Personalized Query Suggestion with Searching Dynamic Flow for Online Recruitment

Zile Zhou[*][†]
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
ziyao9902@ruc.edu.cn

Xiao Zhou[*]
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
xiaozhou@ruc.edu.cn

Mingzhe Li
Wangxuan Institute of Computer Technology, Peking University
Beijing, China
li_mingzhe@pku.edu.cn

Yang Song
BOSS Zhipin NLP Center
Beijing, China
songyang@kanzhun.com

Tao Zhang
BOSS Zhipin
Beijing, China
kylen.zhang@kanzhun.com

Rui Yan[‡]
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
rui.yan.pku@gmail.com

ABSTRACT

Employing query suggestion techniques to assist users in articulating their needs during online search has become increasingly vital for search engines in an age of exponential information growth. The success of a query suggestion system lies in understanding and modeling user search intent behind each query accurately, which can hardly be achieved without personalization efforts on taking advantage of dynamic user feedback behaviors and rich contextual information. This valuable area, however, has been still largely untapped by current query suggestion systems. In this work, we propose Dynamic Searching Flow Model (DSFM), a query suggestion framework that is capable of modeling and refining user search intent progressively in recruitment scenarios by leveraging a dynamic flow mechanism. Here the concepts of *local flow* and *global flow* are introduced to capture the real-time intention of users and the overall influence of a session, respectively. By utilizing rich semantic information contained in resumes and job requirements, DSFM enables the personalization of query suggestions. In addition, weighted contrast learning is introduced into the training process to produce more extensive targeted query samples and partially alleviate the exposure bias. The adoption of attention mechanism allows the selection of the most relevant information to compose the final intention representation. Extensive experimental results on different categories of real-world datasets demonstrate the effectiveness of our proposed approach on the task of query suggestion for online recruitment platforms.

[*]Both authors contributed equally to this research.
[†]Work done during an internship at BOSS Zhipin.
[‡]Corresponding author: Rui Yan (rui.yan.pku@gmail.com).

CCS CONCEPTS

• **Information systems → Query suggestion**; **Personalization**; **Learning to rank**.

KEYWORDS

Query Suggestion, Personalized Recommendation, Intelligent Recruitment

ACM Reference Format:
Zile Zhou, Xiao Zhou, Mingzhe Li, Yang Song, Tao Zhang, and Rui Yan. 2022. Personalized Query Suggestion with Searching Dynamic Flow for Online Recruitment. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3511808.3557416

1 INTRODUCTION

Given the enormous volume of data generated continuously on the world wide web every day and the complexity of information exploration, users often struggle with finding appropriate queries to ask when they interact with search engines [17, 59]. Query suggestion systems come to the rescue [36] and improve the usability of modern search engines [6] by offering a set of alternative queries that are generally more expressive than the original user input based on her search behaviors [16, 28]. To do this, query suggestion systems are expected to understand, specify, and deduce the hidden user search intent during a search session accurately and comprehensively [28, 37]. However, to assist users in refining their original queries so as to guide them to the desired information is rarely an easy or straightforward task [5], which is especially true when facing complex search needs, e.g., job hunting [37].

The challenges query suggestion systems confront are multifold. Among them, one of the factors contributing to the difficulty most is the inherent ambiguity of search queries issued by users [59]. In

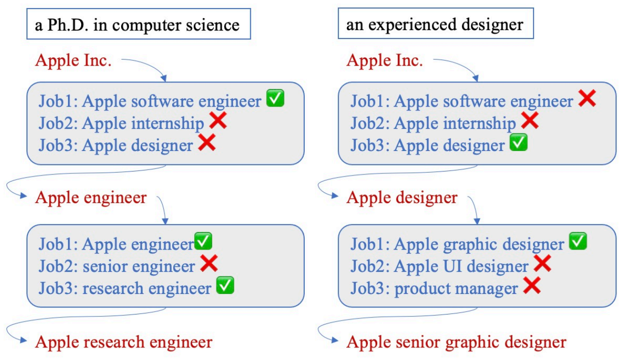


Figure 1: An example to illustrate the sign Dynamic Searching Flow in the semantic space. When a user is searching a series of queries, the response to the work list (click or no click) could explain to some extent the direction of the offset in next search.

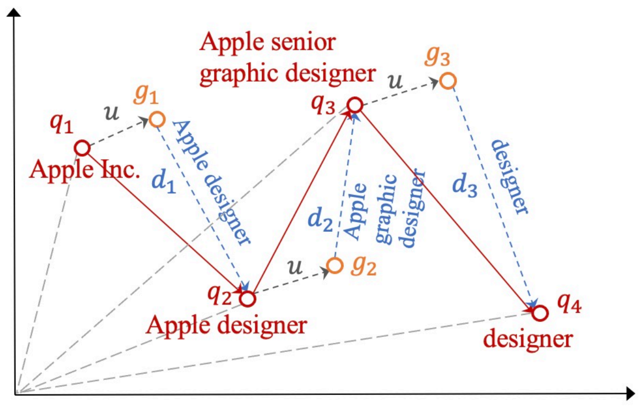


Figure 2: Illustration of Dynamic Searching Flow in the semantic space. q_i denotes historical queries, u denotes the user's information, g_i denotes search intent extracted by current query and user's profile and d_i denotes the user's intention extracted by job descriptions after each search. The difference between q_{i-1} and q_i is derived from g_{i-1} and d_{i-1}.

practice, casual users with little prior knowledge regarding the information they are searching for would usually find it difficult to formulate expressive search queries initially that query suggestion systems usually may have to refine the queries multiple times to project their specific needs [28, 37]. Moreover, the search queries entered into the web search engines by users are typically very short (less than five terms according to a survey [3]) [24, 50]. To tackle this, the exploitation of additional information sources, such as user feedback like clicks from search sessions, demographic data about users, and relevant textual information available on the platform is of necessity for query suggestion systems to potentially reduce the ambiguity and better understand customer search intent, which, unfortunately, has been seldom considered in most existing studies [28, 37].

Another limitation existing commonly in current query suggestion systems is the lack of personalization, which holds for both traditional term-frequency-based approaches and the more recent word-embedding-based techniques [56, 59]. For query suggestion systems using word embeddings, personalization can be hard to achieve without learning on user profile and custom contextual information. However, to meet the growing need for personalized customer experiences, search engines are increasingly expected to deliver qualified content recommendations that are individually tailored to specific users. Different from general purpose search engines (e.g., Google), the requirement of personalization for specialist search platforms is even higher that it has been no longer a nice-to-have but an essential need-to-have characteristic. Take the recruitment platform as an example, users from a diverse range of backgrounds are interested in different job types and levels. An example of job search scenario is presented in Figure 1. If two users both search for "Apple Inc.", a Ph.D. graduate in computer science would probably be interested in "Apple research engineer", while an experienced designer is more likely to be looking for positions like "Apple senior graphic designer". In this search, the Ph.D. graduate subsequently clicked on works related to Apple software engineer and ignored the work of Apple internship and designer, which indicates that his/her next search would be more inclined to engineer. After searching "Apple engineer", the user clicked on the work related to Apple engineer and research engineer but skipped the job of senior engineer that required work experience, indicating that the user's next search would probably shift to the direction of research engineer. For the experienced designer, after the first search, he/she clicked on jobs of designer, which suggests that he/she wanted to find a designer position. The second query of this user is indeed "Apple designer". Then, he/she clicked on a job entitled Apple graphic designer, indicating that he/she has some graphic design's experience. Then the third query of the designer is "Apple senior graphic designer". In this case, without knowing any personal information and searching context of the job seeker, a query suggestion system would probably recommend queries irrelevant to the user search intent, leading to poor user experiences and customer losses [59].

