PSGAN: A Minimax Game for Personalized Search with Limited and Noisy Click Data

Shuqi Lu[1], Zhicheng Dou[1,2], Jun Xu[1,2], Jian-Yun Nie[4] and Ji-Rong Wen[1,2,3]
[1]School of Information, Renmin University of China, [4]DIRO, Université de Montréal
[2]Beijing Key Laboratory of Big Data Management and Analysis Methods
[3]Key Laboratory of Data Engineering and Knowledge Engineering, MOE
lusq@ruc.edu.cn,dou@ruc.edu.cn,junxu@ruc.edu.cn,nie@iro.umontreal.ca,jirong.wen@gmail.com

ABSTRACT

Personalized search aims to adapt document ranking to user's personal interests. Traditionally, this is done by extracting click and topical features from historical data in order to construct a user profile. In recent years, deep learning has been successfully used in personalized search due to its ability of automatic feature learning. However, the small amount of noisy personal data poses challenges to deep learning models to learn the personalized classification boundary between relevant and irrelevant results. In this paper, we propose PSGAN, a Generative Adversarial Network (GAN) framework for personalized search. By means of adversarial training, we enforce the model to pay more attention to training data that are difficult to distinguish. We use the discriminator to evaluate personalized relevance of documents and use the generator to learn the distribution of relevant documents. Two alternative ways to construct the generator in the framework are tested: based on the current query or based on a set of generated queries. Experiments on data from a commercial search engine show that our models can yield significant improvements over state-of-the-art models.

KEYWORDS

personalized web search; generative adversarial network;

ACM Reference Format:
Shuqi Lu, Zhicheng Dou, Jun Xu, Jian-Yun Nie, and Ji-Rong Wen. 2019. PSGAN: A Minimax Game for Personalized Search with Limitedand Noisy Click Data. In SIGIR'19: The 42st International ACM SIGIR Conference on Research & Development in Information Retrieval, July 21-25, 2019, Paris, France. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3331184.3331218

1 INTRODUCTION

Traditional search engines employ the one-size-fits-all strategy. They use the same ranking function for a query for any user. It is known that this strategy cannot cope with the different search information needs of users behind the same query. Personalized search offers a possible solution to the problem. Most existing personalized search approaches extract click and topical features from users' search history and calculate document relevance according to both the query and the induced user interests [1, 5, 8, 11, 26, 29, 32, 33, 35]. However, the features are usually designed manually. It is difficult to expect that these features have a complete coverage of the important factors. Deep learning offers a new alternative to personalized search [9, 16, 27]. Compared with the traditional methods, deep learning models can automatically learn the representations of documents, user profiles, and other relevant features from training data without manual design and extraction. They could also cover a wider range of features.

However, training data is a critical issue for deep learning methods, which involve a large number of parameters and need a considerable amount of training data. This is a very challenging issue for personalized search because a personalization model heavily relies on user's personal data while the search history of a specific user is always limited. There are only a few clicks on each search. In addition, the available data is noisy. For example, if the user has clicked on a search result and not on another, it is usually assumed that the former is preferred to the latter, which may not always be true. This is because the user may click on some of the relevant documents only, and she may also click on irrelevant documents. Blindly using all such data may lead to a wrong user profile. In such a context, it is important to select appropriate training data that bring the most useful information. Even among the true preferences, some are deemed more useful than others. For example, assuming that a user has a historical query "JAVA Language", then for the current query on "JAVA", the preference of a document on "Java IDE" over another one on "Java island" does not provide much additional information about the user's interests because the difference is easy to make whatever the user profile is. However, a preference of "JAVA IDE" over "JAVA book" is more difficult to detect, and may help detect the subtle preference of the user on the topic. This is the type of preference that can help the most. In this paper, our goal is to devise a method to select the latter type of preference as training data.

Inspired by IRGAN [34], our method is based on Generative Adversarial Network [10], which contains a generator and a discriminator. Through the minimax game of adversarial training, generator tries to generate high-quality negative examples to confuse discriminator, while discriminator provides reward for generator in order to help generator adjust the data distribution. However, due to the discreteness of text data, it is not possible to generate a free text vector as a negative sample. Following IRGAN [34], we sample examples from the negative data space (e.g., unclicked document or unlabelled document) as the generated examples.

SIGIR '19, July 21–25, 2019, Paris, France
© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-6172-9/19/07…$15.00
https://doi.org/10.1145/3331184.3331218

We design a general framework, named PSGAN, for personalized search, which generalizes the above idea based on GAN. We propose two implementations of the general model. In the first one, we borrow the idea of IRGAN [34] for personalization. We let the generator directly select a negative sample from the document candidates according to the relevance distribution for the current query and user interests. We call it *document selection based GAN model for personalized search.* In the second method, we first generate related queries that are consistent with the user's intent (through historical information) and the current query, and then calculate the relevance of the document through the generated queries. We hope that document relevance can be better estimated through the generated queries because of enriched information. We call this model *query generation based GAN model for personalized search.*

We compare these two implementation alternatives with the query log data from a commercial search engine. Experimental results show that adversarial training can effectively improve the quality of search personalization, and can yield significant improvements over state-of-the-art models. In addition, the second model can select better negative examples than the first model, leading to a better personalization.

The main contribution of this paper is threefold: first, this is the first attempt to apply GAN to personalized search; second, we enhanced the training data and improved the training effect by using the Generative Adversarial Network; third, we put forward a method of generating the queries consistent with user intent in order to determine better document samples. Experimental results on a real search engine log confirm the effectiveness of our method.

The rest of paper is organized as follows. We introduce related work in Section 2. Our GAN-based personalized search model is presented in Section 3. We describe experimental results Section 4, then conclude our work in Section 5.

2 RELATED WORK

Traditional Personalized Search. Personalized search has been extensively studied for selecting documents that fit user's search intent. This is usually done through analyzing the historical search data in order to infer the user's current interest, which is used in turn to influence document ranking [4]. Traditional methods rely on user's click behavior and document topic in historical search data. Among click-based approaches, Dou et al. [8] proposed a method called P-click, to evaluate relevance according to the number of clicks on the documents. A similar approach was used in [30]. Other approaches relied on topical user profiles defined by terms or topics of the clicked documents [1, 5, 11, 18, 26, 32, 33, 35]. The click-based features and topic-based features have been combined in some studies [31, 35] and learning to rank methods have bee used to combine them [2, 3, 37]. SLTB [2] is a state-of-the-art approach among these approaches and we will compare with it in Section 4.

Despite the improvements they can bring, a common limitation of the traditional approaches is that the features used to help personalized document ranking are defined manually. Not only the manual design of such features is a heavy burden, but also we may miss important features not yet discovered by experts. Deep learning provides an interesting alternative.

Application of Deep Learning in Personalized Search. Deep learning methods automatically learn representations of documents and the interaction between queries and documents, which can alleviate the problems with manually designed features. This interesting characteristic is being widely exploited in the fields of information retrieval and personalized search. A wide range of deep learning approaches have been developed for IR. One can learn document and query representations through word embedding [19, 22], document and query pair embedding [13, 25], or LSTM encoding [12, 23] and estimate the semantic similarity between the query embedding and document embedding. Severyn et al. [24] used a convolutional deep neural network to capture semantic similarity features. Song et al. [27] adapted a generic RankNet for personalized search. Li et al. [16] generated semantic features from in-session contextual information, and incorporated them into the ranking model. Dai et al. [21] used a capsule network to embed the user, query and document, and re-rank the results through the embedding. Ge et al. [9] trained a hierarchical RNN to capture both long-term and short-term interests of users, and used attention mechanism to dynamically construct user profiles.

