Front. Comput. Sci.
DOI 10.1007/s11704-017-6518-6

REVIEW ARTICLE

User behavior modeling for better Web search ranking

Yiqun LIU (✉), Chao WANG, Min ZHANG, Shaoping MA

Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

Abstract Modern search engines record user interactions and use them to improve search quality. In particular, user click-through has been successfully used to improve click-through rate (CTR), Web search ranking, and query recommendations and suggestions. Although click-through logs can provide implicit feedback of users' click preferences, deriving accurate absolute relevance judgments is difficult because of the existence of click noises and behavior biases. Previous studies showed that user clicking behaviors are biased toward many aspects such as "position" (user's attention decreases from top to bottom) and "trust" (Web site reputations will affect user's judgment). To address these problems, researchers have proposed several behavior models (usually referred to as click models) to describe users? practical browsing behaviors and to obtain an unbiased estimation of result relevance. In this study, we review recent efforts to construct click models for better search ranking and propose a novel convolutional neural network architecture for building click models. Compared to traditional click models, our model not only considers user behavior assumptions as input signals but also uses the content and context information of search engine result pages. In addition, our model uses parameters from traditional click models to restrict the meaning of some outputs in our model's hidden layer. Experimental results show that the proposed model can achieve considerable improvement over state-of-the-art click models based on the evaluation metric of click perplexity.

Keywords user behavior, click model, Web search

Received October 28, 2016; accepted March 27, 2017

E-mail: yiqunliu@tsinghua.edu.cn

1 Introduction

Learning from user feedback is a popular approach used in modern search engines. Commercial search engines usually record large-scale user interaction logs every day, and many research issues related to Web search (e.g., click prediction, Web search ranking, query suggestion) are closely related to these behavior logs.

Although user clicks provide implicit information about user's perceived relevance on results, they do not represent true and accurate relevance feedback. Therefore, various methods have been proposed to cope with the noisy nature of user clicks. Joachims et al. [1] worked on extracting reliable implicit feedback from user behaviors and concluded that click logs are informative yet biased. Previous studies revealed several biases such as "position" [1,2], "trust" [3], and "presentation" [4] factors. To address these issues, researchers have proposed a number of click models to describe user behavior with respect to search engine result pages (SERPs) and to obtain an unbiased estimation of result relevance [5–7]. Through this kind of estimation, a search engine can achieve better search ranking either through reranking or incorporating the estimation as a ranking signal.

Most existing click models are formulated within the framework of a probabilistic graphic model. In these models, a group of variables are typically used to model each search result for a specific query. The variables include the observable click actions and some hidden variables such as user examination, result relevance, and user satisfaction after the result is viewed. Different click models make different user behavior assumptions (e.g., cascade assumption [2]) to construct a network structure for variables. Once constructed, these click models can be trained on a large set of user click-

through logs and can then be used to predict click probabilities for results or to rerank the search result list according to the inferred relevance.

Although these click models have achieved much success in click/relevance prediction for Web and sponsored searches, many limitations exist with the framework of probabilistic graphic model. The main limitation is that simple assumptions (e.g., cascade assumption) used to structure graphic models cannot account for complex user behaviors in a modern Web search, which are influenced by many other factors. Liu et al. [8] showed that behavior patterns are very different when users encounter different result presentation styles. Wang et al. [9] showed that in a majority of practical search sessions, users do not follow the top-down examination sequence as assumed in most existing models. If a probabilistic graphic model tries to consider all of these observed behavior patterns, it must add many new variables and edges. However, all such edges must be linked with explicit user behavior assumptions, even though designing realistic relationship with less obvious influence factors is difficult. For this reason, most existing click models only consider certain "important" user behavior assumptions and ignore others.

Among the factors ignored are result content and context features, which strongly influence user behaviors. The content information is usually referred to as caption information. Previous studies [10,11] showed that search users rely on this type of information to determine whether they should click on a result. They also found that the content of snippets affects user judgments on clicks. Regarding context information, it has been well documented [12,13] that search results are not independent; they can be similar or redundant. This factor considerably affects user click decisions. Despite the importance of these factors, few previous studies have attempted to incorporate them. Wang et al. [14] are among the first to consider content information for click modeling and have shown the effectiveness of combining both content and behavior information. Unfortunately, their work does not consider the relationships between different search results on the same SERP (result context information), and further extending their model is difficult.

The limitations of graphic models have clearly prevented us from incorporating more useful factors into click models. Therefore, we must use an alternative framework that is more flexible. In this work, we turn to the framework of deep neural networks (DNNs) [15]. A DNN effectively maps the meaning of one object (e.g., a search result) to a continuous representation space. In particular, it has been recently shown that convolutional neural networks (CNNs) can efficiently achieve promising results in many NLP tasks based on learning from large input signals [16,17]. However, a deep learning framework is sufficiently flexible to incorporate various types of input signals without their relationships being manually defined. In fact, DNNs can be trained in an end-to-end manner. Thus, we can simply combine all possible influential features without expending considerable effort in feature engineering.

Some recent studies have successfully used CNN to model click behaviors during question-answering and sponsored-search tasks [18,20]. These CNN-based models predict relevance based on content. They use words in queries and results to generate a feature vector representation and then use a CNN framework to learn the relevance scores. These works demonstrated that a hierarchical semantic structure embedded in a query and document can be extracted using DNNs, and based on this, a good relevance/click prediction can also be learned.

Inspired by these studies, we propose a novel click model framework for general Web search based on a CNN architecture. The proposed model tries to combine information sources of result content, context information, and user behavior patterns to better predict click behaviors. We use distributional sentence models generated from content and context information as the basic input of CNN. These underlying sentence models work in parallel by mapping one query and ten corresponding results to their distributional vectors. The vectors are then used to learn the semantic similarity between them. User behavior patterns are then adopted as additional features and incorporated into a click model layer under certain regularization constraints to output click probabilities.

Our contributions in this study are:

- We review recent efforts both in modeling user click behaviors based on probabilistic graphic model frameworks and in improving search result ranking based on DNNs.
- Different from traditional probabilistic graphic model structures, a novel click model construction framework based on CNN is proposed. The proposed model incorporates result content, context, and user behavior information to model complex Web-search user behaviors.
- The proposed framework can be used to reconstruct most existing click models by introducing behavior assumptions into the hidden layer as regularization constraints, which leads to improved performance over the original models.

The remainder of the paper is organized as follows. Recent

efforts in constructing click models and adopting DNNs for behavior modeling are reviewed in Section 2. In Section 3, we formally introduce our new behavior model framework. In Section 4, we report our experiments on the proposed model and compare it with existing click models. Finally, we present the conclusion of our study and discuss future work in Section 5.