Apart from the challenges above, query suggestion systems also suffer from the data sparsity problem [6, 25, 28, 37]. To address this issue, one of the main strategies is to cluster queries into denser groups and share information within the clusters [4, 8, 9, 18, 31, 35, 41, 53]. Some other researchers attempt to incorporate statistical features into query suggestion models to learn user reformulation behaviors better and partially alleviate the sparsity problem [39, 42, 43]. Another category of techniques that have been developed to address the sparsity issue is deep learning approaches. Here, query suggestion systems based on techniques of recurrent neural networks [25, 29, 44], memory networks [51], and knowledge graphs [22] have been presented. Even though the sparsity problem in query suggestions has attracted increasing attention and promoted some intriguing studies, it remains a tricky issue hurting the performance of query suggestion systems.

To overcome the aforementioned limitations of existing techniques for query suggestions, we present a novel framework of

Dynamic Searching Flow Model (DSFM) to fulfil search tasks in e-recruiting contexts. Here we emphasize the value of user feedback flow within search sessions as well as the rich semantic information extracted from resumes and job descriptions for more accurate and personalized query recommendations. To capture the dynamic user search intent behind queries in each search session, we employ a dynamic flow mechanism, inspired by a recent work in dialog understanding [30]. The graphical representation of our mechanism in job search scenario is presented in Figure 2. The basic underlying assumption here is that the difference between two successive queries in semantic space is derived from the most recent user search intent, user profile, and contextual information of job descriptions. We also introduce *local flow* and *global flow* to depict the real-time and overall user intention, respectively. This design enables the DFSM to learn the immediate search interest captured by recent feedback and the long-term search need extracted from historical behaviors in the entire session simultaneously. Considering the different levels of influence might be brought by search actions occurred at different time points during a session to the current search intent modeling, attention mechanism is adopted. In addition, we propose a weighted contrastive learning module to mitigate the effect of data sparsity and obtain more meaningful negative samples based on the similarities among users.

Our main contributions can be summarized as follows:

- We propose a novel query suggestion framework that shows superiorities in modeling dynamic user search intent and shift of intention naturally by exploiting immediate and historical implicit user feedback within the search session as signals using more sophisticated refinement techniques.
- Leveraging additional personal information about users with rich textual context supplied, the proposed model is able to effectively capture the deep semantics of search needs and suggest queries that match particular user specification.
- We devise a more intelligent negative sampling scheme to address the sparsity problem in query suggestion systems via the weighted contrast learning.
- We show that the DFSM model significantly improves the performance of query recommendations compared with competitive baseline methods on real-world datasets in online recruitment setting.

2 RELATED WORK

In this section, we discuss related work from certain angles of mainstream approaches and personalization techniques in existing query suggestion systems.

2.1 Retrieval-based Query Suggestion

Query suggestion is a key function of search engines that allows them to help users refine queries by recommending a list of relevant ones that are more expressive than the anchor query initially created by users [40, 47]. To do so, prior work in the field of query suggestions primarily rely on two strategies, which are retrieval-based and generation-based methods. Among the retrieval-based strategies, ranking approaches leveraging query co-occurrence [20, 23] and discriminative chracterization are considered as the most effective ways for traditional query suggestions [1, 39]. More recently, motivated by the success of deep neural networks in multiple retrieval tasks [7, 19, 21, 34], various deep learning algorithms have been developed for query suggestions. For instance, Chen et al. [13] devised an attention-based hierarchical neural network to detect hidden dependencies between quires and users for query suggestions. Ahmad et al. [2] proposed a context-aware neural retrieval model that enables the joint optimization of two companion tasks of document ranking and query suggestions. Although effective, these retrieval-based strategies are inherently restricted by what is existing in the search logs, and thus suffer from the lack of appropriate candidates for tail queries [15].

2.2 Generation-based Query Suggestion

To overcome the limitations of some retrieval-based query suggestion approaches, such as the lack of semantic understanding of queries and low coverage for rare or unseen queries [59], and to produce effective continuations of the user search intent, generation-based strategies have been pursued by regarding the query suggestion as a natural language generation problem [11]. In this line of research, the Sequence-to-Sequence (Seq2Seq) model [46] serves as one of the most widely used query generation techniques [15, 26, 45, 49, 58]. Specifically, the Seq2Seq model enables this functionality by employing an encoder-decoder architecture, in which word semantics are summarized through the encoder and queries are generated via the decoder [38]. Apart from this, query suggestion systems employing other state-of-the-art algorithms have also been developed, including reinforcement learning [6], dual learning [40], and knowledge graph [32]. These query generation approaches have shown their effectiveness in helping users explore fresh opportunities [11, 38]. However, they usually ignore the role that implicit user feedback might play in query suggestion process and further aggravate the ambiguity problem [11, 51, 55].

2.3 Personalized Query Suggestion

Although delivering personalized query suggestions to users is a critical need for search engines, relatively few previous studies have attempted to enhance the personalization performance of query suggestion systems, resulting in weak personalization a common drawback for both retrieval-based query suggestions and vani-lla Seq2Seq-based query generation approaches [38]. Early personalization efforts [14, 60] focused primarily on the analysis of individual user data stored locally, which makes these models can hardly learn shared features among users and thus suffer from severe data sparsity problem. Then Mei et al. [35] presented a method that allows personalized query generation by employing a bipartite graph. More recent publications on personalized query suggestions tended to pay increased attention to query semantics. For instance, Yin et al. [56] devised a transformer-based hierarchical encoder architecture to model different types of user search behaviors through multiple distinct views, and thus enhanced the personalization performance of the system.

3 PRELIMINARIES

Given the limitations of current query suggestion systems as introduced above, the main goal of this study is to fill in these gaps and present a novel integrated framework that allows personalized and fair query suggestions by taking full advantage of dynamic user feedback and auxiliary information. Since the primary application scenario of our proposed approach is online recruitment, this section offers a statement concerning how the specific research problem is formulated and defined.

3.1 Problem Statement

Imagine on a recruitment platform, the query suggestion task is fulfilled through the interactions between the search engine and a user by keeping formulating queries, clicking queries, examining, and clicking search results until his/her job hunting need is satisfied. During this process, a series of interactive behaviors, e.g., query reformulation and clicking on the jobs recommended, often exhibit strong inter-dependence. Clicked or not clicked jobs potentially provide rich context information for the system to understand user search intent and improve their query suggestion performance.

