right]\right),$$

where the superscript *last* means taking the last position as the output. Finally, the output represents the user representation which contributes to the personalization of search results.

3.5 Training and Optimization

The training process of PSSL includes two stages: pre-training and fine-tuning. At the first stage, we optimize the loss of the four self-supervised tasks ($\mathcal{L}_{DP}$, $\mathcal{L}_{QP}$, $\mathcal{L}_{SAP}$, and $\mathcal{L}_{UP}$) to pre-train the sentence encoder and sequence encoder. At the second stage, two encoders are initialized by the pre-trained parameters, and we use the ranking task to fine-tune them and train the whole network. The ranking loss is computed by cross entropy in a pairwise way:

$$\mathcal{L}_{Rank}(d_i, d_j) = -\left(\overline{p}_{ij}\log(p_{ij}) + \overline{p}_{ji}\log(p_{ji})\right),$$

where d_i is the positive sample and d_j is the negative sample, p_{ij} and $\overline{p}_{ij}$ represent the predicted probability and the real probability that d_i is more relevant than d_j. The p_{ij} is computed by $p(d_i|H_u, q) - p(d_j|H_u, q)$ with sigmoid normalization.

Table 1: Basic statistics of the datasets.

Dataset	AOL	Commercial
# Users	110,439	33,204
# Queries	736,454	267,479
# Sessions	279,930	97,858
Average query length	2.87	3.25
Average #click per query	1.11	1.19

4 EXPERIMENTAL SETUP

4.1 Dataset

We conduct our experiments on AOL search logs [21] and a commercial dataset. The basic statistics are shown in Table 1.

AOL dataset: The AOL search log is a public dataset, which contains users' click-through data from 1st March, 2006 to 31st May, 2006. Following the previous work [1], we remove all non-alphanumeric characters from the queries, and regard the document title as the content to compute the relevance. Since the dataset only records clicked documents without the returned document lists, BM25 algorithm [25] is used to select candidate documents from the top results. To identify a session, similarity between two consecutive queries is considered with threshold of 0.5. Based on the above operations, each piece of data includes an anonymous user ID, a session ID, a query, the query issued time, a document URL, the ranking position, the document title, and a click tag. Since the personalized search requires historical information to build the basic user profile, we use the first five weeks data as background set. The last eight weeks data is regarded as the experimental set, which is further divided into training set, validation set and test set in a 4:1:1 ratio. To ensure that the data is sufficient for personalization, we remove users whose background set or training set is empty.

Commercial dataset: This dataset records the query log in January and February, 2013 from a large commercial search engine. There are some differences between this dataset and AOL in data processing. First, the candidate documents are collected directly from the document list returned by the search engine. The original ranking quality is much higher than BM25. Second, we crawl the content corresponding to the URL to represent the document, instead of just the title. This will provide us with more accurate document representations. Third, since this dataset records the click dwell time of each URL, we treat the URL whose click dwell time is greater than 30s as the satisfied document. We keep the same ratio as AOL to collect the background set and the experimental set.

4.2 Baselines

To compare the performance of our model and other neural ranking models, we select several typical ad-hoc search models and personalized search models as baselines. They are:

KNRM [39]. It is a neural ranking model based on kernel-pooling for ad-hoc search. Multi-level soft matching features are extracted from the word similarity matrix for ranking.

Conv-KNRM [9]. It adds a convolutional layer on the KNRM to model n-gram soft matches. Contextual information of surrounding words are considered to improve matching accuracy.

BERT [24]. This model applies the pre-trained BERT model to query-document matching task. The concatenated query-document

Table 2: The results of all models on two datasets. The percentage is based on the SOTA baseline. '†' indicates the model outperforms all baselines significantly with paired t-test at $p < 0.05$ level. Best results are denoted in bold.