2 Recent efforts in click model constructions

2.1 Position-based click models

Most click models adhere to the following examination hypothesis [2]: a document that is clicked on ($C_i = 1$) should satisfy ($\rightarrow$) two conditions: it has been examined ($E_i = 1$) and it is relevant ($R_i = 1$) (most click models assume $P(R_i = 1) = r_u$, which is the probability of the perceived relevance). These two conditions are independent of each other.

$$C_i = 1 \rightarrow E_i = 1, R_i = 1. \tag{1}$$

$$E_i = 0 \rightarrow C_i = 0. \tag{2}$$

$$R_i = 0 \rightarrow C_i = 0. \tag{3}$$

Following this assumption, the probability that a document will be clicked on is determined as follows:

$$P(C_i = 1) = P(E_i = 1)P(R_i = 1). \tag{4}$$

As position-based click models do not consider click sequence, the click action is simply mapped to each search result's ranking position. Based on the assumption that a user's examination proceeds from a top to a bottom position, this type of click model naturally takes position bias into account.

Craswell et al. [2] proposed the cascade model, which assumes that as a user examines results from top to bottom sequentially, he or she immediately determines whether to click on a result:

$$P(E_1) = 1. \tag{5}$$

$$P(E_{i+1} = 1|E_i = 1, C_i) = 1 - C_i. \tag{6}$$

Here, the examination of the $(i + 1)$th result indicates the ith result has been examined but not clicked. Although the cascade model performs well in predicting click-through rates, this model is applicable only to a single-click scenario.

Extending the cascade hypothesis, the dependency click model (DCM) [5] tries to model user interactions within multi-click sessions. DCM assumes that a certain probability exists that a user will examine the next document after clicking on the current document, and this probability is influenced by the ranking position of the result. The DCM model is characterized as follows:

$$P(E_{i+1} = 1|E_i = 1, C_i = 0) = 1, \tag{7}$$

$$P(E_{i+1} = 1|E_i = 1, C_i = 1) = \lambda_i, \tag{8}$$

where λ_i represents the preservation probability of the position i.

The user browsing model (UBM) [6] further refines the examination hypothesis by assuming that the event of a document being examined depends on both the preceding click position and the distance between the preceding and current click positions:

$$P(E_i = 1|C_{1\cdots i-1}) = \lambda_{r_i,d_i} \tag{9}$$

where r_i represents the preceding click position and d_i is the distance between the current rank and r_i.

The dynamic Bayesian network model (DBN) [7] is the first to consider presentation bias due to snippet (rather than ranking position). This model distinguishes the actual from the perceived relevance, in which the perceived relevance indicates the relevance represented by titles or snippets in SERPs, and the actual relevance is the relevance of the landing page. DBN is characterized by:

$$P(R_i = 1) = r_u, \tag{10}$$

$$P(S_i = 1|C_i = 1) = s_u, \tag{11}$$

$$P(E_{i+1}|E_i = 1, S_i = 0) = \lambda, \tag{12}$$

where S_i indicates whether the user is satisfied with the ith document, s_u is the probability of the event, r_u is the probability of the perceived relevance, and λ represents the probability that the examination process will continue.

The click chain model (CCM) [21] uses Bayesian inference to obtain the posterior distribution of the relevance. In contrast to other existing models, this model introduces skipping behavior. CCM is scalable for large-scale click-through data, and experimental results show that it is effective for low-frequency (also known as long-tail) queries.

Wang et al. [14] proposed the first click model that considers content information, and showed that such information is useful. They proposed a number of patterns to combine content with user behavior information. However, further extending their model beyond cascade browsing assumption is difficult.

In addition to these models based on result positions, the partially sequential click model (PSCM) [9] considers click sequence information. The PSCM model proposes two additional user behavior assumptions based on eye-tracking experiments. The first assumes that although the examination behavior between adjacent clicks can be regarded as locally

unidirectional, users may skip some results while examining a single result at some distance from the current one following a certain direction. The second assumes that between adjacent clicks, users tend to examine search results without direction changes, and this direction is usually consistent with that of clicks.

$$P(C_t|C_{t-1},\ldots,C_1) = P(C_t|C_{t-1}). \tag{13}$$

$$P(C_t = n|C_{t-1} = m) = P(\bar{C}_m = 1,\ldots,\bar{C}_i = 0,\ldots,\bar{C}_n = 1). \tag{14}$$

$$P(\bar{E}_i = 1|C_{t-1} = m, C_t = n) = \begin{cases} \gamma_{imn}, m \leqslant i \leqslant n \text{ or } n \leqslant i \leqslant m; \\ 0, \text{otherwise.} \end{cases} \tag{15}$$

$$\bar{C}_i = 1 \Leftrightarrow \bar{E}_i = 1, R_i = 1. \tag{16}$$

$$P(R_i = 1) = \alpha_{uq}. \tag{17}$$

The first equation encodes the *first-order click hypothesis*, whereas the second encodes the *locally unidirectional examination assumption* by restricting the examination to a one-way process from m to n.

This model distinguishes the result position from the examination order and achieves a better click prediction performance than do position-based click models.

2.2 Temporal click models

In addition to click behaviors, the duration of click dwell time (the time the user spends on a clicked result) has also been regarded as an important feedback feature for relevance estimation because it clearly correlates with result-level satisfaction or document relevance [22–24]. Longer dwell time on a clicked page has traditionally been used to identify satisfied (SAT) clicks. Although click-through statistics can sometimes be misleading due to order and caption biases, click dwell time is a more robust measure.

Click dwell time has been successfully used in a number of retrieval applications (e.g., implicit relevance feedback [25] and re-ranking [26]). In those applications, SAT clicks are simply identified by some predefined time threshold (i.e., a click is SAT if its dwell time equals or exceeds that threshold). A dwell time equal to or exceeding 30s, as proposed in [24], has typically been used to identify clicks with which searchers are satisfied.

Because click dwell time is a critical feedback signal, some recent efforts have also tried to incorporate temporal information into constructing click models.

Xu et al. [27] first proposed a temporal click model (TCM) to model user click behavior for sponsored searches. They enumerated all possible permutations of click sequences for search results. This model can only handle two results/ads in a SERP. This means coping with an entire ranked result list as in other click models is impossible.

Wang et al. [28] introduced a partially observable Markov model (POM) to model arbitrary click orders. The POM model treats user examination events as a partially observable stochastic process. Although POM can model non-sequential behaviors, it only considers the examination transition at each position (i.e., different users and queries share the same examination sequence parameters). Therefore, this model cannot predict the click probability or relevance for a specific query, and thus cannot be used in a practical search environment. Because of this limitation, POM cannot be compared with other state-of-the-art click models such as UBM and DBN, which must predict click probability and relevance for a specific query-URL pair. It also makes the first-order examination assumption that the current examination behavior depends only on its previous examination step, which might not align with the real user behavior.