Apart from this, as the job information required and searching behavior patterns vary significantly from person to person, modeling personalized suggest context before using it in query suggestion tasks is the key to unleashing its vast potential. We have already presented an example in Figure 1 and discussed this in Section 1. Hence, we argue that query suggestion systems should recommend queries in the region that match the user's resumes and searching context.

Furthermore, the distribution of queries in the corpus is usually long-tailed, which would lead to the phenomenon that the query suggestion system always recommends queries that are relatively more popular to users. In addition, users generally tend to click on jobs offered by well-known companies, even though their requirements may not match those jobs. In this case, it is necessary to select and add unseen query samples in the training process to alleviate the problems of exposure bias and low matching rate.

Based on these thoughts, we propose DFSM, a query suggestion framework that is capable of modeling and refining user search intent progressively in recruitment scenarios by leveraging a personalized module, a dynamic flow mechanism, and weighted contrastive learning to solve the problems mentioned above.

3.2 Definition

Before we zoom into the details of our model, we first specify the definitions of several important concepts and the notations. Consider a user u, whose profile is denoted as U, is typing a query q_{cur} in the search box to apply for a job. Before the current query, the recent queries searched by the user from the latest to the earliest are $Q = \{q_1, q_2, ..., q_k\}$, and the jobs listed by the recruitment platform after every search are $J = \{J_1, J_2, ..., J_k\}$, where each J_i contains a pair of $\{(j_1^+, j_2^+, ..., j_{m_1}^+), (j_1^-, j_2^-, ..., j_{m_2}^-)\}$. Here j^+ are jobs browsed by u in search i, while j^- are those ignored by u. The main target is: Given a set of candidate Q_{can}, how to design a ranking model that can predict the rank of each query $q_{can} \in Q_{can}$ conditioned on the user, the current query, and sequential behaviors of searching and browsing with relevant job descriptions provided.

4 METHODOLOGY

Our proposed learning framework tackles the problem of simply concatenating recommendation and long-tailed distribution in query suggestion tasks by modeling searching flow via weighted contrastive learning. Figure 3 presents an overview of our framework.

4.1 Model Overview

Our framework mainly consists of a **Personalized Module** and a **Flow Module**. Specifically, we process U and J in Personalized Module by 1) Multi-View Encoder of Users and Jobs. After that, all queries, including searched queries, current queries, and candidate queries are encoded by 2) Query-level Encoder. Then, we get a combination of q, U, and J by 3) Combining the User Information, and 4) Combining the Job Description. In Flow Module, we utilize job descriptions to obtain the embedding of a session according to 5) Searching Dynamic Flow, and finally 6) Matching session embedding and suggesting query. Next, we discuss the details of each module in our approach.

4.2 Personalized Module

In the personalized module, we first design an encoder of the user, job, and query, and combine the query representation in every time step with user representation and job representation to get a personalized representation of user intention.

Multi-View Encoder of User and Job. To initialize the embedding of users and jobs, we first represent the context using pre-trained RoBERTa to initialize the text-based input, including the user resume and job descriptions. We then separately concatenate the text embedding with other grouped discrete characteristics of them. Subsequently, we get the context-aware user representation h_u and job representation h_j.

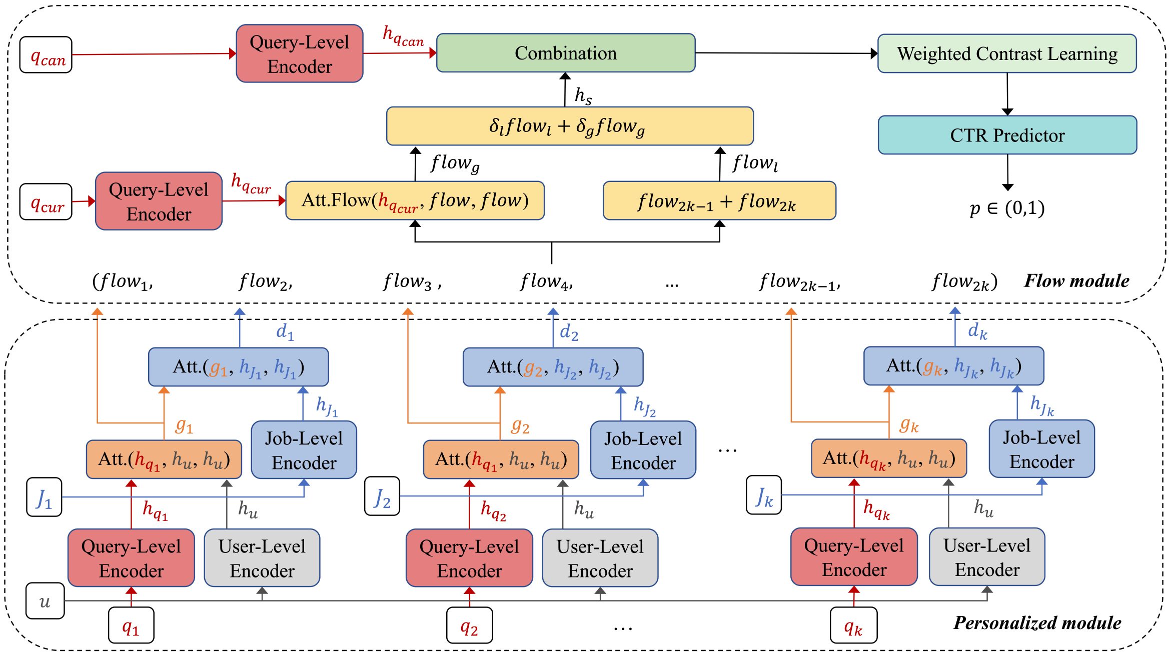
Query-level Encoder. Given a training instance with historical queries, searched queries, and suggestion candidate queries, we use the query encoder to derive their representations h_{q_i} and $h_{q_{can}}$. Here, we use the same encoder to get the representation of two types of queries, since it is more convenient for us to recommend queries according to their similarities.