Model	AOL dataset						Commercial dataset							
	MAP		MRR		P@1		MAP		MRR		P@1		P-improve	
Ad-hoc search baselines														
Ori.	.2504	-64.9%	.2596	-64.2%	.1534	-75.6%	.7399	-10.2%	.7506	-10.0%	.6162	-15.6%	-	-
KNRM	.4291	-39.8%	.4391	-39.5%	.2704	-56.9%	.4916	-40.3%	.5001	-40.1%	.2849	-61.0%	.0655	-75.3%
Conv-KNRM	.4738	-33.5%	.4849	-33.2%	.3266	-48.6%	.5872	-28.7%	.5977	-28.4%	.4188	-42.7%	.1422	-46.5%
BERT	.5033	-29.4%	.5135	-29.3%	.3552	-43.4%	.6232	-24.4%	.6326	-24.2%	.4475	-38.7%	.1778	-33.1%
Personalized search baselines														
HRNN	.5423	-23.9%	.5545	-23.6%	.4854	-22.7%	.8065	-2.1%	.8191	-1.8%	.7127	-2.4%	.2404	-9.5%
PSGAN	.5480	-23.1%	.5601	-22.8%	.4892	-22.1%	.8135	-1.3%	.8234	-1.3%	.7174	-1.8%	.2489	-6.3%
RPMN	.5926	-16.9%	.6049	-16.7%	.5322	-15.2%	.8238	-	.8342	-	.7305	-	.2656	-
HTPS	.7091	-0.5%	.7251	-0.1%	.6268	-0.1%	.8224	-0.2%	.8324	-0.2%	.7286	-0.3%	.2552	-3.9%
PEPS	.7127	-	.7258	-	.6279	-	.8221	-0.2%	.8321	-0.3%	.7251	-0.7%	.2545	-4.2%
Our method														
PSSL	**.7359**†	+3.3%	**.7484**†	+3.1%	**.6431**†	+2.4%	**.8301**†	+0.8%	**.8398**†	+0.7%	**.7338**†	+0.5%	**.2688**†	+1.2%

sequence is fed into the pre-trained BERT model. The last layer's representation of '[CLS]' token is regarded as the matching features. The BERT model are fine-tuned during the training.

HRNN [12]. For personalized search, this work models the sequential information of query logs and learns dynamic user profiles based on the current query. Hierarchical recurrent neural networks with query-aware attention is used to implement this idea.

PSGAN [17]. This study concentrates on data augmentation based on generative adversarial network for personalized search. It aims to extract valid training data from limited and noisy click data. Considering the cost of training, we take the discriminator in the document selection based model as the baseline.

RPMN [45]. This is a memory network-based personalized search model, which attempts to identify potential re-finding behaviors in personalized search. It devises three external memories to cover two types of re-finding behavior.

PEPS [40]. This model trains personal word embeddings for each user based on his historical data, and abandons the construction of user profiles. Personal and global word embeddings are both considered for better data representations.

HTPS [44]. This is a personalized search framework based on hierarchical transformer. Transformer encoder is first used to encode history as contextual information to disambiguate the query.

4.3 Implementation Details

For our proposed model PSSL[1], the word embedding matrix are initialized by the word2vec [20] model following [12, 17]. It will be fixed in the pre-training phase, and be fine-tuned during the training of ranking task. We conducted multiple experiments to select the parameters of the model. Finally, the dimension of the word embedding is 100. The hidden size of transformer is 512. The number of attention heads in transformer is 6. The number of transformer layers is 6. The number of MLP hidden units is 128. The learning rate of the pre-training task and the ranking task are set to $1e^{-3}$ and $3e^{-4}$ respectively. At the pre-training stage, for sequence augmentation strategies, we change 50% of user behaviors in the sequence. For four tasks (DP, QP, SAP, and UP), we sample about 163k, 293k, 728k, 128k contrastive pairs on the AOL dataset and 52k, 103k, 264k, 21k contrastive pairs on the commercial dataset. The weights for the four losses ($\mathcal{L}_{DP}$, $\mathcal{L}_{QP}$, $\mathcal{L}_{SAP}$, and $\mathcal{L}_{UP}$) are set as 0.5, 0.5, 1.0, 0.2.

[1]The code of the model is available on https://github.com/smallporridge/PSSL.