Xu et al. [29] proposed a temporal hidden click model (THCM) to cope with non-sequential click actions. They focused on revisiting behavior and assumed that, after a user clicks on a search result, a probability exists that he or she will examine previous results (bottom-up). However, their model was also based on a one-order Markov examination assumption and thus supposes that users examine results one by one during the examination process, which does not necessarily correspond to the practical user behavior.

In [30], Liu et al. proposed a time-aware click model (TACM) to extend the PSCM model. They introduced a new hidden state (satisfaction state): $S = \langle S_0, S_1, S_2, \ldots, S_t, \ldots, S_T \rangle$, where each $S_t = 1$ means that, after a user's tth click, he or she has already obtained sufficient information and plans to finish the search process.

$$P(C_t|C_{t-1},\ldots,C_1, S\,t-1,\ldots,S_1) = P(C_t|C_{t-1}, S_{t-1}). \tag{18}$$

$$S_{t-1} = 1 \rightarrow C_t = 0. \tag{19}$$

$$P(C_t = n|C_{t-1} = m) = P(\bar{C}_m = 1,\ldots,\bar{C}_i = 0,\ldots,\bar{C}_n = 1). \tag{20}$$

$$P(\bar{E}_i = 1|C_{t-1} = m, C_t = n) = \begin{cases} \gamma_{imn}, m \leqslant i \leqslant n \; or \; n \leqslant i \leqslant m; \\ 0, \text{otherwise.} \end{cases} \tag{21}$$

$$\bar{C}_i = 1 \Leftrightarrow \bar{E}_i = 1, R_i = 1. \tag{22}$$

$$P(R_i = 1) = \alpha_{uq}. \tag{23}$$

$$P(S_t = 1) = P(R_t = 1) \times F(DwellTime_t). \quad (24)$$

We can see that the first and third equations still follow the *first-order click hypothesis* and *locally unidirectional examination assumption* as in the PSCM model. The second equation shows that the user may stop the browsing process if he or she feels satisfied. TACM also follows the examination hypothesis as described in most existing click models. α_{uq} corresponds to the relevance of the document URL u at position i for the specific query q. We can see that click dwell time information is used to describe user information gains. Four kinds of mapping functions are typically adopted to measure information gain after a certain dwell time: linear, quadratic, exponential, and Rayleigh. Experimental results show that, of the four options, the exponential mapping function works the best.

2.3 Click modeling with neural networks

Unlike click models that utilize user behavior information to predict future clicks or result relevance, the application of DNNs in Web search mainly depends on result content information as the model input, in which the aim is to improve search ranking performance.

Recently, deep learning methods have been successfully applied to a variety of natural language processing (NLP) and information retrieval (IR) tasks [31–33]. By exploiting deep architectures, these techniques can discover the hidden structures in training data. In addition, features at different levels of abstraction can also be discovered for certain tasks. In [31], Salakhutdinov and Hinton extended the LSA model using a deep network (auto-encoder) to discover the hierarchical semantic structure embedded in queries and documents. They proposed a semantic hashing (SH) method that uses bottleneck features learned from the deep auto-encoder for information retrieval. Based on this work, Huang and Shen et al. [18,34] proposed a framework to develop a series of new latent semantic models with deep structures that project queries and results into a common low-dimensional space, where the relevance of a document given a query is regarded as the distance between the document and the query. Their deep structured semantic models are discriminatively trained by maximizing the conditional likelihood of the clicked documents given a query derived from click logs.

Although this model has performed well in relevance prediction, extending it to search environments that use non-Latin characters (e.g., Chinese, Japanese, or Korean (CJK)) is difficult. This is because the word hashing process is based on letter n-grams that cannot be implemented for non-Latin characters. Therefore, Liu et al. [20] proposed a new convolutional click prediction model (CCPM) for sponsored searches. This model can extract key local-global features from an input instance having varied elements and can be implemented for not only single but also sequential advertisement impressions. To the best of our knowledge, this is the first approach that attempts to leverage a CNN to determine click probability. However, this work still fails to consider the context information of results, which, according to many existing studies such as [12,13], is critical for user decision-making processes in search interactions. Moreover, unlike most models, this approach cannot predict result relevance and user examination information.

Zhang et al. [35] proposed a click prediction framework based on recurrent neural networks (RNN). This framework is designed for sponsored search and directly models the dependency on user sequential behaviors into the click prediction process through the recurrent structure in RNN. Borisov et al. [36] also proposed an RNN-based click model to model user sequential click behaviors. These models only consider click sequence information and ignore the effect of different click dwell times among click actions.

Severyn and Moschitti [19] proposed a deep learning architecture for reranking short texts. This model uses existing word embedding results to build a feature matrix for each query and answer. The authors then use the approach from [16] to compute the similarity of the query-answer pairs. The advantage of this model is that its architecture easily enables including additional features to improve learning performance. However, this framework cannot consider user behavior patterns or assumptions to improve learning performance.

Unlike these existing efforts in click model construction and IR-oriented deep learning studies, we choose the CNN as the framework for the new click model and use it to combine information sources from result content, context, and user behavior. The distinctive properties of our model are: (i) unlike existing works that can predict only single units of information (either click probability or result relevance), our framework can predict click probability, result relevance, and user examination information as efficiently as in traditional click models; (ii) we include the parameters learned from traditional click models as additional features and constraints in order to combine content information and user behavior patterns; (iii) unlike with existing solutions based on CNNs [19,20] that use point-wise learning strategies, we learn 10 results from a single SERP using a pair-wise strategy. This approach proves to be more effective than previous learning-

to-rank studies.

3 DNN-based click model

This section describes our DNN-based click model. Its main building blocks are $T + 1$ distributional sentence models based on CNNs (1 for query, and T for corresponding results, where, in this study, T equals 10, representing the number of results on a SERP). These underlying sentence models work in parallel to map queries and documents to vectors, which are then used to learn the semantic similarity between each query-result and result-result pair. In this work, we use the titles of search results to represent their content, as incorporating the entire content is inefficient and may introduce too many noises.

The architecture of the proposed CNN architecture for mapping sentences to feature vectors is mainly inspired by the architectures used in [17,19,37] for performing various sentence classification tasks. The network (see Fig. 1) is composed of a single wide convolutional layer followed by nonlinearity and simple max pooling. Following the pooling process, the generated query-feature and result-feature vectors are used to compute content and context similarities. All features are combined with their behavior information and then grouped and mapped by the joint and hidden layers. Finally, the click model layer uses these features to generate the examination probability and relevance estimation for each result and then predict click probabilities. The network input are raw words that must be translated into real-valued feature vectors to be processed by subsequent layers of the network. This process is implemented using word embedding techniques. The framework of our model is shown in Fig. 1. We provide a detailed explanation of the main components of the proposed model as follows.