Combine the User Information. In DFSM, we first encode personalized intent depending on historical queries and the user's profile. To reward features that pinpoint the search intent of the user, user level attention mechanism is applied. Specifically, we obtain historical personalized query embedding $G = \{g_1, g_2, ..., g_k\}$ of the user by:

$$\alpha_{ui} = \frac{exp(h_{q_i} h_u^T)}{\sum_k exp(h_{q_i} h_u^T)} \tag{1}$$

$$g_i = \alpha_{ui} h_u. \tag{2}$$

Combine the Job Description. To introduce the concept of searching dynamic flow in our paper, we first define some symbols. In a session, the semantic difference between previous g_{i-1} and new g_i is denoted as d_{i-1}. Thus $D = \{d_1, d_2, ..., d_k\}$ are semantic influences in this session. We leverage J to extract D from the user's interaction with listed jobs. Intuitively, the user browse jobs that he/she is interested in and ignore others. To accurately extract the user's searching intention behind listed jobs, we try to retain different information between positive jobs J_i^+ and negative ones



q_{can}
Query-Level Encoder
$h_{q_{can}}$
Combination
Weighted Contrast Learning
h_s
$\delta_l flow_l + \delta_g flow_g$
CTR Predictor
$flow_g$
$flow_l$
q_{cur}
Query-Level Encoder
$h_{q_{cur}}$
Att.Flow($h_{q_{cur}}$, flow, flow)
$flow_{2k-1} + flow_{2k}$
$p \in (0,1)$
($flow_1$, $flow_2$, $flow_3$, $flow_4$, ... $flow_{2k-1}$, $flow_{2k}$)
Flow module
d_1
Att.(g_1, h_{J_1}, h_{J_1})
g_1
h_{J_1}
Att.(h_{q_1}, h_u, h_u)
Job-Level Encoder
J_1
h_{q_1}
h_u
Query-Level Encoder
User-Level Encoder
u
q_1
d_2
Att.(g_2, h_{J_2}, h_{J_2})
g_2
h_{J_2}
Att.(h_{q_1}, h_u, h_u)
J_2
h_{q_2}
q_2
d_k
Att.(g_k, h_{J_k}, h_{J_k})
g_k
h_{J_k}
Att.(h_{q_k}, h_u, h_u)
J_k
h_{q_k}
q_k
Personalized module

Figure 3: The detailed architecture of the Dynamic Searching Flow Framework. The model consists of personalized module and flow module. Our key novelty is to encode information in query suggestion and search context into dynamic context-attentive representations to facilitate query suggestion task.

J_i^- and remove the similar part. Therefore we can get the direction of offset between g_{i-1} and g_i. Specifically, we measure the similarity between J_i^+ and J_i^- and obtain the DisMask in every time step by:

$$DisMask = 1 - CosineSimilarity(h_{J_m^+}, h_{J_i^-}). \tag{3}$$

$$h_{J_i^+} = \frac{1}{m_1}\sum_{m_1} h_{j_m^+}, h_{J_i^-} = \frac{1}{m_2}\sum_{m_2} h_{j_m^-}. \tag{4}$$

Then we multiply averaged positive jobs' embedding by this DisMask to get searching intentions s_i in the time step i:

$$s_i = \frac{1}{m_1}\sum_{m_1} DisMask * h_{J_i^+}, \tag{5}$$

and we finally get the semantic influences by:

$$\alpha_{gi} = \frac{exp(g_i s_i^T)}{\sum_k exp(g_i s_i^T)} \tag{6}$$

$$d_i = \alpha_{gi} g_i \tag{7}$$

4.3 Flow Module

As mentioned in 4.2, now we have the output $\{g_1, g_2, ..., g_k\}$ and $\{d_1, d_2, ..., d_k\}$ from personalized module. Note that g_i denotes search intent extracted by current query and the user's profile, d_i denotes the user intention extracted from job descriptions after each search, we organize them and name them as searching dynamic flow, before feeding them into the flow module. Here attention mechanism is applied to calculate the weight of each g_i and d_i.

Searching Dynamic Flow. Consider a user who is searching for a suitable job, his/her searching intention is highly related to both g_i and d_i extracted by Personalized Module. Hence, we need to leverage them to obtain the dynamic information and denote the Searching Dynamic Flow simply by:

$$\begin{aligned} flow &= \{flow_1, flow_2, ..., flow_{2k-1}, flow_{2k}\} \\ &= \{g_1, d_1, g_2, d_2, ..., g_k, d_k\}. \end{aligned} \tag{8}$$

Suppose that when making a new query, the user draws inspiration both from his/her last search and historical search. Here we separately user local flow and global flow to extract these two types of intention.

At time step k, g_k means the last query's personalized representation and d_k means the semantic difference between g_k and the clicked query, which is the ground truth in a sample. Then *local flow* can be defined as:

$$flow_l = g_k + d_k. \tag{9}$$

local flow $flow_l$ represents real-time intention, which means during the process of the last search, the user observes some intriguing information. Thus he/she shifts his/her intention and turns to search for another query relevant or irrelevant to historical queries.

Moreover, we introduce *global flow* to describe the overall influence of the session. The flow-level embedding $flow_i$ of a flow is defined as:

$$flow_i = \frac{1}{l_q}\sum_{j=1}^{l_q} flow_{i,j}, \tag{10}$$

Since not all search contribute equally to the user's intention, attention mechanism is applied to extract such informative contributing terms in the flow and aggregate the embeddings of these terms to construct a *global flow* vector.

$$\alpha_{fi} = \frac{exp(h_{q_{cur}} flow_i^T)}{\sum_{2k} exp(h_{q_{cur}} flow_i^T)}, \tag{11}$$

$$flow_g = \sum_{2k} \alpha_{fi} * flow_i, \tag{12}$$

the final embedding of session s is:

$$h_s = \delta_l * flow_l + \delta_g * flow_g, \tag{13}$$

where δ_l and δ_g are parameters trained with the model. Note that:

$$\delta_g = 1 - \delta_l. \tag{14}$$

Matching session representation and suggest query. Given the instance of a session s and the candidate query q_{can}, we aim to explicitly derive the matching score between s and q_{can}. To learn from the supervision of click behaviors, we introduce a CTR prediction task following [57]. In this task, we first leverage the attention method to get the matching score between s and q_{can}:

$$\alpha_{q_{can}} = \frac{exp(h_s h_{q_{can}}^T)}{\sum_{l_q} exp(h_s h_{q_{can}}^T)}. \tag{15}$$

Following approaches [28], we define the output of this matching score layer as a probability $p(s, q_{can})$ representing the click-through rate (CTR) of the specific candidate as:

$$p(s, q_{can}) = h_s^T W_{sim}(\alpha_{q_{can}} h_{q_{can}}), \tag{16}$$

where l_q in Eq. 15 is the length of a query. To map p into the range of (0,1), we select sigmoid as the activation function.

4.4 Training Objective

Different from previous studies which used query-query pair only or simply concatenated queries embedding in the session, our model is trained with all the information flow, including queries and searching context. Correspondingly, we design Weighted Contrast Learning to create more query samples and utilize these samples to solve the problem of long-tailed distribution and exposure bias.

Weighted Contrastive Learning. Exposure bias happens as users are only exposed to a part of specific items so that unobserved interactions do not always represent negative preference [10]. In our recruitment system, users are only exposed to top-10 queries and their historical queries are often have long-tailed distribution. Hence, weighted contrastive learning is introduced in our model to solve the problem of exposure bias and long-tailed distribution. Based on the consideration that two user behaviors are probably tend to be the same if their profiles have high similarity, we construct a weighted contrastive learning mechanism in the following way: first, we calculate the similarities between current user embedding and others, then select top n users and add their candidate queries as negative samples after filter the ground truth query; they are also combined with randomly selected queries which have not been clicked as negative samples. Then we use these samples to optimize the following point-wise ranking loss function:

$$\mathcal{L}_{WCL} = \frac{1}{\sum_n w_r} \sum_n w_r * (y \log p(x) + (1 - y) \log(1 - p(x)), \tag{17}$$

Table 1: The statistics of dataset in different categories.