4.4 Evaluation Metrics

Since the AOL dataset does not contain user click dwell time, we simply label the clicked documents as relevant, while the satisfied documents in the commercial dataset are regarded as relevant. To evaluate the model performance, we employ mean average precise(MAP), mean reciprocal rank (MRR), and precision@1 (P@1) to measure the ranking quality. However, the above metrics are somewhat problematic due to the position bias [15] in re-ranking tasks. To measure the ranking results in a more objective manner, we apply another metric called P-improve to evaluate reliable improvements on the inverse document pair following previous works [12, 17]. Since the candidate documents of AOL dataset are not presented to users, we only use this metric on the commercial dataset which suffers from the position bias.

5 RESULTS AND ANALYSIS

5.1 Overall Performance Comparison

The results of different models on the two datasets are shown in Table 2. It can be observed that:

(1) Our method vs. baselines. Our proposed model PSSL outperforms all baseline models on both datasets. Compared with the best baseline model, PSSL shows significant improvements in all evaluation metrics with paired t-test at $p < 0.05$ level. Specifically, our model improves the ranking quality by 3.3% on MAP on the AOL dataset compared with PEPS, while outperforms RPMN by 0.8% on the commercial dataset. This indicates that the self-supervised learning framework helps the model learn better data representations and further improves the personalization. The improvement on the two datasets also verifies the generalizability of our model.

(2) Personalized search vs. ad-hoc search. The ad-hoc search concentrates on improving matching accuracy between the query

Table 3: Performance of ablation studies on self-supervised tasks of the PSSL model.

Model	AOL dataset						Commercial dataset							
	MAP		MRR		P@1		MAP		MRR		P@1		P-improve	
Tasks of sentence encoder														
w/o. DP	.7296	-0.9%	.7434	-0.7%	.6389	-0.7%	.8296	-0.1%	.8390	-0.1%	.7331	-0.1%	.2678	-0.4%
w/o. QP	.7252	-1.5%	.7398	-1.1%	.6368	-1.0%	.8288	-0.2%	.8381	-0.2%	.7325	-0.2%	.2662	-1.0%
w/o. DP+QP	.7200	-2.2%	.7357	-1.7%	.6341	-1.4%	.8287	-0.2%	.8381	-0.2%	.7324	-0.2%	.2658	-1.1%
Tasks of sequence encoder														
w/o. SAP	.7212	-2.0%	.7362	-1.6%	.6343	-1.4%	.8242	-0.7%	.8348	-0.6%	.7288	-0.7%	.2584	-3.9%
w/o. UP	.7288	-1.0%	.7430	-0.7%	.6382	-0.7%	.8270	-0.4%	.8366	-0.4%	.7306	-0.4%	.2632	-2.1%
w/o. SAP+UP	.7170	-2.6%	.7330	-2.1%	.6316	-1.8%	.8222	-1.0%	.8321	-1.0%	.7272	-0.9%	.2548	-5.2%
PSSL	.7359	-	.7484	-	.6431	-	.8301	-	.8398	-	.7338	-	.2688	-

and the document, while the personalized search focuses on how to model user interests based on historical behaviors. All neural personalized search models outperform ad-hoc baselines significantly, while the improvement on the metric P@1 is more obvious. This reflects the personalization models perform well on modeling user's re-finding behavior. Our model PSSL combines the advantages of these two types of search models, using a sentence encoder for query-document matching and a sequence encoder for user representation learning. With a self-supervised learning framework, this is proven to be effective in improving the ranking quality.

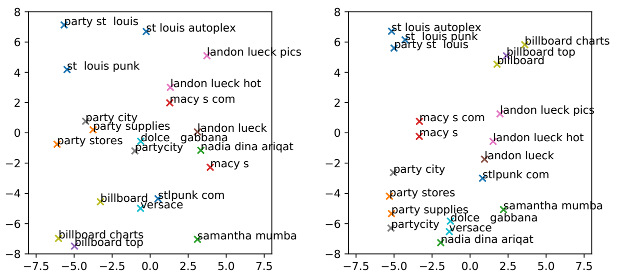
(3) AOL dataset vs. commercial dataset. The commercial dataset has relatively higher original ranking quality than AOL, which leads to the results that ad-hoc search baselines perform worse than the original ranking on the commercial dataset. The model HTPS and PEPS, which incorporate interactive matching features between the query and the document, make a significant improvement on the AOL dataset, but they underperform the RPMN on the commercial dataset. This shows that the AOL dataset tests the model on modeling user interests and text matching at the same time, while the commercial dataset focuses more on testing the personalization capabilities of the model. Our model performs well on both datasets, which further proves the robustness of PSSL model.