Our model is also built within the deep learning framework. Different from the above studies that focus on the design of neural models, we address the problem of selection of training examples, which is important in general deep learning training, and especially critical for personalized search model due to the very limited amount of data available for a user. Inspired by IRGAN [34], we propose an adversarial training framework for this task.

Applications of Generative Adversarial Network (GAN). For each user, the available data is usually limited and contains inevitably some noise. This will heavily affect deep learning models to be trained. The adversarial framework provides an interesting idea of data enhancement through semi-supervised training and interaction between the generator and discriminator. The original GAN [10] aims to generate realistic simulation pictures. Since then GAN has been gradually used for different tasks in NLP. Yu et al. [38] employed GAN to generate sentences which are similar to natural language. Lin [17] employed GAN to enable the model to analyze and rank a collection of human-written and machine-written sentences in order to produce better generated sentences. GAN has also been used in information retrieval. Wang et al. proposed the IRGAN [34], which combines a generative retrieval method and a discriminative retrieval method. The generator aims to estimate the relevance distribution which is used to select good/confusing training examples, while the discriminator tries to distinguish good and bad examples. Our model is inspired by IRGAN. However, we apply the idea to personalizing search results. This paper discusses how to enhance a deep learning model by means of adversarial training to more accurately depict the user's search intent. In the next section, we will provide details of our model.

3 PSGAN - A GAN FRAMEWORK FOR PERSONALIZED SEARCH

3.1 Problem Formulation

As we have introduced in Section 1, despite the successes achieved by existing personalized search methods, several limitations remain due to the noisy and limited historical user data. Inspired by the data enhancement mechanism in GAN, in this paper, we propose the use of GAN for personalized search. Through the minimax game in adversarial training between the generator and the discriminator, we expect that the discriminator can provide reward signals to the generator, for better approaching the distribution of documents satisfying user intent, and the generator can generate high-quality negative examples and better sample weights for the discriminator. In such a way, the generator can provide additional semi-supervised information to enhance the training data and further to promote the training of the discriminator to better model search intent.

The notations used in the paper are listed in Table 1. Suppose that we are given set of queries Q and each query $q \in Q$ is issued by a user u. Let's use $\mathcal{U}$ to represent all historical search behaviours (search sessions) of u before the current query q. We split the sessions in $\mathcal{U}$ into two parts according to their timing: the past sessions $\mathcal{L}_u$ and the current session $\mathcal{S}_M$, i.e., $\mathcal{U} = \mathcal{L}_u \cup \{\mathcal{S}_M\}$. We have $\mathcal{L}_u = \{\mathcal{S}_1, \cdots, \mathcal{S}_i, \cdots, \mathcal{S}_{M-1}\}$, where M is the number of sessions associated with u. Each session $\mathcal{S}_i$ is comprised of a sequence of queries and each query includes a query string and a list of documents returned by the search engine. For example, if there are n queries in the i-th session, we have $\mathcal{S}_i = \left\{\left(\boldsymbol{q}_1^i, \mathcal{D}_1^i\right), \cdots, \left(\boldsymbol{q}_j^i, \mathcal{D}_j^i\right), \cdots, \left(\boldsymbol{q}_n^i, \mathcal{D}_n^i\right)\right\}$, where $\boldsymbol{q}_j^i$ is the j-th query in i-th session, and $\mathcal{D}_j^i$ is the search results. $\mathcal{S}_M$ includes the queries issued before q in the same session.

Without causing ambiguity, in the remaining of the paper we omit the superscripts and subscripts of the notations. We use d to denote a document in the results of query q issued by user u, whose historical search data is denoted by $\mathcal{U}$.

Let's define $p_{\text{true}}(d|q, \mathcal{U}, r)$ as the underlying true distribution of relevance r which is the personalized relevance preference of user u over document d with respect to query q and u's historical search data $\mathcal{U}$. Similar to IRGAN [34], we have two components in the adversarial framework:

discriminator: tries to learn relevance distribution $f_\phi(d, q, \mathcal{U})$ between $\mathcal{U}$, query q and d, that is, to distinguish relevant documents from irrelevant documents w.r.t. to q and $\mathcal{U}$. The data sampled from relevant documents are treated as positive examples and the data generated by the generator are treated as negative examples. f_ϕ is the discriminator function.

generator: tries to learn a distribution $p_\theta(d|q, \mathcal{U}, r)$ to approximate $p_{\text{true}}(d|q, \mathcal{U}, r)$ through a function g_θ, and generates negative examples according to the learned p_θ to confuse discriminator.

Next, we will introduce the proposed framework, namely PSGAN, to adapt the generative adversarial network to personalized search, and two different implementations of models within the framework.

3.2 PSGAN - the Framework

The minimax game in PSGAN can be described as: given a query posted by a user, the generator tries to produce a (negative) document that looks like fitting the user's intent and to fool the discriminator; while the discriminator tries to draw a clear distinction between the relevant documents and the negative document samples generated by the generator. Formally, given a set of queries Q, we have:

Table 1: Notations in our framework.

N.	Explanation	N.	Explanation
Q	a set of queries	q, q'	a query, generated query
d	a document	d'	a negative sample
u	a user	$\mathcal{U}$	u's historical data
$\mathcal{L}_u$	past sessions of u	$\mathcal{S}_M$	the current session
θ	parameters in generator	ϕ	parameters in discriminator
g_θ	function of generator	f_ϕ	function of discriminator
$\mathcal{D}$	a set of documents	p_θ	generated distribution

$$J^{G^*, D^*} = \min_\theta \max_\phi \sum_{q \in Q} \Big(\mathbb{E}_{d \sim p_{\text{true}}(d|q,\mathcal{U},r)} \log D_\phi(d|q,\mathcal{U}) + \mathbb{E}_{d \sim p_\theta(d|q,\mathcal{U},r)} \log(1 - D_\phi(d|q,\mathcal{U})) \Big), \tag{1}$$

where the generator G is written as $p_\theta(d|q, \mathcal{U}, r)$ and the discriminator D is the estimated relevance probability calculated by:

$$D_\phi(d|q,\mathcal{U}) = \sigma\left(f_\phi(d,q,\mathcal{U})\right) = \frac{\exp f_\phi(d,q,\mathcal{U})}{1 + \exp f_\phi(d,q,\mathcal{U})}.^1 \tag{2}$$

Note that different from IRGAN, PSGAN has an additional component $\mathcal{U}$ for modeling the user profiles.

3.2.1 Optimizing Discriminator. According to Eq. (1), optimizing the discriminator is to optimize ϕ to maximize the whole result given the relevant documents and the ones selected from the current optimal generator $p_\theta(d|q, \mathcal{U}, r)$, i.e.,

$$\phi^* = \arg\max_\phi \sum_{q \in Q} \Big(\mathbb{E}_{d \sim p_{\text{true}}(d|q,\mathcal{U},r)} \log D_\phi(d) + \mathbb{E}_{d \sim p_\theta(d|q,\mathcal{U},r)} \log(1 - D_\phi(d)) \Big). \tag{3}$$

We transform the above learning form into pairwise training. With a positive and a negative sample in the form of a document pair, we change Eq. (3) into:

$$\phi^* = \arg\max_\phi \sum_{q \in Q} \mathbb{E}_{d_+, d_\theta} \left[\log D_\phi(d_+) + \log\left(1 - D_\phi(d_\theta)\right) \right],$$

where d_+ is sampled by $p_{\text{true}}(d|q, \mathcal{U}, r)$ and d_θ is sampled by the generator $p_\theta(d|q, \mathcal{U}, r)$. It can be further written as Eq. (4) without changing the optimization objective:

$$\phi^* = \arg\max_\phi \sum_{q \in Q} \mathbb{E}_{d_+, d_\theta} \left(\log D_\phi(d_+) - \log D_\phi(d_\theta) \right). \tag{4}$$