Unlike the models constructed in [35,36], that described in Fig. 1 adopts a CNN instead of an RNN structure. This is because focusing on the inter-relationship between results and queries is desirable, as this information is not considered by previous click models. RNN models capture the behavior sequential information in user sessions, but that model may have problems modeling sessions in which only one or a few clicks occur.

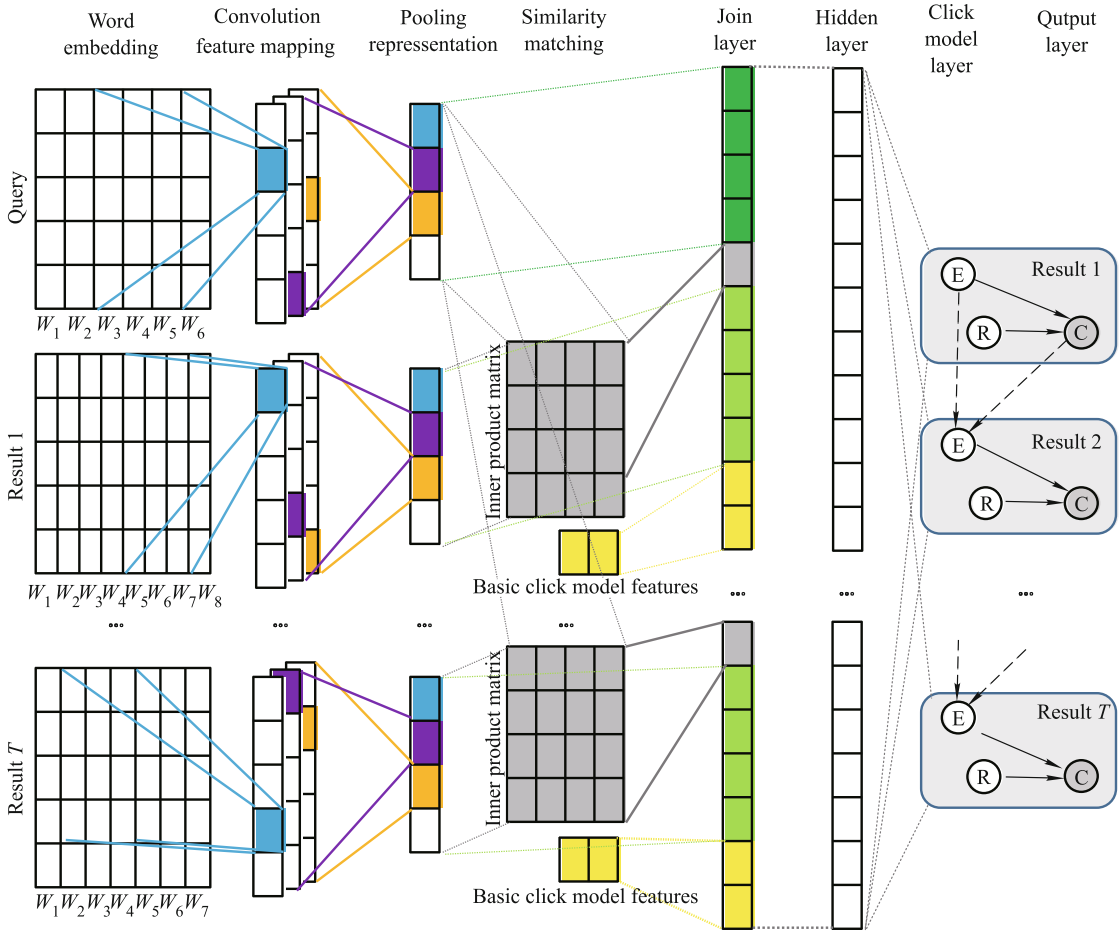


Fig. 1 Architecture for the proposed DNN-based click model

3.1 Sentence matrix construction

Several methods are used to generate content-based features for a given query or result. Huang et al. [18] proposed a word-hashing method that is based on the letter n-grams of a given text. Liu et al. [20] proposed a word-embedding method that uses embedding vectors to represent each word of a given textual document. As we mentioned in Section 2, the letter n-gram method is not well adapted to non-Latin language environments. Considering the fact that nearly all major commercial search engines provide services in CJK languages, choosing a word-embedding method to generate a sentence matrix is applicable.

In the proposed framework, one sentence s is treated as a sequence of words: $[w_1, w_2, \ldots, w_{|s|}]$, where each word w is drawn from a vocabulary V. Words are represented by distributional vectors W looked up in a word-embedding matrix $W \rightarrow |V|$ that is formed by concatenating the embeddings of all words in V. In our study, the word-embedding dataset is trained by a commercial search engine company that utilizes more than 10 million Web pages with open-source tools developed by Mikolov et al. [38]. The dimension of each word vector was $n = 100$ in our experiments.

For convenience in lookup operations in W, words are mapped to integer indices $1, 2, \ldots, |V|$. For each input sentence s, we build a sentence matrix S, where each value v_{ij} in column j and row i represents the ith element of a word-embedding vector w_j at the corresponding position j in a sentence:

$$S = \begin{Bmatrix} v_{00} & \cdots & v_{s0} \\ \vdots & & \vdots \\ v_{0n} & \cdots & v_{sn} \end{Bmatrix}. \tag{25}$$

The aim of the convolutional layer is to extract patterns (i.e., discriminative word sequences found within input sentences) that are common throughout the training instances. We use the *wide* type to compute the boundary in the sentence matrix as done in Kalchbrenner et al. [37]. The benefits of the *wide* type is that wide convolution can better handle words at boundaries, thereby giving equal attention to all words in the sentence. This is in contrast to narrow convolution, where words that are near boundaries are seen fewer times. More importantly, wide convolution also guarantees that valid values are always generated even when s is shorter than the filter size m.

To enable the network to learn non-linear decision boundaries, each convolutional layer is typically followed by a non-linear activation function $\alpha()$ applied element-wise to the output of the preceding layer. Nair and Hinton [39] showed that a rectified linear unit has considerable benefits over *sigmoid* and *tanh* by overcoming some of their shortcomings. Therefore, we choose to use a rectified linear unit $\max(0, x)$ in our proposed framework.

3.2 Pooling

For most DNN implementations, two common choices are usually employed in the pooling operation: average and max. Both operations are applied to columns of the feature mapping matrix by mapping them to a single value. The max pooling method is used more widely and does not suffer from the drawbacks of weakening strong activation values [40].

Because the purpose of our model is to test the effectiveness of combining CNN with click models, we test only the max pooling strategy in our model. Future work will include the testing of other pooling methods such as k-max pooling [37].