In summary, the results indicate that **self-supervised learning with contrastive sampling for personalized search is conducive to refine data representations and promote search results personalization.** To test the model in more detail, we conduct several supplementary experiments: ablation studies, effect of self-supervised learning, and performance on different query sets.

5.2 Ablation Studies

To verify the necessity of each of our self-supervised tasks, we conduct ablation experiments on two datasets for the whole model PSSL. Specifically, we explore the role of each self-supervised task on the two encoders respectively, including the tasks of document pair (DP), query pair (QP), sequence augmentation pair (SAP), and user pair (UP). We also remove the pre-training of sentence encoder (DP+QP) or sequence encoder (SAP+UP) to observe the results.

As shown in Table 3, the removal of each self-supervised task will damage the results on all evaluation metrics. Concretely, deleting the task of SAP causes the most obvious impact on performance on both datasets. This indicates that our sequence augmentation strategies help the sequence encoder model the user representations more accurately. Meanwhile, the task of UP also makes some



(a) Initialized **(b) Self-supervised**

Figure 3: The query distribution of the user #104.

contributions to the results, which shows that closing the distance between similar users is useful for user modeling. Additionally, we find that removing the pre-training of sentence encoder causes a severe drop on the AOL dataset, while it has little effect on the commercial dataset. A possible reason is that the pre-training of sentence encoder is more helpful for computing ad-hoc relevance. For the AOL dataset, the ad-hoc relevance is more useful due to the poor original ranking quality. But for the commercial dataset, its high-quality original ranking results have provided effective ad-hoc relevance. It can be seen that the task of QP is more important than DP for personalizing the results. This indicates that the queries can provide more additional personalized information for the model.

5.3 Effect of Self-supervised Learning

In order to explore the impact of self-supervised tasks on data representations in more detail, we visualize the quality of the sentence encoder and sequence encoder respectively.

Quality of Sentence Encoder. The self-supervised tasks for sentence encoder mainly enhance the representations of queries and documents, so as to close the distance between similar data. In order to verify the enhancement of data representations by the self-supervised tasks, we compare the distribution of initialized and self-supervised query vectors. Specifically, we randomly select a user from the AOL dataset and map his high-dimensional query vectors to a two-dimensional space through PCA. Queries containing the same clicked document are set to the same color.

The results are shown in Figure 3. We find the initialized query distribution is more dispersive, and self-supervised learning makes the distance of queries with the same color closer. For instance, the

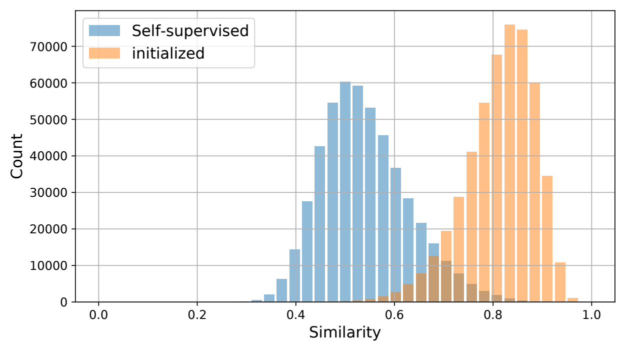


Figure 4: Distribution of similarity between users.

queries with red color "macy s" and "macy s com" have a certain distance in the initialized distribution, but are obviously closer after the self-supervised learning. This indicates that although these two queries have some differences in words, they tend to reflect the same query intent for the user in search scenario. Additionally, the self-supervised learning intends to map the queries into multiple groups. The distance within the group is close, while the boundary between the groups is relatively clear. This shows our model can not only close the distance between similar queries, but also widen the distance between different groups.

