Deep Learning Powered In-Session Contextual Ranking using Clickthrough Data

Xiujun Li
Department of Computer Sciences
University of Wisconsin-Madison
Madison, WI 53706, USA
lixiujun@cs.wisc.edu

Chenlei Guo, Wei Chu, Ye-Yi Wang
Microsoft Bing
One Microsoft Way
Redmond, WA 98052, USA
{chgu, wechu, yeyiwang}@microsoft.com

Jude Shavlik
Department of Computer Sciences
University of Wisconsin-Madison
Madison, WI 53706, USA
shavlik@cs.wisc.edu

Abstract

User interactions with search engines provide many cues that can be leveraged to improve the relevance of search results through personalization. The context information (history of queries, clicked documents, etc.) provides strong signals about users' search intent, which can be used to personalize the search experience and improve a web search engine. We demonstrate how to generate the semantic features from in-session contextual information with deep learning models, and incorporate these semantic features into the current ranking model to re-rank the results. We evaluate our approach using a large, real-world search log data from a major commercial web search engine, and the experimental results show our approach can significantly improve the performance of the search engine. Furthermore, we also find that the domain-specific, click-based features can effectively decrease the unsatisfied clicks for the current ranking model to improve the search experience.

1 Introduction

Search engines have become a ubiquitous part of everyday life: users issue queries, examine ordered results, click on certain links, spend some time on pages, reformulate their queries, and launch their search again. During this interaction with the search engine, users provide some valuable implicit or indirect judgment to the search engine, partially revealing cues about their search intent. In recent years, both in academia and industry, there is growing interest in investigating how users' contextual information, including users' interests, search history, clickthrough data etc., can be used to improve various search related tasks, such as query reformulation, query suggestion, shopping recommendation, ranking etc. Generally two categories of context information can be mined from these search logs: *Long-term context* refers to some long-term (across the whole search history), stable information about users, such as users' general interests; *Short-term context* is the immediate information surrounding users' current search needs in a short time span, e.g., users' search history in a session, their clicked documents, their dwelling time over the clicked documents etc. Both kind of contexts provide valuable information for personalization.

Currently, some commercial search engines take into account some coarse-grained signals derived from historical queries. For example, one existing whole-page feature in the major commercial

search engine considered here is "AveQueryOverlap," which computes the token overlap between pairs of consecutive queries in the current session. Obviously this feature is coarse, lacking some semantic information derived from historical queries and current query. Another example is whether it has some domain features from previous queries, e.g. all the queries issued by one user in the last month; and it also has some distribution information about users' click behaviors to reflect users' domain preference, but not strictly associated to their search satisfaction. Here is an example.

EXAMPLE 1: Table 1 shows two queries (2nd and 6th queries) in one session from real log data. The difference between query 2 and query 6 is query specification by adding the term of "wiki" in query 6. In query 2, the corresponding Wikipedia page was clicked as a unsatisfied click (red color in the table), and in this case it is good to demote the Wikipedia page from the first position of the 6th query since it was examined as a unsatisfied page in query 2, while, in reality, the wikia page in the second position of the 6th query is a satisfied click (blue color in the table) from users log. Thus, it is better to promote the position of wikia page in the ranking result. We will address this problem at the end of this paper.

Table 1: Two queries from the current ranker

Query 2: the dangerous days of daniel x	Query 6: the dangerous days of daniel x Altrelda wiki
http://en.wikipedia.org/wiki/The_Dangerous_Days_of_Daniel_X	http://en.wikipedia.org/wiki/The_Dangerous_Days_of_Daniel_X
http://www.amazon.com/The-Dangerous-Days-Daniel-X/dp/0316119709	http://fanon.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X
http://www.goodreads.com/book/show/2235597.The_Dangerous_Days_of_Daniel_X	http://www.amazon.com/The-Dangerous-Days-Daniel-X/dp/0316119709
http://www.daniel-x.co.uk/books/dangerous-days/	http://danielx.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X_(novel)
http://www.freebooknotes.com/summaries-analysis/the-dangerous-days-of-daniel-x/	http://danielx.wikia.com/wiki/Daniel_X
http://www.jamespatterson.com/books_danielX.php#.VCR0vOfUe1A	http://www.goodreads.com/series/49946-daniel-x
http://jamespatterson.com/books_daniel_x.php#.VCR01efUe1A	http://en.wikipedia.org/wiki/Daniel_X:_Watch_the_Skies
http://books.google.com/books/about/The_Dangerous_Days_of_Daniel_X.html?id=2UBONTvr_BEC	http://www.jamespatterson.com/books_danielX.php

In this work, we focus on the short-term context, employ latent semantic deep learning models to generate the semantic features from in-session context, and investigate how to use the short-term signals to improve ranking. We incorporate some semantic, expressive signals (e.g. topic, domain) derived from in-session context into the ranking, to re-rank the results. Here, a session can be considered as a sequence of interactions for the same information need within a short period. Automatically detecting the boundary of the sessions is beyond this paper, which has been studied previously [10]. We assume that the session boundaries are known. The specific contributions of this paper include:

- Beyond preference, we correlate the users clicks with their satisfaction and propose a set of fine-grained semantic features to explore the relations between in-session history queries, clicked URLs to the current query and URLs.
- Employ the deep learning models [2, 3] to measure the semantic similarity for the above semantic features.
- Besides semantic features, we also find that domain click-based features could effectively decrease the unsatisfied click rate to improve the search experience of users.