3.3 Content and context similarity calculation

In the previous process, our model learns to map input sentences to vectors, which can then be used to compute the similarities between the query and results. Therefore, we use the vectors generated from the pooling step to compute the query-result similarity and result-result context influence. Supposing that each SERP contains T results ($T = 10$ in general), we can calculate T query-result similarity scores and $T \times (T-1)/2$ result-result influence scores.

Given the vectors of our model after processing queries and results in the pooling step, the resulting vector representations x_q and x_{d_i} for the ith result in SERP can be used to compute a similarity/influence score. We follow the approach of Bordes et al. [41] that defines the similarity between x_q and x_{d_i} as follows:

$$\begin{aligned} Sim(x_q, x_{d_i}) &= x_q^{\mathrm{T}} M_{qr} x_{d_i}, \\ Sim(x_{d_i} x_{d_j}) &= x_{d_i}^{\mathrm{T}} M_{rr} x_{d_j}, \end{aligned} \tag{26}$$

where M_{qr} and M_{rr} are hidden matrices to be trained during the training process. These two equations can be viewed as a model of the noisy channel approach from machine translation, which has been widely used as a scoring model in information retrieval and question answering [42]. In this model, we seek a transformation of the candidate result that is closest to the input query.

3.4 Joint layer

The joint layer is used to incorporate user behavior information into our model framework. For each query and T results in a session, most existing click models can provide the ex-

amination probability and relevance for each result after the training processes. Therefore, we use the examination probability and relevance score as additional features to enrich our model.

The joint layer concatenates all intermediate vectors, the similarity score, and additional features into a single vector:

$$\begin{aligned} x_{joint} &= [x_c; x_m], \\ x_c &= [x_{content}; x_{context}], \\ x_{content} &= [x_q; x_{d_i}; i, j = 1, 2, \ldots, T], \\ x_{context} &= [Sim(x_q, x_{d_i}); Sim(x_{d_i} x_{d_j}); i, j = 1, 2, \ldots, T], \\ x_m &= [E_i; R_i, i = 1, 2, \ldots, T], \end{aligned} \tag{27}$$

where x_c represents the feature from content and context information, and x_m represents the feature from the traditional click model.

This vector is then passed through a fully connected hidden layer, which allows us to model interactions between the components of the joined representation vector. The hidden layer computes the following transformation:

$$\alpha(w_h \times x_{joint} + b) \tag{28}$$

where w_h is the weight vector of the hidden layer and $\alpha()$ is the non-linearity. Our model includes an additional hidden layer immediately preceding the click model layer to enable interactions between the components of the intermediate representation to be modeled. Finally, the output of the hidden layer is fed to the click model layer, which generates the final click probabilities.

3.5 Click model layer

The nodes in the click model layer are divided into two different groups. One group represents examination and the other represents relevance. The final click probability is generated by the *examination hypothesis* [2] that most click models use:

$$P(Click = 1) = P(E = 1) \times P(R = 1). \tag{29}$$

The examination probability and relevance is mapped by a sigmod function from the input feature:

$$P(E = 1) = \frac{1}{1 + e^{-1 \times [\lambda \times (x_c^{\mathrm{T}} \theta_c) + (1-\lambda) \times (x_m^{\mathrm{T}} \theta_m)]}}, \tag{30}$$

$$P(R = 1) = \frac{1}{1 + e^{-1 \times [\lambda' \times (x_c^{\mathrm{T}} \theta'_c) + (1-\lambda') \times (x_m^{\mathrm{T}} \theta'_m)]}}, \tag{31}$$

where $\theta_c, \theta'_c, \theta_m, \theta'_m$ are weight parameters used to combine all feature values, and λ, λ' are weight parameters for balancing the influence of content and click model features.

Although neural networks have a strong capacity for learning complex decision functions, they tend to overfit, especially on small datasets. To mitigate the overfitting problem, we augment the cost function using l2-norm regularization terms for the parameters of the network.

We also adopt another regularization strategy to restrict the meaning of both examination and relevance nodes. We use the input click model features to restrict the output of each node:

$$\begin{aligned} cost = C_e \times \sum [||P(E_i = 1) - x_{E_i}||^2] \\ + C_r \times \sum [||P(R_i = 1) - x_{R_i}||^2], \end{aligned} \tag{32}$$

where $x_{examination}, x_{relevance}$ are examination and relevance features from x_m, and C_e, C_r are hyperparameters to balance the importance of restrictions. In our experiments, we set $C_e = C_r = 0.5$.

The model is trained to minimize the cross-entropy cost function:

$$\begin{aligned} Loss = cost + C||\theta||^2 - \sum [y_i \log P(Click_i = 1) \\ + (1 - y_i) \log(1 - P(Click_i = 1))], \end{aligned} \tag{33}$$

where $y_i = 0, 1$ is the real click information and θ contains all the parameters optimized by the network.

4 Experiments

The previously described model can be built on most probabilistic-graphic-model-based click models. Therefore, we choose two of them to show the effectiveness of our proposed framework. The first is the UBM model, which is a popular click model and shows good performance across different application scenarios [6,7,43]. The second is the PSCM model [9], which considers non-sequential user behaviors and produces good results.

Two groups of experiments were performed to test the effectiveness of our model. First, we evaluated the click model in terms of predicting click probabilities (click perplexity) from search logs, which is a widely adopted metric to evaluate click model performances [6,7,29]. Then, we used the predicted relevance as a signal for document ranking and evaluated each click model's ranking performance using traditional IR metrics (in this study, NDCG [44]).

4.1 Experimental setups

Zhang et al. [35] were among the first to use DNN in click prediction with search engine logs. We implemented this so-

called RNN model as a baseline to which our model is compared. In addition to the features used in the original model [35], we also added features mentioned in Section 3.4 to this RNN-MODEL for a fair comparison. We found that these features can improve the model's performance.

For other baseline models (UBM and PSCM), we used the open source implementations from Wang et al. [9]. Two new click models, DEEP-UBM-ALL and DEEP-PSCM-ALL, were built from UBM and PSCM using our framework as described in Section 3. For the purpose of testing the effectiveness of our feature grouping strategy (using content, context, and user behavior information in different layers), we removed some feature groups to show their impact on the features. The first group used only the content features (i.e., without x_m and $x_{context}$, and referred to as DEEP-UBM-T and DEEP-PSCM-T). The second group used the content and context features (i.e., without x_m and referred to as DEEP-UBM-TC and DEEP-PSCM-TC). Although these models did not employ traditional click model outputs as features (x_m), they still used them as constraints (Section 3.5). Therefore, DEEP-UBM-T and DEEP-PSCM-T (DEEP-UBM-TC and DEEP-PSCM-TC) still managed to predict different click probabilities.