Quality of Sequence Encoder. The sequence encoder is designed for learning accurate user representations, which is a critical indicator to distinguish users. In order to observe the impact of self-supervised tasks on the sequence encoder, we compare the difference between initialized and self-supervised user representations. Specifically, we randomly select 1000 users from the AOL dataset and calculate the similarity between the representations of every two users. We divide the bar at 0.025 intervals, and count the number of user pairs that fall in each range.

From Figure 4, it can be seen that the similarity between user representations generally obeys normal distribution. The initialized user representations show higher consistency, which means that the model cannot effectively identify the differences between users. With the self-supervised tasks, the average similarity between users becomes smaller. This indicates that the differences between users are magnified. Another interesting finding is that for the self-supervised user representations, the distribution is not completely symmetrical. Ranges with greater similarity contain more user pairs. This may benefit from the self-supervised task of UP, which aims to close the distance between similar users.

5.4 Performance on Different Query Sets

In the search process, according to the different search purposes, the user's queries can be classified as the navigational query and the informational query. For navigational queries, different users tend to have the same query intent. The informational queries are usually ambiguous and have multiple meanings. To measure the query ambiguity, we compute the click entropy and set 1.0 as the threshold to divide the queries into two subsets. We choose the best baseline model for comparison. Additionally, we remove the pre-training of sentence encoder (w/o. SenE) or sequence encoder (w/o. SeqE) for detailed analysis. We use the improvement of MAP over original ranking to show the performance.

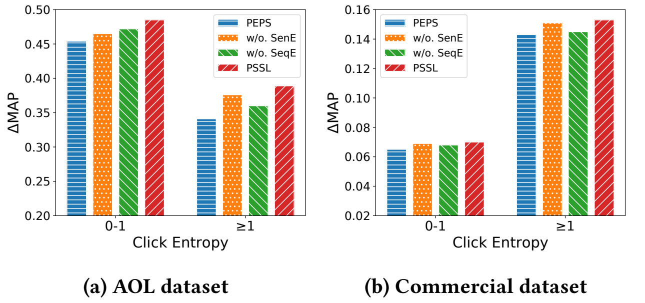


Figure 5: The results on queries with different click entropies with the threshold at 1.0.

As shown in Figure 5, all models perform better on navigational queries for the AOL dataset, but this is inconsistent with the results on the commercial dataset. A possible reason is that the commercial dataset has a high-quality original ranking, which has little room for improvement on navigational queries. Our proposed model PSSL outperforms PEPS on both query sets, especially on the queries with larger click entropy. This indicates that our model is able to learn high-quality user representations when facing ambiguous queries. Specifically, removing the pre-training of sequence encoder causes severe decline on informational queries. This shows that the pre-trained sequence encoder contributes to modeling user interests more. For sentence encoder, the self-supervised tasks show effectiveness on the AOL dataset, but the contribution on the commercial dataset is limited. This is in line with the characteristics of the two datasets. When the dataset requires more attention on ad-hoc relevance, our pre-trained sentence encoder can provide more accurate query-document matching.

6 CONCLUSION

In this paper, we proposed a self-supervised learning framework for personalized search to enhance data representations. First, we presented a ranking model which consists of a sentence encoder and a sequence encoder. Next, we designed two angles of contrastive sampling methods to generate paired self-supervised data from users' query logs. Four auxiliary tasks were devised to pre-train the two encoders for personalized search. Endowed with the benefit of pre-trained parameters, we could get better data representations to improve the personalized results and the generalizability of the model. Experimental results confirmed the effectiveness and robustness of our proposed two-stage training framework.

ACKNOWLEDGMENTS

Zhicheng Dou is the corresponding author. This work was supported by National Natural Science Foundation of China (No. 61872370 and No. 61832017), Beijing Outstanding Young Scientist Program (No. BJJWZYJH012019100020098), Shandong Provincial Natural Science Foundation (No. ZR2019ZD06), and Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. We also wish to acknowledge the support provided and contribution made by Public Policy and Decision-making Research Lab of Renmin University of China.