Dataset	Categories		
	Technology	Sales	Design
#Resumes	34,669	44,602	16,854
#Jobs	1,859,967	2,430,880	1,073,002
#Queries	352,346	438,509	208,138
Average Negative Jobs	2.8	3.0	2.9
Average Positive Jobs	8.4	9.1	8.7
Average Session Length	5.6	6.1	5.8

where w_r denotes the similarity between current user and user rank r selected by the method mentioned above.

Searching Flow Modeling. To describe the impact of subsequent information on the next query, DFSM predicts the searching flow based on the user's clicks and browses. To better capture the semantic shift during searching process, we minimize the Euclidean distance between the predicted semantic influence and real semantic influence as follows:

$$\mathcal{L}_{SFM} = \sum_n ||d_k - (g_{k+1} - g_k)||_2^2. \tag{18}$$

Considering Eq.17 and Eq.18, we eventually denote the overall training objective of DSFM as:

$$\mathcal{L} = \mathcal{L}_{WCL} + \mathcal{L}_{SFM}. \tag{19}$$

5 EXPERIMENT

5.1 Dataset

The largest online recruiting platform named "BOSS Zhipin" (the BOSS Recruiting)[1] in China provides a large real-world dataset for us to evaluate our model. Since the original amount of data is huge, we randomly sample a fraction of the entire data, containing user resumes, input queries, candidate queries, and job descriptions within a session. To test the robustness of our model for different domains, the original dataset is partitioned into three subsets, including Technology, Sales and, Design. It should be noted that there is an overlap between users in different categories, because users could enter at most three domains for their desired industries. We preprocess the data by removing characters and spelling errors that do not conform to Chinese grammar rules. Following [28], we segment the queries into sessions by a heuristic method, i.e., an idle interval of at least 30 minutes represents a session boundary. Then, sessions with only one query session are filtered out because they do not contain subsequent job context information and Average Session Length are counted. We also use Average Negative Jobs to denote the average number of jobs viewed but not clicked by a user after searching a query and Average Positive Jobs to denote the average number of jobs the user clicked. Table 1 shows a brief statistic of datasets. As we can see: (1)Data in different categories have dissimilar characteristics, e.g. Sales is a large dataset, Technology is medium in size with less Negative Jobs and Positive Jobs, and Design is the smallest one with similar

[1]https://www.zhipin.com

characteristics to Sales; (2)For each category, after the search of each query, there are a certain amount of Positive Jobs and Negative Jobs, and the proportion of Positive Jobs is high. As discussed before, the user's behavior for each item after searching the query helps predict the user's next search. Our study attempts to model the effects of both positive and negative items, which enables users to know which jobs they want or do not want to employ.

5.2 Experiment Setup

Each session is combined with user resume, a series of queries, a group of candidate queries, and job descriptions which are divided into positive group and negative group, meaning whether the person interacts with this job information or not after this search. We only select top-10 queries because this is the amount that a user can see in one search. The ratio of the positive samples and negative samples for the candidate queries is 1:9, representing each session is correspond to one positive query which is the ground truth and nine negative queries. In this paper, we apply a heuristic way to generate a session that for a user, if the time interval between two searches is greater than 30 minutes, the latter search is regarded as in the new session. Furthermore, to partition search sessions into training and test sets, the first 90% data are utilized for training, and the remaining data are used for the test. Among the training data, 10% of data are randomly sampled as validation data for parameter fine-tuning.

5.3 Evaluation

For evaluation, we follow the previous setting [12, 15, 28] and employ Mean Reciprocal Rank(MRR)[48], Precision at position 1(P@1), Normalized Discounted Cumulative Gain when selecting top-5 or top-10 queries(N@5, N@10) as evaluation metrics. Given candidate queries Q, let $Y(Q)$ be a ranked list of recommended queries determined by a method. We use $rank(Y(Q))$ to denote the rank of the clicked query in $Y(Q)$.

5.4 Baseline Methods

The following six classic baselines are selected. We will introduce the reason why we choose these baseline methods and then describe each in detail. First, We consider a popularity-based traditional method MPS and a traditional machine learning method GRM as our classical IR baselines for query suggestion. To compare DFSM with neural ranking models, we also select some neural network-based methods. BHM is selected to compare our model's performance with a generation-based method. DESTINE is selected to test our model's performance in personalization. CFAN is selected to verify that the design of searching dynamic flow is a benefit to the overall performance. And M^2A is a personalized method with state-of-art performance in query suggestion task. These methods are described in detail below:

Most Popular Suggestion(MPS) [15, 45], a simple yet effective baseline is a maximum likelihood method. It ranks the queries based on co-occurrence frequencies of the last query in the search session.

GBDT-based Ranking Model(GRM) [52], based on GBDT to rank the candidate queries. Each value of resumes, jobs, and queries are preprocessed to features as input to the model. We represent the prefix, context, and query candidate with Bag-of-Words (BoW) vectors, then train a LambdaMART model with these features.

Behavioral Hypotheses Model(BHM) [12], presents an encoder-decoder model that includes behavioral hypotheses. It also leverages tokenwise attention to aggregate multiple behavior hypotheses encoded by a shared Transformer encoder BART.

DisentanglEd Self-atTentIve NEtwork(DESTINE) [54], a framework explicitly decouples the computation of unary feature importance from pairwise interaction. We use DESTINE to build user embedding and job embedding, then concatenate them with the query-level Transformer encoder to get the representation of a session.

Click Feedback-Aware Network(CFAN) [28], considers user clicks on previously suggested queries as the user feedback, and improves the robustness of the query suggestion system through adversarial training.

Multi-View Multi-Task Attentive Approach(M^2A) [57], a multi-view multi-task attentive framework to learn personalized query auto-completion models. It combines CTR prediction task and query generation task and obtains the state-of-art performance in public dataset.

5.5 Implementation Details

Our model and the six baselines are all implemented in Python. GRM is implemented using a open-source learning-to-rank framework[2]. We train our framework using the following implementation details. We embed features (except text) of resumes and job informations, set the dimension of embedding by empirical value, and initialize them with the standard normal distribution. For the textual features in the user resumes and job descriptions, we separately use 4-layers 512-dim pre-trained Chinese RoBERTa[33] encoder provided by huggingface[3]. The maximum length of text in profiles and job descriptions are 50 and 100 respectively. All embedding of text in historical queries, user profile, and job description is with the dimension of 512 and vocabulary size of 21,128. In order to align the dimension, we use different MLP layers to process and the concatenation of discrete characteristics (only in user profiles and job descriptions) and text embeddings, and get the final representations of user profiles, job descriptions and queries. Activation functions are ReLU for previous layers and sigmoid for the final prediction. Each query in the input queries and the candidates are restricted to a maximum length of 10. The sample number of weighted contrast learning is 5. Truncation and zero padding are applied if required. We optimize the models using the Adam optimizer [27] and set the mini-batch size to 128 for the model. The learning rate is set to 1e-3, with the learning rate exponential decay applied. For the framework, we train the model for 6 epochs. We update the network parameters with the query suggestion model parameters after each epoch.