To evaluate the click models, we utilized sample commercial search logs from a popular commercial search engine. We collected eight days worth of logs between April and May 2015, and filtered out the queries with too many or too few sessions, as commonly done in most click modeling studies. Detailed statistics about the dataset can be found in Table 1.

Table 1 Statistics on the query log dataset

#Distinct queries	#Distinct URLs	#Sessions
227,651	1,259,272	6,525,520

4.2 Evaluation of click prediction

We split all query sessions into training, developing, and testing sets at a ratio of 50% : 20% : 30% as done in previous studies [19].

Click perplexity [4] measures the probability of the actual click events occurring during each session and at each position.

$$Perplexity_i = 2^{-\frac{1}{N}\sum_j^N (C_i \log p_i + (1-C_i)\log(1-p_i))}, \qquad (34)$$

where $Perplexity_i$ is the perplexity score in the ith result position, N is the total session count, C_i is the actual user click information, and p_i is the predicted click probability. The overall click perplexity score is the average of all positions (10 in our dataset).

Click perplexity indicates how well a model can predict clicks. A smaller perplexity value indicates a better modeling performance, where the value reaches 1 in the ideal case. The improvement of click perplexity CP_1 over CP_2 is usually calculated as $\frac{CP_2-CP_1}{CP_2-1} \times 100\%$ [4,43].

Table 2 presents the overall click perplexity of each model. Note that all differences are statistically significant based on a t-test with $p\text{-}value < 10^{-5}$. We can see that DEEP-PSCM-ALL achieved the best overall results among all click models and that both deep structured click models (DEEP-PSCM-ALL and DEEP-UBM-ALL) achieved better performance than did the respective baseline models (PSCM and UBM). We also observed that the RNN-MODEL [35] did not achieve a better performance than in the other models. The main reason may be that for most users in our dataset, a login was not required. In addition, forming very long action sequences as in [35] was not possible. Considering that DEEP-RNN used the same dataset and behavior/content features as in our model, we believe that it was a fair comparison and that RNN-MODEL may not work for datasets that do not have long behavior sequences.

Table 2 Overall click perplexity of each model

Model	Perplexity	Improvement/%
RNN-MODEL	1.588	-
PSCM	1.413	-
DEEP-PSCM-T	1.401	+2.8 (VS. PSCM)
DEEP-PSCM-TC	1.399	+3.2 (VS. PSCM)
DEEP-PSCM-ALL	1.013	+96.9 (VS. PSCM)
UBM	1.289	-
DEEP-UBM-T	1.401	–38.8 (VS. UBM)
DEEP-UBM-TC	1.400	–38.3 (VS. UBM)
DEEP-UBM-ALL	1.145	+49.9 (VS. UBM)

We noticed that the PSCM's overall performance was worse than that of UBM, whereas our DEEP-PSCM-ALL model achieved better performance than the DEEP-UBM-ALL model. We conducted further analysis on different result positions. Table 3 shows perplexities in different positions. We noticed that the PSCM model achieved better performance than did the UBM model on top positions (1 and 2). These results were consistent with those shown in [9]. When we examined closely the UBM model's output, we found that the UBM model tended to predict very small click probability for all sessions in bottom positions. Because users rarely click on bottom positions, the perplexity scores for these bottom positions were small. By contrast, PSCM tended to predict different click probabilities for different sessions in bottom positions. Although the PSCM model received a larger perplexity score penalty for those non-click sessions in bottom

positions, it received a much smaller perplexity score penalty for sessions with clicks in bottom positions. Therefore, the PSCM model provided stronger features than did the UBM model, as the features from the UBM model in bottom positions were nearly the same for all sessions. According to Table 3, DEEP-PSCM-ALL achieved the best performance among all positions compared to other click models. We also observed that our proposed framework improved click prediction performance in most positions compared with the baseline click models, especially for top positions.

We also discovered that our model continually performed better as it properly incorporated content, context, and user behavior information (DEEP-X-T, DEEP-X-TC and DEEP-X-ALL). For the UBM model, if we considered only content and context information (DEEP-UBM-T and DEEP-UBM-TC), the performances were worse than in the baseline model (UBM). For the PSCM model, if we considered only content and context information (DEEP-PSCM-T, DEEP-PSCM-TC), the performances were slightly better than in the baseline model (PSCM). As the improvement was minimal, we could conclude that user behavior information was critical and, thus, our model cannot perform competitively with traditional click models when using a simple application of content-based DNN methods. Our model was built to combine content and user behavior information in different layers. Our experimental results revealed that this strategy can achieve a better performance than can existing methods.

To analyze further the influence of incorporating content and context information into click models, we show a case study in Fig. 2 from our experimental dataset. In this case, a user searched for "the official website of China's railway ticketing service" and clicked the top three results in the corresponding SERP. In Fig. 2, we display the top five results and show the examination probabilities estimated by each click model (UBM, DEEP-UBM, PSCM, and DEEP-PSCM). A warmer font color means a higher examination probability. We can see that our proposed models (DEEP-UBM and DEEP-PSCM) predicted much lower examination

Table 3 Click perplexity of different models in different result ranking positions

	1	2	3	4	5	6	7	8	9	10
RNN-MODEL	3.8786	1.8685	1.5566	1.4152	1.3191	1.2356	1.1848	1.1538	1.1360	1.1302
PSCM	1.2720	1.4798	1.4932	1.4780	1.4542	1.4253	1.4012	1.3839	1.3722	1.3657
DEEP-PSCM-ALL	1.0005	1.0009	1.0011	1.0012	1.0014	1.0017	1.0051	1.0203	1.0462	1.0489
Impr. (VS. PSCM)/%	+99.8	+99.8	+99.8	+99.7	+99.7	+99.6	+98.7	+94.7	+87.6	+86.6
UBM	1.7562	1.6006	1.4144	1.3154	1.2362	1.1707	1.1294	1.1024	1.0861	1.0774
DEEP-UBM-ALL	1.0104	1.1647	1.1902	1.2174	1.2341	1.1807	1.1425	1.1143	1.0975	1.0949
Impr. (VS. UBM)/%	+98.6	+72.6	+54.1	+31.1	+0.9	−5.9	−10.1	−11.6	−13.3	−22.6

Fig. 2 Example showing the influence of incorporating content and context information into click models (The third and fourth results are from the same website and have similar titles)

Table 4 Relevance estimation performance for different models

Model	NDCG@3	Improvement/%	NDCG@5	Improvement/%
UBM	0.735	-	0.770	-
DEEP-UBM-ALL	0.747	+1.5(*)	0.797	+1.5(**)
PSCM	0.717	-	0.767	-
DEEP-PSCM-ALL	0.765	+6.7(*)	0.799	+4.1(*)

Note: Paired t-test with $p < 0.05$ is labeled by $*$ and $p < 0.01$ is labeled by $**$

probabilities for the fourth result compared to those from UBM and PSCM. The reason is that the third and fourth results were actually derived from the same website, www.tieyou.com, and used very similar title descriptions ("12306 Train Tickets Online - Train Tickets Online [Official Website of tieyou.com]" VS. "12306 Train Tickets - 12306 Railway Tickets - Train Tickets [Official Website of tieyou.com]"). When our model considered this fact, it generated a much lower examination probability than did the baseline models for the fourth result. The lower examination probability in turn led to a lower click probability, and according to the user's actual click actions, our model actually generated a better click prediction than did the baseline model.