REFERENCES

[1] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. 2018. Multi-task learning for document ranking and query suggestion. (2018).

[2] Paul N Bennett, Krysta Svore, and Susan T Dumais. 2010. Classification-enhanced ranking. In *Proceedings of the WWW'2010*. ACM, 111–120.

[3] 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 SIGIR'2012*. ACM, 185–194.

[4] Fei Cai, Shangsong Liang, and Maarten De Rijke. 2014. Personalized document re-ranking based on bayesian probabilistic matrix factorization. In *Proceedings of the SIGIR'2014*. ACM, 835–838.

[5] Mark J. Carman, Fabio Crestani, Morgan Harvey, and Mark Baillie. 2010. Towards query log based personalization using topic models. In *Proceedings of the CIKM'2010*. 1849–1852.

[6] Wei-Cheng Chang, Felix X. Yu, Yin-Wen Chang, Yiming Yang, and Sanjiv Kumar. 2020. Pre-training Tasks for Embedding-based Large-scale Retrieval. In *ICLR*. OpenReview.net.

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML (Proceedings of Machine Learning Research, Vol. 119)*. PMLR, 1597–1607.

[8] Steve Cronen-Townsend and W Bruce Croft. 2002. Quantifying query ambiguity. In *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc., 104–109.

[9] Zhuyun Dai, Chenyan Xiong, Jamie Callan, and Zhiyuan Liu. 2018. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In *Proceedings of the eleventh ACM international conference on web search and data mining*. ACM, 126–134.

[10] 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).

[11] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *WWW'2007*, Carey L. Williamson, Mary Ellen Zurko, Peter F. Patel-Schneider, and Prashant J. Shenoy (Eds.). ACM, 581–590.

[12] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. 2018. Personalizing Search Results Using Hierarchical RNN with Query-aware Attention. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM '18)*.

[13] Morgan Harvey, Fabio Crestani, and Mark J Carman. 2013. Building user profiles from topic models for personalised search. In *CIKM'2013*. ACM, 2309–2314.

[14] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. 2020. Momentum Contrast for Unsupervised Visual Representation Learning. In *CVPR*. IEEE, 9726–9735.

[15] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. 2005. Accurately interpreting clickthrough data as implicit feedback. In *SIGIR'2005*. 154–161.

[16] Xiujun Li, Chenlei Guo, Wei Chu, Ye-Yi Wang, and Jude Shavlik. 2014. Deep learning powered in-session contextual ranking using clickthrough data. In *NIPS'2014*.

[17] Shuqi Lu, Zhicheng Dou, Xu Jun, Jian-Yun Nie, and Ji-Rong Wen. 2019. PSGAN: A Minimax Game for Personalized Search with Limited and Noisy Click Data. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19)*.

[18] Xinyu Ma, Jiafeng Guo, Ruqing Zhang, Yixing Fan, Xiang Ji, and Xueqi Cheng. 2021. PROP: Pre-training with Representative Words Prediction for Ad-hoc Retrieval. In *WSDM*. ACM, 283–291.

[19] Zhengyi Ma, Zhicheng Dou, Guanyue Bian, and Ji-Rong Wen. 2020. PSTIE: Time Information Enhanced Personalized Search. In *CIKM*. ACM, 1075–1084.

[20] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168* (2013).

[21] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search.. In *InfoScale*, Vol. 152. 1.

[22] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *NAACL-HLT*. Association for Computational Linguistics, 2227–2237.

[23] Hongjin Qian, Xiaohe Li, Hanxun Zhong, Yu Guo, Yueyuan Ma, Yutao Zhu, Zhanliang Liu, Zhicheng Dou, and Ji-Rong Wen. 2021. Pchatbot: A Large-Scale Dataset for Personalized Chatbot. In *Proceedings of the SIGIR 2021*. ACM. https://doi.org/10.1145/3404835.3463239

[24] Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the Behaviors of BERT in Ranking. *arXiv preprint arXiv:1904.07531* (2019).