To force the discriminator to pay more attention to those documents that are difficult to distinguish, we assign a weight for each document pair. A higher weight means that the negative document is more similar to a relevant document. So more attention should be paid to it. We weight (d_+, d_θ) by r_θ, according to the generation probability produced by the generative model:

$$\phi^* = \operatorname{argmax}_\phi \sum_{q \in Q} \mathbb{E}_{d_+, d_\theta} r_\theta(d_+, d_\theta) \Big(\log D_\phi(d_+) - \log D_\phi(d_\theta) \Big) \tag{5}$$

[1] In the remaining part of the paper, we may use $D_\phi(d)$, $f_\phi(d)$, $p_\theta(d|q)$ as the abbreviation of $D_\phi(d|q, \mathcal{U})$, $f_\phi(d, q, \mathcal{U})$, and $p_\theta(d|q, \mathcal{U}, r)$ to save space.

Here $r_\theta(d_+, d_\theta) = p_\theta(d_\theta|q, \mathcal{U}, r) - p_\theta(d_+|q, \mathcal{U}, r) + 1$, and $r_\theta(d_+, d_\theta)$ gets the highest score 2.0 when $p_\theta(d_\theta|q, \mathcal{U}, r)$ is 1 while that of other documents are 0, and in this case d_θ is extremely difficult to distinguish. $r_\theta(d_+, d_\theta)$ gets 1 when the two documents have equal probability and are hard to distinguish.

3.2.2 Optimizing Generator. In practice, it is difficult to generate text data due to its discrete nature. Following IRGAN [34], we generate the negative examples $\mathcal{D}'$ by selecting documents with high quality from candidate document set $\mathcal{D}$. Formally, the gradient of the generative model is:

$$\nabla_\theta J^G(q) \simeq \frac{1}{|\mathcal{D}'|} \sum_{d \in \mathcal{D}'} \nabla_\theta \log p_\theta(d|q, \mathcal{U}, r) \log\left(1 + \exp(f_\phi(d))\right).$$

Then, a document d is selected according to the probability p_θ. The weight of the document is set as $r_\theta = r_\theta(d_+, d_\theta)$. The feedback component $\log\left(1 + \exp(f_\phi(d))\right)$ given by the discriminator works as the reward to the generator.

So far we have introduced PSGAN, the general GAN framework for personalized search. In the next section, we will introduce two models based on this framework, and introduce the specific parametrisation of $p_\theta(d|q, \mathcal{U}, r)$ and $f_\phi(d, q, \mathcal{U})$.

3.3 Document Selection based Model

In this model, we adopt a **consistent structure** for both the discriminator and the generator. We start from the definition of function f_ϕ which will be used in the discriminator by Eq. (2).

3.3.1 The Discriminator. Recall that the discriminator tries to estimate personalized document relevance given the current query q and user data $\mathcal{U}$, i.e, $f_\phi(d) = \text{score}_\phi(d|q, \mathcal{S}_M, \mathcal{S}_{M-1}, ..., \mathcal{S}_1)$. In this model, we think the relevance can be further divided into three parts: the relevance of the document to the query, the relevance to the user's long-term profile and the relevance to the short-term profile. Specifically, we define $f_\phi(d)$ as:

$$f_\phi(d) = \mathcal{F}_f\left(\text{score}_\phi(d|q), \text{score}_\phi(d|\mathcal{L}_u), \text{score}_\phi(d|\mathcal{S}_M)\right), \qquad (6)$$

where $\text{score}_\phi(d|q)$ reflects the adhoc relevance between the document and the current query, $\text{score}_\phi(d|\mathcal{S}_M)$ represents the personalized relevance of a document in terms of the short-term user profile, and $\text{score}_\phi(d|\mathcal{L}_u)$ is the relevance between the document and the long-term user profile. $\mathcal{F}$ is dense layer which is used to combine the three scores and output a final personalized relevance score.

We follow the state-of-art work HRNN[2] [9] which uses a hierarchical RNN model and attention mechanism for building the long-term and short-term user profiles. However, HRNN only takes the user's historical queries and satisfied click as historical data, but ignores irrelevant documents. Note that we follow existing work [2, 9, 14, 33] and regard the click that has a dwelling time of more than 30 seconds or is the last click in a session as a satisfied click and regard those skipped documents above a satisfied click and the unclicked next document as irrelevant documents. We propose to improve the model by taking those irrelevant documents into account. We define this model as HRNN+. The structure of the model is shown in Figure 1. The details are introduced as follows.

[2]Note that here we use HRNN to represent the state-of-art work HRNN+QA in [9] for convenience

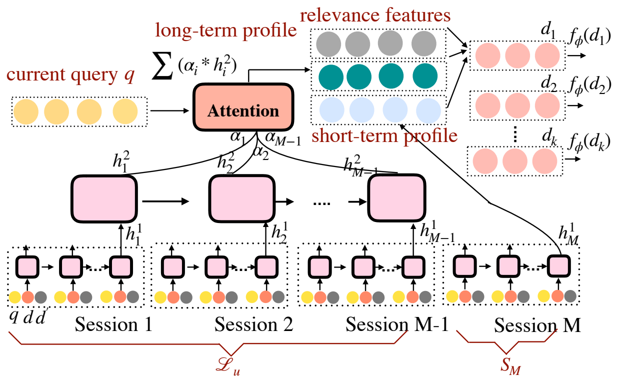

Figure 1: Structure of the HRNN+, which is used as an discriminator in the document selection model.

(1) For $\text{score}_\phi(d|q)$, we follow SLTB [2] and extract the original ranking of documents, query click entropy and other topical features as relevance features $r_{q,d}$, and we have:

$$\text{score}_\phi(d|q) = \tanh(\mathcal{F}_q(r_{q,d})), \qquad (7)$$

where $\mathcal{F}_q$ is a dense layer, and $r_{q,d}$ represents for relevance features.

(2) For $\text{score}_\phi(d|\mathcal{S}_M)$, for each query in session $\mathcal{S}_M$, we generate a vector by concatenating the average vector of relevant documents and the average vector of irrelevant documents as input to an RNN layer, i.e., we have $\boldsymbol{x}_i = [\boldsymbol{q}_i, \boldsymbol{v}_{d_i^+}, \boldsymbol{v}_{d_i^-}]$, where $\boldsymbol{x}_i$ is the input in i-th step in the session, $\boldsymbol{q}_i$ is the vector of the query string. $\boldsymbol{v}_{d_i^+}$ is the average vector of relevant documents, and $\boldsymbol{v}_{d_i^-}$ is the average vector of irrelevant documents as introduced previously. For session S_M with n queries before q, the last-step output is the encoding of the session, and is taken as the user's short-term profile, i.e., $\boldsymbol{h}_M^1 = \text{RNN}(\boldsymbol{h}_{M,n-1}^1, \boldsymbol{x}_n)$, Then the relevance score of document d in terms of user's short-term interest is:

$$\text{score}_\phi(\boldsymbol{d}|\mathcal{S}_M) = \tanh\left(\mathcal{F}_s(\boldsymbol{h}_M^1, \boldsymbol{d})\right). \qquad (8)$$