6 RESULT AND DISCUSSION

6.1 Experimental Results

Table 2 summarizes the results of baseline models and DFSM on different categories of Dataset from the BOSS Zhipin platform. By

[2]https://github.com/jma127/pyltr
[3]https://huggingface.co

Table 2: Performance comparison between baselines Experiments on different datasets.

Dataset	Technology				Sales				Design			
Metric	MRR	P@1	N@5	N@10	MRR	P@1	N@5	N@10	MRR	P@1	N@5	N@10
MPS	0.442	0.263	0.494	0.580	0.453	0.256	0.484	0.579	0.494	0.293	0.501	0.589
GRM	0.473	0.268	0.503	0.593	0.474	0.269	0.505	0.594	0.503	0.301	0.514	0.593
BHM	0.529	0.347	0.564	0.633	0.501	0.331	0.543	0.627	0.517	0.336	0.545	0.621
DESTINE	0.556	0.383	0.586	0.664	0.544	0.370	0.572	0.658	0.545	0.377	0.576	0.656
CFAN	0.554	0.380	0.581	0.663	0.557	0.383	0.586	0.665	0.555	0.382	0.584	0.664
M^2A	0.561	0.389	0.595	0.671	0.559	0.384	0.589	0.667	0.561	0.388	0.593	0.669
DSFM	**0.579**	**0.407**	**0.605**	**0.675**	**0.575**	**0.403**	**0.602**	**0.673**	**0.601**	**0.432**	**0.613**	**0.697**

Table 3: Experiments on different categories of Datasets by removing each module of out model to evaluate their benefits to overall performance.

Dataset	Technology				Sales				Design			
Metric	MRR	P@1	N@5	N@10	MRR	P@1	N@5	N@10	MRR	P@1	N@5	N@10
Mq + Mf + Mc	0.553	0.377	0.586	0.663	0.551	0.375	0.580	0.662	0.562	0.387	0.599	0.678
Mp + Mf + Mc	0.547	0.371	0.575	0.660	0.543	0.369	0.571	0.657	0.566	0.388	0.581	0.678
Mp + Mq + Mc	0.564	0.390	0.594	0.669	0.563	0.389	0.593	0.668	0.574	0.400	0.610	0.673
Mp + Mq + Mf	0.569	0.394	0.598	0.672	0.566	0.390	0.596	0.670	0.587	0.406	0.605	0.681
DFSM	**0.579**	**0.407**	**0.605**	**0.675**	**0.575**	**0.403**	**0.602**	**0.673**	**0.601**	**0.432**	**0.613**	**0.697**

utilizing dynamic information in the sessions, DFSM achieves the highest scores in all evaluation indices on all datasets.

Generally, Neural Network methods BHM, DESTINE, CFAN, and M^2A outperform the traditional framework by a large margin. The generation-based method BHM which assumes four behavioral hypotheses performs the worst among all deep learning-based frameworks, indicating that the pure generation-based method generates some ambiguous queries in the task of query suggestion. DESTINE is a framework that explicitly decouples the computation of unary feature importance from pairwise interaction, and the improvement from BHM to DESTINE demonstrates that it is important to accurately organize and utilize user information and job description in query suggestions. The framework CFAN utilizes more context including input queries and search queries, and introduces adversarial learning to improve the overall performance. It has better performance than DESTINE, indicating that introducing extra information and adversarial learning benefits the model. Moreover, the multi-view multi-task attentive method M^2A is slightly better than CFAN. This is mainly because M^2A proposes a behavior-level transformer encoder to precisely leverage user behavior.

On Design, the performance of MPS is closer to GRM than that on Technology and Sales. We believe that this is because the occupation distribution of the design industry is more niche and concentrated than that in the sales and technology industry. As we can see, our framework DSFM offers a significant improvement on Design over other baselines. This implies that our model can better handle smaller and more centralized datasets.

On all the datasets, DFSM outperforms baseline methods, which demonstrates the effectiveness of our proposed model. The rich text contains more user information, job description, and click feedback, which helps deduce the query ambiguity and thus contributes to the ranking performance.

6.2 Ablation Analysis and Discussions

In this section, we construct a series of ablation experiments and quantitative analyses to show how the components of DFSM contribute to the query suggestion task. DFSM is a complicated model with multiple views. To evaluate the impact of each module on the overall performance, we conduct an ablation study in this section. We partition our model into four sections: 1)personalized module depending on the user's profile - M_p, 2)searched queries view - M_q, 3)flow view based on the design of dynamic searching flow - M_f, 4)contrastive learning view - M_c. We separately train the model by removing M_p, M_q, M_f or M_c. The results of the ablation study without each view are reported in Table 3 and discussed next.

Benefit of Personalized Module. To understand the impact of the personalized module which combines queries, user information, and job description, we alternatively strip off the personalized module discussed in section 4.2, and simply use a series of query representations as the input of attention mechanism. As presented in the first block of Table 3 on the three datasets, DFSM loses in all the metrics. This result suggests that modeling user intention by personalized information is crucial for the query suggestion task.

Benefit of Queries and Flow Module. To evaluate the impact of modeling sessions and searching context, we compare the origin model, the model without searched queries view, and the model without dynamic searching flow. Note that the model without dynamic searching flow is trained with only L_{WCL} as its training objective. As presented in the fourth and fifth rows in Table

Table 4: Examples of query suggestions from different models depends on the same historical queries and current input. Fanno is a cross-border e-commerce platform, Weibo is a social platform, all other terms include Baidu, Meituan, Alibaba and ByteDance are well-known Internet enterprises in China.

queries	Fanno Baidu group Meituan operation Alibaba group ByteDance editing ByteDance group
current query	e-sports
ground truth	**e-sports operation**
MPS	e-sports, e-sports hotel, e-sports club
GRM	e-sports hotel, e-sports, e-sports club
BHM	e-sports group, e-sports editing, e-sports
CFAN	e-sports, **e-sports operation**, Weibo e-sports
DESTINE	e-sports hotel, e-sports, e-sports club
M^{2}A	e-sports, e-sports club, **e-sports operation**
DFSM	**e-sports operation**, e-sports, e-sports event

3, without modeling searched queries, DFSM loses 5.5%, 5.6% and 5.8% in MRR; without modeling dynamic searching flow, it loses 2.6%, 2.1% and 4.5% in MRR, respectively. This result clearly shows that both modeling queries and searching context are important. Furthermore, the modeling of past queries is more crucial for the query suggestion task.