[25] Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval* 3, 4 (2009), 333–389.

[26] Ahu Sieg, Bamshad Mobasher, and Robin Burke. 2007. Web search personalization with ontological user profiles. In *CIKM'2007*. ACM, 525–534.

[27] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. 1999. Analysis of a very large web search engine query log. In *ACm SIGIR Forum*, Vol. 33. ACM, 6–12.

[28] Yang Song, Hongning Wang, and Xiaodong He. 2014. Adapting deep ranknet for personalized search. In *WSDM'2014*. ACM, 83–92.

[29] Zhan Su, Zhicheng Dou, Yutao Zhu, Xubo Qin, and Ji-Rong Wen. 2021. Modeling Intent Graph for Search Result Diversification. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai (Eds.). ACM, 736–746. https://doi.org/10.1145/3404835.3462872

[30] Jaime Teevan, Susan T Dumais, and Daniel J Liebling. 2008. To personalize or not to personalize: modeling queries with variation in user intent. In *SIGIR'2008*. ACM, 163–170.

[31] Jaime Teevan, Daniel J Liebling, and Gayathri Ravichandran Geetha. 2011. Understanding and predicting personal navigation. In *WSDM'2011*. ACM, 85–94.

[32] 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.

[33] Maksims Volkovs. 2015. Context models for web search personalization. *arXiv preprint arXiv:1502.00527* (2015).

[34] Thanh Vu, Dat Quoc Nguyen, Mark Johnson, Dawei Song, and Alistair Willis. 2017. Search personalization with embeddings. In *ECIR'2017*. Springer, 598–604.

[35] Thanh Vu, Dawei Song, Alistair Willis, Son Ngoc Tran, and Jingfei Li. 2014. Improving search personalisation with dynamic group formation. In *SIGIR'2014*. 951–954.

[36] Thanh Vu, Alistair Willis, Son N Tran, and Dawei Song. 2015. Temporal latent topic user profiles for search personalisation. In *ECIR'2015*. Springer, 605–616.

[37] Ryen W White, Wei Chu, Ahmed Hassan, Xiaodong He, Yang Song, and Hongning Wang. 2013. Enhancing personalized search by mining and modeling task behavior. In *WWW'2013*. ACM, 1411–1420.

[38] Xu Xie, Fei Sun, Zhaoyang Liu, Jinyang Gao, Bolin Ding, and Bin Cui. 2020. Contrastive Pre-training for Sequential Recommendation. *CoRR* abs/2010.14395 (2020).

[39] Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. 2017. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR conference on research and development in information retrieval*. ACM, 55–64.

[40] Jing Yao, Zhicheng Dou, and Ji-Rong Wen. 2020. Employing Personal Word Embeddings for Personalized Search. In *SIGIR*. ACM, 1359–1368.

[41] Jing Yao, Zhicheng Dou, Jun Xu, and Ji-Rong Wen. 2020. RLPer: A Reinforcement Learning Model for Personalized Search. In *WWW*. ACM / IW3C2, 2298–2308.

[42] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *CIKM*. ACM, 1893–1902.

[43] Yujia Zhou, Zhicheng Dou, Bingzheng Wei, Ruobing Xie, and Ji-Rong Wen. 2021. Group based Personalized Search by Integrating Search Behaviour and Friend Network. In *SIGIR*. ACM, 92–101.

[44] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2020. Encoding History with Context-aware Representation Learning for Personalized Search. In *SIGIR*. ACM, 1111–1120.

[45] Yujia Zhou, Zhicheng Dou, and Ji-Rong Wen. 2020. Enhancing Re-finding Behavior with External Memories for Personalized Search. In *WSDM*. ACM, 789–797.

[46] Yutao Zhu, Kun Zhou, Jian-Yun Nie, Shengchao Liu, and Zhicheng Dou. 2021. Neural Sentence Ordering Based on Constraint Graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 14656–14664. https://ojs.aaai.org/index.php/AAAI/article/view/17722