(3) For $\text{score}(d|\mathcal{L}_u)$, we use the output encodings of historical sessions in the first layer as an input to the second RNN layer, i.e., $\boldsymbol{h}_i^2 = \text{RNN}(\boldsymbol{h}_{i-1}^2, \boldsymbol{h}_i^1)$ (superscript '2' means the second layer). Similar to HRNN [9], we use an attention mechanism for weighting historical sessions on building the long-term profile, and $\alpha_i = \frac{\exp(\boldsymbol{e}_i)}{\sum_{j=1}^{M-1}\exp(\boldsymbol{e}_j)} = \text{softmax}(\boldsymbol{e}_i)$, where $\boldsymbol{e}_i = \boldsymbol{u}_i^T \boldsymbol{u}_d. \boldsymbol{u}_i = \tanh(\mathcal{F}_d(\boldsymbol{q}, \boldsymbol{h}_i^2))$ is a vector representing the matching degree between q and session S_i. Then we compute the dynamic long-term user profile by $\boldsymbol{h}_{M-1}^2 = \sum_{i=1}^{M-1} \alpha_i \boldsymbol{h}_i^2$. So,

$$\text{score}_\phi(\boldsymbol{d}|\mathcal{L}_u) = \tanh\left(\mathcal{F}_l(\boldsymbol{h}_{M-1}^2, \boldsymbol{d})\right). \qquad (9)$$

HRNN+ simply compute the relevance of a document according to $f_\phi(d)$ we just introduced, and rank the documents based on the relevance. In our document selection based model, we directly use $f_\phi(d)$ as the discriminator function. The difference between HRNN+ and the document selection based model is that HRNN+ only uses the historical click information for training, and it calculates $f_\phi(d)$ for one time and output the ranking list. Whereas in the document selection based model, $f_\phi(d)$ is trained in multiple epochs, and in each epoch, the training data will be updated by the generator.

3.3.2 *The Generator.*

The generator aims at selecting a negative document from the set of candidate documents that looks like a relevant document. As introduced, we use a similar model for the generator as the discriminator. Specifically, the generator function g_θ is defined as follows:

$$g_\theta(d, q, \mathcal{U}) = \mathcal{F}_g\left(\text{score}_\theta(d|q), \text{score}_\theta(d|\mathcal{L}_u), \text{score}_\theta(d|\mathcal{S}_M)\right)$$

The component p_θ defined in Eq. (1) is then defined by:

$$p_\theta(d|q, \mathcal{U}) = \frac{\exp\left(g_\theta\left(d, q, \mathcal{U}\right)\right)}{\sum_{d' \in \mathcal{D}} \exp\left(g_\theta\left(d', q, \mathcal{U}\right)\right)} = \text{softmax}\left(g_\theta\left(d, q, \mathcal{U}\right)\right).$$

3.4 Query Generation based Model

Since the user's intent is likely to deviate from his current query, in order to better fit the distribution of documents to the real intent of users, we further propose the second method, namely the query generation based model. **We use the same way as in the document selection based model, to train the discriminator**, to learn the relevance relationship between the query, the document and the user profiles. However, for the generator, to better estimate the user's real intent, we propose to first generate queries more likely to be in line with the user's current intent through analyzing the user's historical search log. And then we use the generator to judge the document relevance through the generated queries to better estimate the distribution of documents with respect to the user's real intent. There are some models used to generate queries in query recommendation [15, 20, 28, 36]. They take the next query in the search log as the target query and train the model by minimizing the cross entropy of the generation probability between the generated query and the target query. Different from those training methods, we train the generator to fit the distribution of related queries through the feedback given by the discriminator.

3.4.1 *The Generator.*

If we generate k queries, we can calculate the generation probability of each query using the softmax function. More specifically, for each generated query q', we have:

$$p_\theta(q'|q, \mathcal{U}) = \text{softmax}\left(g_\theta(q', q, \mathcal{U})\right) \tag{10}$$

Where g_θ is the function of generator, and p_θ is the probability distribution calculated by the generator.

At the same time, we can calculate the probability distribution of the documents under each generated query through the function given by discriminator. We have:

$$p_\theta(d|q') = \text{softmax}\left(f_\phi(d, q', \mathcal{U})\right) \equiv p_\phi(d|q')$$

So according to the conditional probability formula, we can define the probability distribution p_θ of the document as:

$$p_\theta(d|q) = \sum_{q'} p_\theta(q'|q, \mathcal{U}) p_\theta(d|q') = \sum_{q'} p_\theta(q'|q, \mathcal{U}) p_\phi(d|q') \tag{11}$$

In this case the gradient of the generative model changes to:

$$\nabla_\theta J^G(q) = \frac{1}{|\mathcal{D}'|} \sum_{d \in \mathcal{D}'} \nabla_\theta \log\left(\sum_{q'} p_\theta\left(q'|q, \mathcal{U}\right) p_\phi\left(d|q'\right)\right) * \log\left(1 + \exp(f_\phi(d))\right)$$

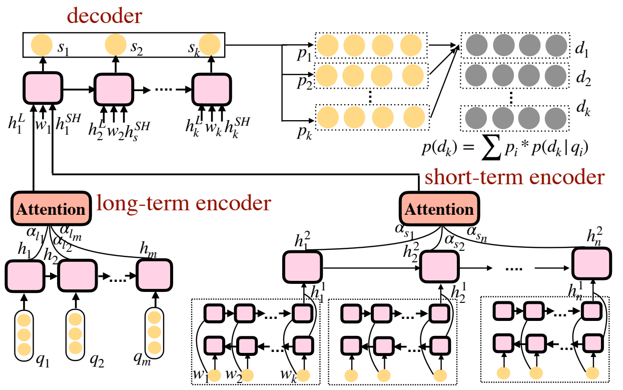

Figure 2: Structure of generator in the query generation based model. The long-term encoder uses a RNN to encode query sequences in the past sessions. The short-term encoder uses a hierarchical RNN to model the fine-grained information in the current session. The decoder uses two attention mechanisms over a RNN to generate queries which help estimate the probability of selecting a document.

As $\log(1+\exp(f_\phi(d)))$ and $p_\phi(d|q', \mathcal{U}, r)$ work as the feedback given by the discriminator, we can see that optimizing the parameters of generator θ is equivalent to making the generated queries more in line with the user's real intent.

For discriminator, we use the same training goal and function f_ϕ shown in Eq. (6) with the document selection based model. Next, let's focus on the generator's function g_θ.

For generator, we employ a sequence-to-sequence model, which will be introduced in the next section, to generate queries, then apply Eq. (11) to select documents.

3.4.2 *Query Generation.*

We employ two encoders to encode historical information, and a decoder to generate queries. Since the user's query intention can be inferred from the relevant queries in the user's historical queries in some cases, we first use a long-term encoder to process the historical information in the past sessions. As the user's query intention is often consistent in a session, in order to accurately depict the user's session intention, we employed a short-term encoder to process the historical queries in the current session. The model is shown in Figure 2.

(1) For the long-term encoder, we use an RNN to take the historical queries sequence in the past sessions $\{\boldsymbol{q}_1, ..., \boldsymbol{q}_i, ...\boldsymbol{q}_m\}$ as input (we assume there are m queries in $\mathcal{L}_u$ in total). For each step when a new query q_i is fed in, we have $\boldsymbol{h}_i = \text{RNN}(\boldsymbol{h}_{i-1}, \boldsymbol{q}_i)$.