4.3 Evaluation of relevance estimation

The normalized discounted cumulative gain (NDCG, [44]) is a common metric for measuring the performance of ranking algorithms. Because each click model can provide its query-result relevance prediction after training is completed, once we obtained the relevance label for each query-result pair, we could test the ranking performances using NDCG.

Because the click model layer in our model framework restricts the meaning of different nodes and because some nodes are restricted according to the estimated relevance, we used the output from these nodes as the relevance estimation of our model. Because the other baseline model RNN-MODEL does not have such relevance estimation nodes, we only tested the performance of four models in this experiment (UBM, PSCM, DEEP-UBM-ALL, and DEEP-PSCM-ALL).

We randomly sampled 600 queries in our dataset and recruited 11 professional assessors (who did not know the purpose of the work) to annotate the relevance scores of top results for each query. The annotation was performed with five grades ("Perfect," "Excellent," "Good," "Fair" and "Bad") as in most existing studies such as [45]. A pooling strategy was used to ensure that the top five results of each model were all labeled and that median voting was adopted to determine the relevance score if conflicts existed (at least three assessors were involved in each query-result pair annotation and we retained an odd number of assessors for each pair). After removing pornographic queries, we collected the relevance information of 333 distinct queries. Because multiple assessors participated in the labeling process, we used the Fleiss' kappa [46] to measure the agreement among the assessors. The kappa coefficient was 0.270, which means a fair level of agreement. Obtaining a lower agreement when more categories exist is natural. In our case, we had five categories (five grades). Therefore, we believe that this agreement was acceptable. Both the queries and annotated URLs will be available to the public after the double-blind review process.

Using the annotation results, we calculated the NDCG@N (N=3,5) scores for different models, and the results are shown in Table 4. We observed that our proposed models achieved statistically significant improvement in NDCG over the corresponding models. This means that by properly incorporating content and context information into click models, we could not only predict more accurate click probability but also generate a better relevance estimation.

5 Conclusion

In this paper, we focused on improving search ranking performance based on user click-through behaviors. We began by reviewing existing efforts to construct different kinds of click models: position-based models, temporal models, and DNNs. We then proposed a novel framework for click models for general Web search based on CNN architecture. Unlike traditional probabilistic graphic model structures, our model was based on CNN, in which we properly incorporated content, context, and user behavior information in different layers to ensure that high-level user behavior features would not be "buried" by extensive content/context features. Our framework is sufficiently general to be adopted with most existing click models. In this study, we considered two common click models: UBM and PSCM. The experimental results on click-through data showed that our models outperform the existing models in terms of click prediction. We also conducted tests on query-result relevance estimation, and the experimental results showed that our models outperform existing models in terms of relevance estimation (NDCG).

This study proposed a general and flexible means of im-

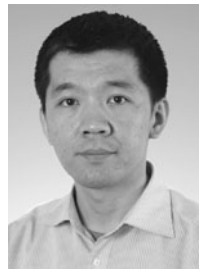
proving click models to fit the ever-changing and increasingly complex search application scenarios. The proposed method can be further improved with respect to several aspects. For example, investigating how it can be adapted to a heterogeneous search environment (modeling the effects of vertical results in the ranking list, considering visual content information, etc.) would be interesting.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos. 61622208, 61732008, 61532011). It is also partly supported by Tsinghua University Initiative Scientific Research Program (2014Z21032) and the National Key Basic Research Program of China (973 Program) (2015CB358700).

References

1. Joachims T, Granka L, Pan B, Hembrooke H, Gay G. Accurately interpreting clickthrough data as implicit feedback. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2005, 154–161
2. Craswell N, Zoeter O, Taylor M, Ramsey B. An experimental comparison of click position-bias models. In: Proceedings of ACM International Conference on Web Search and Data Mining. 2008, 87–94
3. Yue Y S, Patel R, Roehrig H. Beyond position bias: examining result attractiveness as a source of presentation bias in clickthrough data. In: Proceedings of the 19th ACM International Conference on World Wide Web. 2010, 1011–1018
4. Wang C, Liu Y Q, Zhang M, Ma S P, Zheng M H, Qian J, Zhang K. Incorporating vertical results into search click models. In: Proceedings of the 36th ACM International ACM SIGIR Conference on Research and Development in Information Retrieval. 2013, 503–512
5. Guo F, Liu C, Wang Y M. Efficient multiple-click models in web search. In: Proceedings of the 2nd ACM International Conference on Web Search and Data Mining. 2009, 124–131
6. Dupret G E, Piwowarski B. A user browsing model to predict search engine click data from past observations. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2008, 331–338
7. Chapelle O, Zhang Y. A dynamic Bayesian network click model for Web search ranking. In: Proceedings of the 18th ACM International Conference on World Wide Web. 2009, 1–10
8. Liu Z Y, Liu Y Q, Zhou K, Zhang M, Ma S P. Influence of vertical result in Web search examination. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2015, 193–202
9. Wang C, Liu Y Q, Wang M, Zhou K, Nie J Y, Ma S P. Incorporating non-sequential behavior into click models. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2013, 283–292
10. Kaisser M, Hearst M, Lowe J. Improving search results quality by customizing summary lengths. In : Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technlogies (ACLHLT'08). 2008
11. Kanungo T, Orr D. Predicting the readability of short Web summaries. In: Proceedings of the International Conference on Web Search and Web Data Mining. 2009, 325–326
12. Carbonell J, Goldstein J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 1998, 335–336
13. Clarke C L A, Kolla M, Cormack G V, Vechtomova O, Ashkan A, Büttcher S, MacKinnon I. Novelty and diversity in information retrieval evaluation. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2008, 659–666
14. Wang H N, Zhai C X, Dong A L, Chang Y. Content-aware click modeling. In: Proceedings of the 23rd International World-Wide Web Conference. 2013, 175–176
15. Schmidhuber J. Deep learning in neural networks: an overview. Neural Networks, 2015, 61: 85–117
16. Yu L, Hermann K N, Blunsom P, Pulman S. Deep learning for answer sentence selection. In: Proceedings of NIPS Deep Learning and Representation Learning Workshop. 2014, 393–402
17. Kim Y. Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). 2014, 1746–1751
18. Huang P S, He X D, Gao J F, Deng L, Acero A, Heck L. Learning deep structured semantic models for Web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information and Knowledge Management. 2013, 2333–2338
19. Severyn A, Moschitti A. Learning to rank short text pairs with convolutional deep neural networks. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2015, 373–382
20. Liu Q, Yu F, Wu S, Wang L. A convolutional click prediction model. In: Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. 2015, 1743–1746
21. Guo F, Liu C, Kannan A, Minka T, Taylor M J, Wang Y M, Faloutsos C. Click chain model in Web search. In: Proceedings of International Conference on World Wide Web. 2009, 11–20
22. Buscher G, Van Elst L, Dengel A. Segment-level display time as implicit feedback: a comparison to eye tracking. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2009, 67–74
23. Smucker M D, Clarke C L A. Time-based calibration of effectiveness measures. In: Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2012, 95–104
24. Fox S, Karnawat K, Mydland M, Dumais S, White T. Evaluating implicit measures to improve Web search. ACM Transactions on Information Systems, 2005, 23(2): 147–168
25. White R W, Kelly D. A study on the effects of personalization and task information on implicit feedback performance. In: Proceedings of the 15th ACM International Conference on Information and Knowledge Management. 2006, 297–306
26. Agichtein E, Brill E, Dumais S. Improving Web search ranking by incorporating user behavior information. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. 2006, 19–26