Benefit of Weighted Contrastive Learning. As it is necessary to verify the importance of weighted contrast learning during the training process, we remove the weight of Eq. 17 and only train the model with new training objective. As presented in the last row of Table 3, DFSM loses 1.7%, 1.6% and 2.3% in MRR. Although there is a slight decline, it illustrates that weighted contrastive learning promotes the overall performance of our model.

6.3 Case Study

Query suggestion aims to assist the user in formulating queries to express their search intention. Therefore, it is necessary to suggest personalized queries according to users' resumes and behaviors because different people may have diverse intentions when they are typing the same word. The success of a query suggestion system lies in accurately understanding and modeling user search intent and recommending relevant queries depending on different intentions. However, we observe that current search engines tend to recommend popular queries and rank them high in the suggested list, even when these fashionable queries are not relevant to users' intention. In Table 4, we show a case of query suggestion when the user is typing "e-sports" in the search box. Generally, our model is better than baseline methods in two aspects:

Personalized. Some popular words often appear in the recommendation of the baseline methods, which are not relevant to the user intention. To be specific, when the user is typing "e-sports" in the search box, "e-sports", "e-sports hotel" and "e-sports club" are the top-3 popular queries according to the popularity-based method MPS. According to Table 4, in the top-3 suggestions of baseline methods, three popular queries mentioned above often appear. GRM and DESTINE simply suggest these three queries by different ranking, BHM suggests two unusual queries "e-sports group" and "e-sports editing" since it is a generation-based method. Although CFAN and M^{2}A suggest the ground truth, "e-sports" still ranks first. DFSM can suggest some different but more relevant queries like "e-sports operation" according to the user profile and interactions with listed jobs, then brings the user a more personalized experience.

Real-time Relevance. In this case, it is hard to suggest queries relevant to "operation" because all the historical queries searched by the user are broad and related to company names. Being difficult to extract the user's real intention when searching "e-sports operation", CFAN and M^{2}A suggest this ground truth in the second and the third place. Although it seems impossible to accurately estimate the semantic influences according to information listed in Table 4, we observe that the user clicks 12 jobs, among which only 4 are not relevant to the topic of operation. For example, there are sentences from one job description: "The successful candidate should have a strong background in **operation**. He/She should also have the ability to provide corresponding **operation** strategies according to the characteristics of customer groups in the responsible region." These results imply that DFSM can precisely predict the user's intention from historical queries and his/her behavior after each search.

7 CONCLUSION

In this paper, we propose a context attentive ranking model in the task of query suggestion by modeling user intention for online recruitment. It models user intention by explicitly utilizing user profiles, previous queries, and behaviors with jobs from an ongoing search. A personalized module and a flow module are devised to learn dynamic information. We also introduce the concepts of local flow and global flow to capture real-time and long-term search intention simultaneously. Extensive experimentation demonstrates the effectiveness of the proposed user intention modeling approach. The value of each functional component in the system to the query suggestion task is also analyzed and highlighted.

ACKNOWLEDGEMENT

Thanks for the anonymous reviewers for their insightful comments. This work was supported by Natural Science Foundation of China (NSFC Grant No. 62122089, No. 61876196 and No. 6210021830), Beijing Outstanding Young Scientist Program No. BJJWZYJH0120191-00020098, 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 RUC, the Fundamental Research Funds for the Central Universities, and the Research Funds of Renmin University of China (22XNKJ24). Rui Yan is supported by Beijing Academy of Artificial Intelligence (BAAI).

REFERENCES

[1] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. 2018. Multi-task learning for document ranking and query suggestion. In *International Conference on Learning Representations.*

[2] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. 2019. Context attentive document ranking and query suggestion. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 385–394.

[3] Hiteshwar Kumar Azad and Akshay Deepak. 2019. Query expansion techniques for information retrieval: a survey. *Information Processing & Management* 56, 5 (2019), 1698–1735.

[4] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. 2004. Query recommendation using query logs in search engines. In *International conference on extending database technology.* Springer, 588–596.

[5] Ranieri Baraglia, Carlos Castillo, Debora Donato, Franco Maria Nardini, Raffaele Perego, and Fabrizio Silvestri. 2009. Aging effects on query flow graphs for query suggestion. In *Proceedings of the 18th ACM conference on Information and knowledge management.* 1947–1950.

[6] Praveen Kumar Bodigutla. 2021. High Quality Related Search Query Suggestions using Deep Reinforcement Learning. *arXiv preprint arXiv:2108.04452* (2021).

[7] Alexey Borisov, Martijn Wardenaar, Ilya Markov, and Maarten de Rijke. 2018. A click sequence model for web search. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 45–54.

[8] Huanhuan Cao, Daxin Jiang, Jian Pei, Enhong Chen, and Hang Li. 2009. Towards context-aware search by learning a very large variable length hidden markov model from search logs. In *Proceedings of the 18th international conference on World wide web.* 191–200.

[9] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 875–883.

[10] Jiawei Chen, Hande Dong, Xiang Wang, fuli Feng, Meng Wang, and Xiangnan He. 2022. Bias and Debias in Recommender System: A Survey and Future Directions. *arXiv preprint arXiv:2010.03240* (2022).

[11] Ruey-Cheng Chen and Chia-Jung Lee. 2020. Incorporating Behavioral Hypotheses for Query Generation. *arXiv preprint arXiv:2010.02667* (2020).

[12] Ruey-Cheng Chen and Chia-Jung Lee. 2020. Incorporating Behavioral Hypotheses for Query Generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing.* 3105–3110.

[13] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten de Rijke. 2018. Attention-based hierarchical neural query suggestion. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 1093–1096.

[14] Paul-Alexandru Chirita, Claudiu S Firan, and Wolfgang Nejdl. 2007. Personalized query expansion for the web. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval.* 7–14.

[15] Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to Attend, Copy, and Generate for Session-Based Query Suggestion. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management.* 1747–1756.

[16] Magdalini Eirinaki and Sweta Patel. 2015. QueRIE reloaded: Using matrix factorization to improve database query recommendations. In *2015 IEEE International Conference on Big Data (Big Data).* IEEE, 1500–1508.

[17] Apostolos Glenis and Georgia Koutrika. 2021. PyExplore: Query Recommendations for Data Exploration without Query Logs. In *Proceedings of the 2021 International Conference on Management of Data.* 2731–2735.

[18] Jiafeng Guo, Xueqi Cheng, Gu Xu, and Xiaofei Zhu. 2011. Intent-aware query similarity. In *Proceedings of the 20th ACM international conference on Information and knowledge management.* 259–268.

[19] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM international on conference on information and knowledge management.* 55–64.

[20] Chien-Kang Huang, Lee-Feng Chien, and Yen-Jen Oyang. 2003. Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of the American Society for Information Science and Technology* 54, 7 (2003), 638–649.

[21] 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 *Proceedings of the 22nd ACM international conference on Information & Knowledge Management.* 2333–2338.