(2) Since there are several queries in a session, in order to describe more fine-grained, we use a hierarchical structure to encode the current search intent. Both the relationship between the internal terms of a query and the relationship between queries are considered. For the short-term encoder, again we assume there are n queries before q in the current session. We first use a bi-directional RNN to encode each query over the query terms as the first layer, and then use another RNN to encode the outputs of the first layer as the second layer. Specifically, for the first layer, we have $\overrightarrow{\boldsymbol{h}_t^w} = \text{RNN}(\overrightarrow{\boldsymbol{h}_{t-1}^w}, \boldsymbol{w}_t)$ and $\overleftarrow{\boldsymbol{h}_t^w} = \text{RNN}(\overleftarrow{\boldsymbol{h}_{t+1}^w}, \boldsymbol{w}_t)$. And for the outputs of the first layer for q, suppose L_i is the number of terms contained in query string

q_i, we have $\boldsymbol{h}_i^1 = [\overrightarrow{\boldsymbol{h}_{L_i}^w}, \overleftarrow{\boldsymbol{h}_{L_i}^w}]$. Similarly, for the second layer, the representation of the i-th query is encoded by $\boldsymbol{h}_i^2 = \mathrm{RNN}(\boldsymbol{h}_{i-1}^2, \boldsymbol{h}_i^1)$.

(3) For the decoder, we use the last output of the short-term encoder to initialize the state by $s_0 = \tanh(\mathcal{F}_h(\boldsymbol{h}_n^2))$. We further add an attention mechanism for the decoder over both encoders. The generation of t-th word of the query q' is supervised by comparing the similarity between this word and hidden states learned by the long-term and short-term encodes. Specifically, for the i-th query in the long-term encoder, the associated attention is $\alpha_{t,i}^{\mathcal{L}} = \mathrm{softmax}\left(\boldsymbol{u}_i^T \boldsymbol{u}_m\right)$ where $\boldsymbol{u}_i$ is a shorten form for $\boldsymbol{u}_{t,i}^{\mathcal{L}} = \tanh(\mathcal{F}_m(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i))$ and $(\boldsymbol{u}_i^T \boldsymbol{u}_m)$ is used to measure the correlation between the previous state $\boldsymbol{s}_{t-1}$ of the decoder and the i-th query's encoding $\boldsymbol{h}_i$. Finally, the hidden state of the next term is $\boldsymbol{h}_t^{\mathcal{L}} = \sum_i \alpha_{t,i}^{\mathcal{L}} \boldsymbol{h}_i$. Similarly, the hidden state of the next term weighted by the short term encoder is: $\boldsymbol{h}_t^{\mathcal{SH}} = \sum_i \alpha_{t,i}^{\mathcal{SH}} \boldsymbol{h}_i^2$, and $\alpha_{t,i}^{\mathcal{SH}} = \mathrm{softmax}((\boldsymbol{u}_{t,i}^{\mathcal{SH}})^T \boldsymbol{u}_n)$ where $\boldsymbol{u}_{t,i}^{\mathcal{SH}} = \tanh(\mathcal{F}_n(\boldsymbol{s}_{t-1}, \boldsymbol{h}_i^2))$. We concatenate $\boldsymbol{h}_t^{\mathcal{L}}$ and $\boldsymbol{h}_t^{\mathcal{SH}}$ as the final context vector for the decoder to select next term: $\boldsymbol{c}_t = \left[\boldsymbol{h}_t^{\mathcal{L}}, \boldsymbol{h}_t^{\mathcal{SH}}\right]$. And for each step of the decoder, the generation probability is calculated as:

$$P(\boldsymbol{w}_t | \boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_{t-1}, \boldsymbol{c}_t) = \mathrm{softmax}\left(\mathcal{F}_o\left(\mathrm{RNN}\left(\boldsymbol{s}_{t-1}, [\boldsymbol{c}_t, \boldsymbol{w}_t]\right)\right)\right).$$

Finally, for a generated query $q' = [\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_k]$, which is composed of k words, its generation probability is estimated by:

$$g_\theta(q', q, \mathcal{U}) = \prod_{i=1}^{n} P(\boldsymbol{w}_i | \boldsymbol{w}_{1:i-1}). \qquad (12)$$

The implementation of $g_\theta(q', q, \mathcal{U})$ can then be applied to Eq. (10) to calculate $p_\theta(q'|q, \mathcal{U})$, which is used in Eq. (11) for estimating p_θ and train the generator.

3.4.3 Query Candidate Selection. The above model assumes that we generate queries based on the user's previous search histories. However, we find some difficulties if we directly generate queries. Firstly,the impact of the generated queries to the final ranking is difficult to evaluate, because some relevance features between the query and documents such as click information are unavailable if the query never appeared in the search log before. Second, because the word dictionary is too large, the quality of generated query is not stable. Therefore, we require that the generated query must have appeared in our search log, i.e, we train the query generation model but only apply the model to a limited set of candidate queries which are in the query log. This can control the risk to a certain extent and make the model more stable.

To decide the list of candidate queries we can generate, we use the following ranking function to rank the historical queries in the log and select the top 10. For a candidate query q_j, we simply use the following equation to calculate its importance to q:

$$R(q_j|q) = e(q, q_j) + s(q, q_j) + f(q, q_j) + r(q, q_j)$$

where: (1) $e(q, q_j) = \frac{\mathrm{len}(q_j) - \mathrm{len}(q)}{\mathrm{len}(q)}$ when q_j is the expansion of q, otherwise, we let $e(q, q_j) = 0$. This component simply evaluates the specificity between q_j and q. The more words the expansion query q_j contains, the more spefic q_j is to q; (2) $s(q, q_j) = sim(\boldsymbol{q}, \boldsymbol{q}_j)$ is the similarity between two vectors of q and q_j. This evaluates the semantic relationship between the two queries; (3) $f(q, q_j) = \frac{n_{q,q_j}}{n_q}$, n_{q,q_j} represents the number of times the two queries appear in the same session at the same time and n_q represents the number of times q appears in the whole search log; (4) $r(q, q_j) = \frac{c_{q,q_j}}{c_q}$ indicates the click relevance between the two queries, where c_{q,q_j} is the number of URLs both clicked under the two queries q and q_j, and c_q is the number of URLs clicked under q.

Once the candidate queries are selected, we use Eq. (12) to evaluate the importance of each candidate, and select documents based on Eq. (11). In this way, we can provide a better choice when the user's real intent is inconsistent with the current query. In order to ensure that the model can degenerate to the current query in the case that the current query is consistent with the user's intent or there is no better choice than the current query in the candidate set, we also add the current query into the candidate set when training and testing the model.

3.5 Review of Our Models

Our model attempts to solve the problem of limited and noisy data in personalized search by introducing generative adversarial network. By means of adversarial training, the generator and discriminator promote each other and finally yields better ranking models. For the document selection based model, we follow the idea of IRGAN[34] to convert the document generation into document selection. However, IRGAN dose not deal with search result personalization. Our first method is a natural extension of IRGAN to personalized search. For the query generation based model, we focus on how to better personalize the search results by first predicting real user intent. Because the user does not click on the document based on the current query when searching, but judges whether the document is relevant through the search intent in their mind, we assume that the relevance of documents can be better estimated if the search intent can be detected. In the generator of this model, we first considers generating queries to fit the distribution of user intent, and then fits the distribution of documents. With this kind of mechanism, the relevance probability of the document under the user's intent can be better judged, and this can further improves the effectiveness of search result personalization.

4 EXPERIMENTS

4.1 Dataset

We experiment with the proposed models and the baselines based on a search log collected from a commercial search engine during January to February in 2013. Each data record in the log contains user id, query string, time the query was issued, URLs retrieved by the search engine, a tag identifying whether the user clicked a document, and dwelling time for the click document. The log is extracted from the search engine based on a list of sampled user id, to make sure all search histories during the time period for these users are kept in our dataset. The dataset contains 33,204 users and 2,665,625 queries. Following the existing work [2, 8, 9], we segment the logs into sessions according to the query issuing inactivity of longer than 30 minutes without overlap. Finally we obtain 654,776 sessions. Because in the second method proposed in this paper we needs to generate queries based on the current queries, some meaningless queries are filtered, such as queries that contain only a single meaningless word like 'a'.