27. Xu W H, Manavoglu E, Cantu-Paz E. Temporal click model for sponsored search. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2010, 106–113
28. Wang K S, Gloy N, Li X L. Inferring search behaviors using partially observable Markov (POM) model. In: Proceedings of the 3rd ACM International Conference on Web Search and Data Mining. 2010, 211–220
29. Xu D Q, Liu Y Q, Zhang M, Ma S P, Ru L Y. Incorporating revisiting behaviors into click models. In: Proceedings of the 5th ACM International Conference on Web Search and Data Mining. 2012, 303–312
30. Liu Y Q, Xie X H, Wang C, Nie J Y, Zhang M, Ma S P. Time-aware click model. ACM Transactions on Information Systems, 2016, 35(3): 24–34
31. Salakhutdinov R, Hinton G. Semantic hashing. International Journal of Approximate Reasoning, 2009, 50(7): 969–978
32. Socher R, Huval B, Manning C D, Ng A Y. Semantic compositionality through recursive matrix-vector spaces. In: Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. 2012, 1201–1211
33. Tur G, Deng L, Hakkani-Tür D, He X D. Towards deeper understanding: deep convex networks for semantic utterance classification. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing. 2012, 5045–5048
34. Shen Y L, He X D, Gao J F, Deng L, Mesnil G. Learning semantic representations using convolutional neural networks for Web search. In: Proceedings of the 23rd International Conference on World Wide Web. 2014, 373–374
35. Zhang Y Y, Dai H J, Xu C, Feng J, Wang T F, Bian J, Wang B, Liu T Y. Sequential click prediction for sponsored search with recurrent neural networks. In: Proceedings of the 28th AAAI Conference on Artificial Intelligence. 2014, 133–134
36. Borisov A, Markov I, de Rijke M, Serdyukov P. A neural click model for Web search. In: Proceedings of the 25th International Conference on World Wide Web. 2016, 531–541
37. Kalchbrenner N, Grefenstette E, Blunsom P. A convolutional neural network for modelling sentences. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics. 2014
38. Mikolov T, Sutskever I, Chen K, Corrado G S, Dean J. Distributed representations of words and phrases and their compositionality. In: Proceedings of the Neural Information Processing Systems Conference. 2013, 3111–3119
39. Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on Machine Learning. 2010, 807–814
40. Wan L, Zeiler M, Zhang S X, Cun Y L, Fergus R. Regularization of neural networks using dropconnect. In: Proceedings of the 30th International Conference on Machine Learning. 2013, 1058–1066
41. Bordes A, Weston J, Usunier N. Open question answering with weakly supervised embedding models. In: Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2014, 165–180
42. Echihabi A, Marcu D. A noisychannel approach to question answering. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics. 2003, 16–23
43. Chen D Q, Chen W Z, Wang J X, Chen Z, Yang Q. Beyond ten blue links: enabling user click modeling in federated web search. In: Proceedings of the 5th ACM International Conference on Web Search and Data Mining. 2012, 463–472
44. Järvelin K, Kekäläinen J. Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems, 2002, 20(4): 422–446
45. Yang H, Mityagin A, Svore K M, Markov S. Collecting high quality overlapping labels at low cost. In Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2010, 459–466
46. Landis J R, Koch G G. The measurement of observer agreement for categorical data. Biometrics, 1977, 33(1): 159–174



Yiqun Liu is an associate professor at the Department of Computer Science and Technology, Tsinghua University, China. His major research interests are in Web search, user behavior analysis, and natural language processing. He serves as co-Editor-in-chief of Frontiers and Trends of Information Retrieval (FnTIR), Program Co-chair of SIGIR2018, Short Paper Co-chair of SIGIR2017, Program Co-chair of NTCIR-13, General Co-chair of AIRS2016 as well as (senior) program committee members of a number of important international academic conferences including SIGIR, WWW, AAAI, ACL and IJCAI. In 2016, he was supported by NSFC as an Outstanding Young Scholar (2017–2019).

Chao Wang is now working as a researcher at Baidu.com since he obtained his PhD degree from TsinghuaUniversity, China in 2016. His major research interests are in Web search and user behavior analysis. He has published a number of high quality papers in top-tier academic conference and journals such as SIGIR, CIKM and TOIS. He also received the best paper honorable mention award of SIGIR2015

Min Zhang is an associate professor in the Department of Computer Science and Technology, Tsinghua University, China. She specializes in Web search and recommendation and Web user modeling. Currently she is also the vice director of State Key Lab. of Intelligent Technology and

Systems, the executive director of Tsinghua University-Microsoft Research Asia Joint Research Lab on Media and Search. She also serves as associate editor for the ACM Transaction on Information Systems (TOIS), Program co-Chair of WSDM 2017 and AIRS 2016, area chairs or senior PC members at SIGIR, CIKM, and PC members at WWW, IJCAI, KDD, AAAI, ACL, etc. She has published more than 70 papers in important international journals and conferences, and 12 of her patents are filed. She was awarded Beijing Science and Technology Award (First Prize) in 2016.

Shaoping Ma is a professor in the Department of Computer Science and Technology, Tsinghua University, China. He is also the director of "Tsinghua-Sogou" Joint Lab on Web search technology research, acting vice director of “Tiangong” Research Institute of Intelligent Computing of Tsinghua University and vice president of CAAI. He is interested in the research areas of intelligent information processing, information retrieval, and Web data mining. He has been recently focusing on improving search performance with the help of semantic mining of user behaviors.