[22] Zhipeng Huang, Bogdan Cautis, Reynold Cheng, and Yudian Zheng. 2016. Kb-enabled query recommendation for long-tail queries. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management.* 2107–2112.

[23] Alpa Jain, Umut Ozertem, and Emre Velipasaoglu. 2011. Synthesizing high utility suggestions for rare web search queries. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval.* 805–814.

[24] Bernard J Jansen, Amanda Spink, and Tefko Saracevic. 2000. Real life, real users, and real needs: a study and analysis of user queries on the web. *Information processing & management* 36, 2 (2000), 207–227.

[25] Jyun-Yu Jiang and Wei Wang. 2018. RIN: Reformulation Inference Network for Context-Aware Query Suggestion. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management.* 197–206.

[26] Michaeel Kazi, Weiwei Guo, Huiji Gao, and Bo Long. 2020. Incorporating User Feedback into Sequence to Sequence Model Training. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management.* 2557–2564.

[27] Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *ICLR* (2015).

[28] Ruirui Li, Liangda Li, Xian Wu, Yunhong Zhou, and Wei Wang. 2019. Click Feedback-Aware Query Recommendation Using Adversarial Examples. In *Proceedings of the 2019 World Wide Web Conference.*

[29] Xiangsheng Li, Yiqun Liu, Xin Li, Cheng Luo, Jian-Yun Nie, Min Zhang, and Shaoping Ma. 2018. Hierarchical attention network for context-aware query suggestion. In *Asia Information Retrieval Symposium.* Springer, 173–186.

[30] Zekang Li, Jinchao Zhang, Zhengcong Fei, Yang Feng, and Jie Zhou. 2021. Conversations Are Not Flat: Modeling the Dynamic Information Flow across Dialogue Utterances. *arXiv preprint arXiv:2106.02227* (2021).

[31] Zhen Liao, Daxin Jiang, Enhong Chen, Jian Pei, Huanhuan Cao, and Hang Li. 2011. Mining concept sequences from large-scale search logs for context-aware query suggestion. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 1 (2011), 1–40.

[32] Matteo Lissandrini, Davide Mottin, Themis Palpanas, and Yannis Velegrakis. 2020. Graph-query suggestions for knowledge graph exploration. In *Proceedings of The Web Conference 2020.* 2549–2555.

[33] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692* (2019).

[34] Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. *Advances in neural information processing systems* 26 (2013).

[35] Qiaozhu Mei, Dengyong Zhou, and Kenneth Church. 2008. Query suggestion using hitting time. In *Proceedings of the 17th ACM conference on Information and knowledge management.* 469–478.

[36] Tova Milo and Amit Somech. 2020. Automating exploratory data analysis via machine learning: An overview. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data.* 2617–2622.

[37] Agnès Mustar, Sylvain Lamprier, and Benjamin Piwowarski. 2021. On the study of transformers for query suggestion. *ACM Transactions on Information Systems (TOIS)* 40, 1 (2021), 1–27.

[38] Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. *arXiv preprint arXiv:1704.04572* (2017).

[39] Umut Ozertem, Olivier Chapelle, Pinar Donmez, and Emre Velipasaoglu. 2012. Learning to suggest: a machine learning framework for ranking query suggestions. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval.* 25–34.

[40] Kunxun Qi, Ruoxu Wang, Qikai Lu, Xuejiao Wang, Ning Jing, Di Niu, and Haolan Chen. 2021. Dual Learning for Query Generation and Query Selection in Query Feeds Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management.* 4065–4074.

[41] Eldar Sadikov, Jayant Madhavan, Lu Wang, and Alon Halevy. 2010. Clustering query refinements by user intent. In *Proceedings of the 19th international conference on World wide web.* 841–850.

[42] Rodrygo LT Santos, Craig Macdonald, and Iadh Ounis. 2013. Learning to rank query suggestions for adhoc and diversity search. *Information Retrieval* 16, 4 (2013), 429–451.

[43] Milad Shokouhi. 2013. Learning to personalize query auto-completion. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval.* 103–112.

[44] 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 *proceedings of the 24th ACM international on conference on information and knowledge management.* 553–562.

[45] 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 *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.* 553–562.

[46] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems.* 3104–3112.

[47] Saedeh Tahery and Saeed Farzi. 2020. Customized query auto-completion and suggestion — A review. *Information Systems (87)* (2020).

[48] Ellen M. Voorhees. 1999. The TREC-8 Question Answering Track Report. In *Proceedings of The Eighth Text REtrieval Conference.*

[49] Xiao Wang, Craig Macdonald, and Iadh Ounis. 2020. Deep Reinforced Query Reformulation for Information Retrieval. *arXiv preprint arXiv:2007.07987* (2020).

[50] Ji-Rong Wen, Jian-Yun Nie, and Hong-Jiang Zhang. 2001. Clustering user queries of a search engine. In *Proceedings of the 10th international conference on World Wide Web*. 162–168.

[51] Bin Wu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2018. Query suggestion with feedback memory network. In *Proceedings of the 2018 World Wide Web Conference*. 1563–1571.

[52] Qiang Wu, J.C. Christopher Burges, Krysta Marie Svore, and Jianfeng Gao. 2010. Adapting boosting for information retrieval measures. *IR(2010)* (2010), 1645–1648.

[53] Wei Wu, Hang Li, and Jun Xu. 2013. Learning query and document similarities from click-through bipartite graph with metadata. In *Proceedings of the sixth ACM international conference on Web search and data mining*. 687–696.

[54] Yichen Xu, Yanqiao Zhu, Feng Yu, Qiang Liu, and Shu Wu. 2021. Disentangled Self-Attentive Neural Networks for Click-Through Rate Prediction. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 3553––3557.

[55] Dawei Yin, Yuening Hu, Jiliang Tang, Tim Daly, Mianwei Zhou, Hua Ouyang, Jianhui Chen, Changsung Kang, Hongbo Deng, Chikashi Nobata, et al. 2016. Ranking relevance in yahoo search. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 323–332.

[56] Di Yin, Jiwei Tan, Zhe Zhang, Hongbo Deng, Shujian Huang, and Jiajun Chen. 2020. Learning to Generate Personalized Query Auto-Completions via a Multi-View Multi-Task Attentive Approach. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2998–3007.

[57] Di Yin, Jiwei Tan, Zhe Zhang, Hongbo Deng, Shujian Huang, and Jiajun Chen. 2020. Learning to Generate Personalized Query Auto-Completions via a Multi-View Multi-Task Attentive Approach. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2998–2007.

[58] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 31.

[59] Jianling Zhong, Weiwei Guo, Huiji Gao, and Bo Long. 2020. Personalized Query Suggestions. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1645–1648.

[60] Zhengyu Zhu, Jingqiu Xu, Xiang Ren, Yunyan Tian, and Lipei Li. 2007. Query expansion based on a personalized web search model. In *Third International Conference on Semantics, Knowledge and Grid (SKG 2007)*. IEEE, 128–133.