We divide the user search log into historical data and experimental data in order to ensure that each user has historical data. For historical data we used the first six weeks in user's search history of each user, and this part of the data only provides the user's historical information during training and testing. To ensure the this, we filtered the users who have no historical data and experimental data. Then we use the last four weeks data in each user's search log as experimental data. We further divide the experimental sessions into training set, validation set and test set according to the ratio of 4:1:1.

We pre-train a 300-dimension word vector model based on the queries and document content in the experimental data with word2vec. In the experiments, we simply average the word vectors of a query as the representation of the query, and use the tf-idf weighted average word vectors as the document vector.

We follow existing work [2, 9, 33] and regard the click that has a dwelling time of more than 30 seconds or is the last click in a session as a satisfied click (*sat-click*) and call other documents *non-sat-clicked.* The original training data are document pairs formed by sat-clicked documents and some non-sat-clicked documents. We use the *sat-clicked* documents as relevant documents, and the selected *non-sat-clicked* documents as irrelevant. Our models enhance the training data by sampling non-sat-clicked documents which are more difficult to distinguish as irrelevant documents and weighting the data pairs.

4.2 Baselines and Our Models

Although IRGAN [34] applied generative adversarial network to web search, its main goal of optimization is not personalizing the search results, and hence it cannot be directly applied to personalized search. We cannot directly compare with it in the experiments. However, our document selection based model can be seen as a natural extension of IRGAN to personalized search. Our query generation based model goes a step further to utilize the characteristic of personalized search and is expected to be more effective on personalization.

In order to verify the effectiveness of our models, we compare our models with the state-of-the-art personalized models based on traditional features and the deep learning based methods. These models are listed as follows.

P-Click: This method was proposed by Dou et al. in [8]. In the model, documents are reranked by the number of clicks the same user has made on the same query with the Borda Count ranking fusion method. This model can stably reflect the effect of click features in personalization.

SLTB: SLTB [2] integrates clicks features and topical features and uses LambdaMART to train the personalized ranker. Previous works [2, 9] show this is the state-of-the-art approach before deep learning is applied.

HRNN: This model [9] builds user profiles using a hierarchical RNN and uses attention to dynamically highlight different queries when personalizing the next query. It is considered as a state-of-the-art personalized search model based on deep learning.

Table 2: Overall performances. Best results are in bold. $*$ indicates the model significantly outperforms Ori.Ranking, P-Click, and SLTB, and $\star$ indicates the model significantly outperform all baselines ($p < 0.05$ in two-tailed paired t-test).

Model	MAP	MRR	A.Clk	#Better	#Worse	P-Imp.
Ori. R	.7321	.7419	2.211	-	-	-
P-Click	.7426	.7541	2.073	3,662	33	.0687
SLTB	.7876	.7982	1.935	13,181	2,016	.2140
HRNN	.8021	.8139	1.887	15,113	1,958	.2490
HRNN+	$.8029^{*}$	$.8142^{*}$	1.860^{*}	$15,227^{*}$	$1,989^{*}$	$.2506^{*}$
PSGAN-D	$.8082^{\star}$	$.8191^{\star}$	$1.838^{\star}$	$14,952^{\star}$	$\mathbf{1,609}^{\star}$	$.2527^{\star}$
PSGAN-G	$.7923^{*}$	$.8054^{*}$	1.967^{*}	$14,755^{*}$	$2,749^{*}$	$.2273^{*}$
QG-D	$\mathbf{.8104}^{\star}$	$\mathbf{.8214}^{\star}$	$\mathbf{1.834}^{\star}$	$\mathbf{15,458}^{\star}$	$1,716^{\star}$	$\mathbf{.2601}^{\star}$
QG-G	$.8018^{*}$	$.8131^{*}$	1.866^{*}	$15,044^{*}$	$2,654^{*}$	$.2374^{*}$

We experiment with the following ranking models corresponding to the two models we proposed in the paper:

HRNN+: An improved version of HRNN introduced in Section 3.3. It is used to initialize the parameters of the discriminators.

PSGAN-D and **PSGAN-G**: They rank the documents using the document relevance score calculated by the discriminator (**D**) and generator (**G**), using the document selection based model.

PSGAN-QG-D and **PSGAN-QG-G**: These are two ranking models based on the query generation model. We use the shorten form **QG-D** and **QG-G** in the table.

For all deep learning based models, we use GRU cells to construct the recurrent neural network. The size of the word vector and other representations is 300. The size of hidden state of the GRU layer and the size of the decoder output is 512.

4.3 Evaluation Metrics

We evaluate the models using the widely used IR metrics MAP and MRR in this paper, with the assumption that the sat-clicked documents are relevant and others are irrelevant. We further use the average rank position of the sat-clicked documents (Avg. Click) to intuitively indicate where the relevant documents are ranked. A lower Avg. Click value indicates a better personalized ranking because relevant documents are ranked higher. As stated in [7, 14], due to the influence of original ranking of documents in search engine, taking the documents skipped above the click or the non-clicked next document as irrelevant is more creditable. Therefore, we count the number of inverse document pairs on documents skipped above and documents non-clicked next to reflect the personalized effect of the results more credibly. For document pairs constructed by documents skipped above and the clicked documents, reranking the document list can only produce better effect. So we take the metric #Better as the number of inverse document pairs on which the model ranks the clicked document higher than the skipped document. For document pairs constructed by the click and non-clicked next, reranking the documents can only produce worse results. So we take the metric #Worse as the number of inverse document pairs on which the non-clicked next document is ranked higher. Taking the number of all pairs constructed by documents skipped above and clicked documents as S-pair, and the number of all pairs constructed by clicked documents and non-clicked next documents as N-pair, we define a new metric P-Improve as P-Improve = $\frac{\#\text{Better}-\#\text{Worse}}{\text{S-pair+N-pair}}$. P-Improve is an intuitive metric to evaluate the ranking improvement over a baseline ranking with reliable relevance preferences other than absolute relevance ratings.

4.4 Overall Performance

We first give the overall results and compare our models with the baselines to explore whether the PSGAN framework can effectively improve personalization. The overall results are shown in Table 2.

(1) All personalization models outperform the original ranking - the ranking returned by the search engine. The improvement of P-Click model confirms the effectiveness of refinding behavior in search engines. Using learning to rank, SLTB combining simple click features and topical features outperforms the simple click-based model. HRNN, which uses deep neural networks and attentions to build the dynamic user profiles through user's sequential search data, outperforms SLTB. This confirms the effectiveness of deep learning on personalized search. Our proposed model HRNN+ is better than HRNN a little bit, which shows that our method considering the irrelevant documents in historical search data is feasible.

(2) Compared with the baseline models, we find that our discriminator based ranking models (PSGAN-D and PS-QG-D) significantly improve the quality of personalized search in terms of all evaluation metrics. Especially compared with HRNN, our adversarial personalized models also have significant improvements. The MAP has been improved 6.1‰by PSGAN-D and 8.3‰by PSGAN-QG-D (shown as QG-D in the table) compared with HRNN. In terms of P-Improve, which shows the personalization quality from a more credible angle, the improvement made by PSGAN-D is 3.7‰and by PSGAN-QG-D is 11.1‰. These results show adversarial training can promote the effectiveness of deep learning models for personalized search. Note that the improvement of the adversarial model over the existing deep learning model HRNN is not as large as that of the deep learning model over traditional personalization models. We think this is because in the search data we use, the original ranking is already quite good. For example, the value of MAP has reached 0.73. Therefore, it leaves little room for the deep learning model to improve. While based on the deep learning model, the space for the adversarial model is even smaller. However, we can still see that the application of adversarial model is effective in improving the quality of deep learning model by enhancing the data.

(3) The query generation based model outperforms the document selection based model. In terms of all metrics but #Worse, the discriminator of the query generation based model (PSGAN-QG-D) outperforms the document selection based model (PSGAN-D). For generators, we find PSGAN-QG-G outperforms PSGAN-G. This shows that the relevance between documents and real user intent can be better predicted by first fitting the user intents through generating better queries and then fitting the distribution of relevant documents under the generated queries. And the generator with the support of generated queries can also promote the quality of discriminator correspondingly.

(4) Comparing the discriminators (PSGAN-D, PSGAN-QG-D) with the generators(PSGAN-G, PSGAN-QG-G), we can see in both adversarial models, the discriminator performs better than generators. One possible reason is that the generator does not really generate new data and cannot obtain new information by exploring more other data space. The data space is limited in the original space observed by discriminator, and all feedback depends on discriminator, so the generator's effect is difficult to exceed discriminator.

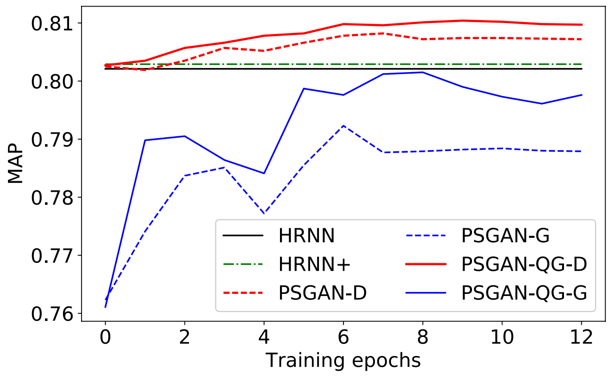

Figure 3: Learning curves of discriminators and generators.

To summarize, experimental results show that **our personalized adversarial framework PSGAN can effectively improve the data quality and promote the training of the deep learning model**. In addition, it is also effective to predict the users' current real query intent and use the generated queries to rerank the documents.

4.5 Learning Curves

In order to analyze the effectiveness of the adversarial training in our models, we show the learning curves during the adversarial training in Figure 3. In this figure, we plot the change of MAP in each epoch of our discriminators and generators in the models. It can be seen that with the increase of training epochs, the quality of the generators and discriminators is continuously improved and finally tends to be stable. And in the process of training, the discriminator and the generator interact and promote each other. Compared with HRNN, the performance of our discriminators is steadily above HRNN. And comparing the two discriminators, the query generation based model (PSGAN-QG-D) can better approach the personalized relevance classification boundary through adversarial training than the document selection based model (PSGAN-D).

In the following analysis, we will mainly compare the effects of our discriminators (PSGAN-D and PSGAN-QG-D) with those of other comparative models.

4.6 Experiments with Click Entropy

As reported by previous study [8, 9, 29], personalized search is less effective on queries with a smaller click entropy than queries with a larger entropy. Here we divide queries into two categories by click entropy [8]. We split the queries into two groups: the queries with click entropy less than 1.0 (usually they are navigational and clear queries) and those with click entropy no less than 1.0 (tending to be ambiguous, broad, or informational queries). We calculate the average MAP improvement of the models on the two categories, and show the results in Figure 4. Here we compare our models PSGAN-D and PSGAN-QG-D with the best deep learning personalization baseline model HRNN and the traditional model SLTB.

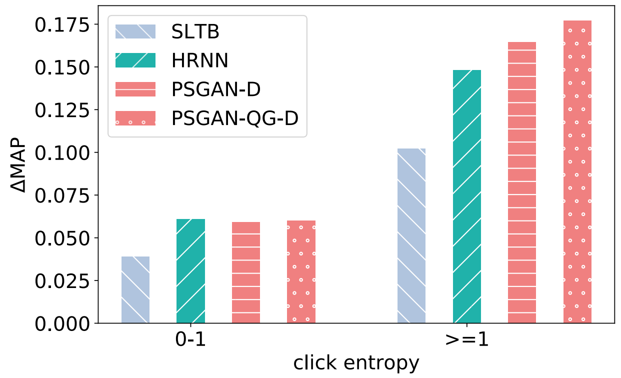

Figure 4: Results on queries with different entropies.

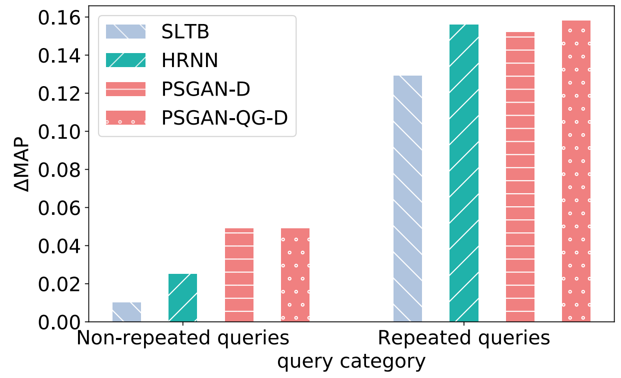

Figure 5: Results on repeated/non-repeated queries.

Figure 4 shows that our models consistently outperform SLTB, no matter whether the query is clear (lower entropy) or ambiguous (larger entropy). The overall improvement of the personalization models over the original ranking is much more significant on queries with a larger click entropy.The improvement of our models over SLTB is also larger on the queries with a larger entropy than the queries with a lower entropy.

Comparing PSGAN with HRNN, we find that our models PSGAN-D and PSGAN-QG-D have little difference from HRNN on the queries with entropy lower than 1.0, but they perform much better on the other part. This shows that our adversarial models have better effect on queries with more personalized features, which meets the experimental expectations. Compare the document selection based model PSGAN-D and the query generation based model PSGAN-GQ-D, we can also see the latter has more improvement on the second group of queries.

4.7 Results on Repeated/Non-repeated Queries

As personalized search heavily relies on the historical user behaviour and previous study [6, 8, 29] has shown that some click based personalization features are very effective on refinding information. To investigate the effectiveness of our models on this dimension, we experiment the models with the queries that the user has issued before (repeated queries) or not (non-repeat queries). The results are shown in Figure 5. Consistently, we find that all personalization models have larger MAP improvement over the original ranking, and the deep learning based models outperform SLTB more significantly on repeated queries than on non-repeated queries. However, we find that our model PSGAN-D slightly underperforms HRNN on the repeated queries but it is much better than HRNN on the non-repeated queries. A possible reason is that our model can better learn the relevance model by means of adversarial training, but not heavily depend on the historical click and refinding behaviour. Even when the query is issued by the user at the first time, the model can successfully predict the personalized relevance of the documents. This further confirms the effectiveness of our model.

Table 3: Overall performances of models trained on the preference data. All proposed PSGAN models (PSGAN-D, PSGAN-G, PSGAN-QG-D, and PSGAN-QG-G) significantly outperform the baseline methods (SLTB and HRNN).

Model	MAP	MRR	A.Clk	#Better	#Worse	P-Imp.
SLTB	.4639	.4732	3.6170	16,792	9,436	.1401
HRNN	.5045	.5146	3.2922	17,475	8,350	.1727
HRNN+	.5065	.5169	3.2819	17, 553	8, 328	.1746
PSGAN-D	.5270	.5376	3.1654	17, 658	8, 072	.1815
PSGAN-G	.6391	.6478	2.6777	**19, 949**	6, 540	.2539
QG-D	.5213	.5316	3.190	17, 586	8, 073	.1801
QG-G	**.6926**	**.7031**	**2.4173**	19, 265	**5, 541**	**.2599**

4.8 Handling the Position Bias

As mentioned in [7, 14], clicks are influenced not only by the relevance of documents but also by the positions. The low-ranked documents have lower probabilities to be viewed by user, so documents that have not been clicked are not necessarily irrelevant. So next we will experiment with a more reasonable way to make use of the search data and eliminate the influence of position bias. We follow [14] to generate more reliable document preference pairs as training data, but not directly use clicks as absolute document relevance ratings. We select document pairs comprised of a sat-clicked document (+) and a skipped document above the sat-click (-), and the pairs comprised of a sat-click (+) and its unclicked next document (-). We treat other pairs as unlabeled. We train SLTB, HRNN, and HRNN+ using these preference like training data. When training the adversarial model, we ask the generator to generate more training data from the unlabeled pairs comprising of a sat-click and other unclicked documents. In this way, the model can solve the problem of limited data for training personalized models.

Results are shown in Table 3. We find that:(1) all our PSGAN models significantly outperform the baseline methods (SLTB and HRNN) in terms of all metrics. Especially, PSGAN-G and PSGAN-QG-G yield over 0.08 improvements in terms of P-Improve. Although the effect of all the baseline models have declined compared with that in Table 2 in P-Improve, PSGAN-QG-G still approaches the best situation in Table 2. This shows the stability of our model and it works well on handling the influence of position bias. (2) Consistent with the previous results, the query generation based models (QG) outperforms the document selection based models, and the query generation based model PSGAN-QG-G performs the best on most metrics. In this setting of training data, in addition to providing weights of data pairs and selecting high-quality negative examples for data enhancement, our adversarial model can also expand data space through the adversarial framework by sampling negative examples from irrelevant and unlabeled document spaces. So this result not only shows the effectiveness of the adversarial model in personalized search, but also shows that using the generative adversarial model can provide a reasonable way to use the unlabeled data in personalized search and thus promoting the training of the deep learning model. (3) Comparing PSGAN-D with PSGAN-G, and PSGAN-QG-D with PSGAN-QG-G, we can see in this case the generator performs better than the discriminator in terms of all metrics. We think this is because the generator can get more information from the expanded unlabeled data space. (4) The MAP, MRR and Avg.Click have obviously declined compared with the result in Table 2. After observing the data, we find that this is because there are some documents in lower rank but are also related to the user's intent, thus the ranking is reranked higher by those models, resulting in the decline of MAP, MRR and Avg.Click.

5 CONCLUSION

In this paper, we proposed PSGAN - a framework for personalized adversarial training, which alleviates the problems of small amount of user data and noisy data in personalized search. We proposed two models within this framework, namely the document selection based model and the query generation based model, to effectively improve the quality of personalization. In the second model, we work out an idea on first generating queries that predict real search intent, then estimating document relevance through the generated queries. Experimental results confirmed the effectiveness of the proposed models. In future work, we plan to consider generating queries in the entire vocabulary space instead of just selecting queries that already exist in the log.

ACKNOWLEDGMENTS

Zhicheng Dou is the corresponding author. This work was supported by National Key R&D Program of China No. 2018YFC0830703, the National Natural Science Foundation of China No. 61872370 and 61872338, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China No. 2112018391 and 2018030246.

REFERENCES

[1] Paul N. Bennett, Krysta Marie Svore, and Susan T. Dumais. 2010. Classification-enhanced ranking. In *WWW*. ACM, 111–120.

[2] 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 *SIGIR*. ACM, 185–194.

[3] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of ICML'2005*. ACM, 89–96.

[4] Fei Cai, Shangsong Liang, and Maarten de Rijke. 2014. Personalized document re-ranking based on Bayesian probabilistic matrix factorization. In *SIGIR*. 835–838.

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

[6] Kevyn Collins-Thompson, Paul N Bennett, Ryen W White, Sebastian De La Chica, and David Sontag. 2011. Personalizing web search results by reading level. In *Proceedings of the CIKM'2011*. ACM, 403–412.

[7] Nick Craswell, Onno Zoeter, Michael J. Taylor, and Bill Ramsey. 2008. An experimental comparison of click position-bias models. In *WSDM'2008*.

[8] Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. A large-scale evaluation and analysis of personalized search strategies. In *WWW*. ACM, 581–590.

[9] Songwei Ge, Zhicheng Dou, Zhengbao Jiang, Jian-Yun Nie, and Ji-Rong Wen. 2018. Personalizing Search Results Using Hierarchical RNN with Query-aware Attention. (2018), 347–356.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. (2014), 2672–2680.

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

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *CIKM'2013*. ACM, 2333–2338.

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

[15] Wei Wang Jyun-Yu Jiang. 2018. RIN: Reformulation Inference Network for Context-Aware Query Suggestion. (2018), 197–206.

[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] Kevin Lin, Dianqi Li, Xiaodong He, Ming-Ting Sun, and Zhengyou Zhang. 2017. Adversarial Ranking for Language Generation. (2017), 3158–3168.

[18] Nicolaas Matthijs and Filip Radlinski. 2011. Personalizing web search using long term browsing history. In *WSDM*. ACM, 25–34.

[19] Bhaskar Mitra and Nick Craswell. 2017. Neural Models for Information Retrieval. *arXiv preprint arXiv:1705.01509* (2017).

[20] Enrique Alfonseca Pascal Fleury Mostafa Dehghani, Sascha Rothe. 2017. Learning to Attend, Copy, and Generate for Session-Based Query Suggestion. (2017), 1747–1756.

[21] Dai Quoc Nguyen, Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Q. Phung. 2018. A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. *CoRR* abs/1808.04122 (2018).

[22] Kezban Dilek Onal, Ye Zhang, Ismail Sengor Altingovde, Md Mustafizur Rahman, Pinar Karagoz, Alex Braylan, Brandon Dang, Heng-Lu Chang, Henna Kim, Quinten McNamara, et al. 2018. Neural information retrieval: At the end of the early years. *Information Retrieval Journal* 21, 2-3 (2018), 111–182.

[23] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *TASLP* 24, 4 (2016), 694–707.

[24] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR'2015*. ACM, 373–382.

[25] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *CIKM'2014*. ACM, 101–110.

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

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

[28] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *CIKM'2015*. ACM, 553–562.

[29] 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*. ACM, 163–170.

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

[31] Maksims Volkovs. 2015. Context Models For Web Search Personalization. *CoRR* abs/1502.00527 (2015).

[32] Thanh Vu, Dat Quoc Nguyen, Mark Johnson, Dawei Song, and Alistair Willis. 2017. Search Personalization with Embeddings. In *ECIR (Lecture Notes in Computer Science)*, Vol. 10193. 598–604.

[33] Thanh Tien Vu, Alistair Willis, Son Ngoc Tran, and Dawei Song. 2015. Temporal Latent Topic User Profiles for Search Personalisation. In *ECIR (Lecture Notes in Computer Science)*, Vol. 9022. 605–616.

[34] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. (2017), 515–524.

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

[36] Bin Wu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Query Suggestion with Feedback Memory Network. (2018), 1563–1571.

[37] Qiang Wu, Chris JC Burges, Krysta M Svore, and Jianfeng Gao. 2008. *Ranking, boosting, and model adaptation.* Technical Report. Technical report, Microsoft Research.

[38] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. (2017), 2852–2858.