2 Related Work

Context information can be useful in identifying users' search needs. A large volume of past research explores different forms of context and search activities, and build numerous predictive models [1, 11-13] to improve the search personalization. Much of the previous work focuses on modeling long-term of user interests in personalized search. Teevan *et al.* [15] developed rich models from users' search and desktop document activities to improve ranking. Tan *et al.* [16] studied statistical language models to mine the contextual information from long-term search history to build the more accurate query language model. White *et al.* [12] leveraged user behavior signals from browser history and query history to re-rank the search results and predict user search interests in the future.

In particular, they built a predictive model to leverage the contextual information by representing user interests as a list of Open Directory Project (ODP) categories so that the model is capable of scaling up to million users with billions of URLs. Matthijs *et al.* [14] developed models to construct user profiles from browsing history and evaluated in an interleaving methodology. Sontag *et al.* [13] developed generative and discriminative probabilistic models from historical clicks data to infer the relevance of documents for a specific user.

Modeling the short-term interests based on search queries and clicked results have also been studied to improve search quality. The context of search activities within a short-term span (i.e., session) has been used to build richer models of interests and improve how the search system interprets the user's current query. Shen *et al.* [8] proposed context-aware ranking algorithms in which they enriched the current query by using context information, and then fitted the enriched query into statistical language models for retrieval. The basic idea is to promote the documents that are more similar to the previous queries and clicked documents within the same session. Fox *et al.* [9] developed the Bayesian model to correlate implicit measures and explicit relevance judgments for both individual users and search sessions, aiming to predict the explicit relevance judgment from implicit user actions, not specifically for ranking. Cao *et al.* [5, 6, 7] extracted context information in web search sessions by modeling search sessions as sequences of user queries and clicks. They learned sequential prediction models such as Hidden Markov Model and Conditional Random Fields from large-scale search log data. Their models were mainly designed to infer user search intents based on context information, which were applied to URL recommendation, query suggestion, and query categorization, but not ranking. Xiang *et al.* [4] formalized the relations between the current query and previous queries and clicked answers as: reformulation, specialization, generalization and general association; derived features from these principles. The work is quite similar with ours, but there are some differences: 1). They do not consider the users' satisfaction to the clicked answers. 2). They do not consider the similarity from the semantic level. 3). They do not handle the unsatisfied cases in the search queries, which is when the previous query is same as the current query.

3 Approach

In this section, we formalize the problem, and then describe our approach. We briefly summarize how to derive the fine-grained semantic features between the current query, URLs and their context information, apply the deep learning models to calculate their semantic similarity, and evaluate the effectiveness of contextual information for ranking using the real log data from a major commercial search engine.

3.1 Problem Definition

In a session, a user may interact with the search engine several times. During interactions, the user would modify his/her query based on the past search experience. Therefore for the current query Q (except for the first query in a search session), there is a query history $Q_1, Q_2, \ldots, Q_m$ associated with it. For each historical query, there are some clicked URLs with multi-fold information, i.e. dwelling time on that page. Hence, the context of current query often contains the queries asked before Q as well as the answers (URLs) to those queries that were clicked on or skipped by the user. For a query Q_t in a session at time t, we constrain the context of Q_t to only the query Q_i asked before Q_t in the same session and the answers to Q_i clicked on or skipped by the user. In practice, the real world log data is usually noisy, contains a lot of "quickback" queries (identical queries issued by the user in a short time, or consecutively), here we would recommend to ignore these quickback queries. Figure 1 illustrates the problem.

Defining

1. $Q_1, Q_2, \ldots, Q_m$ are the historical queries in one session; Q is the current query.
2. $u_{i1}, u_{i2}, \ldots, u_{in}$ are the URLs in one impression page for the query Q_i; it contains satisfied ("sat" for short) or unsatisfied ("unsat") clicks, or both. (*Impression page* is one page of search results returned by search engine, it usually contains 10 URLs; **sat** click is either a click followed by no further clicks for 30 seconds or more, or the last result click in the session; **unsat** click is when the user clicks the URL, and stays on that page less than 30 seconds; and there are also many skipped pages in one impression.)

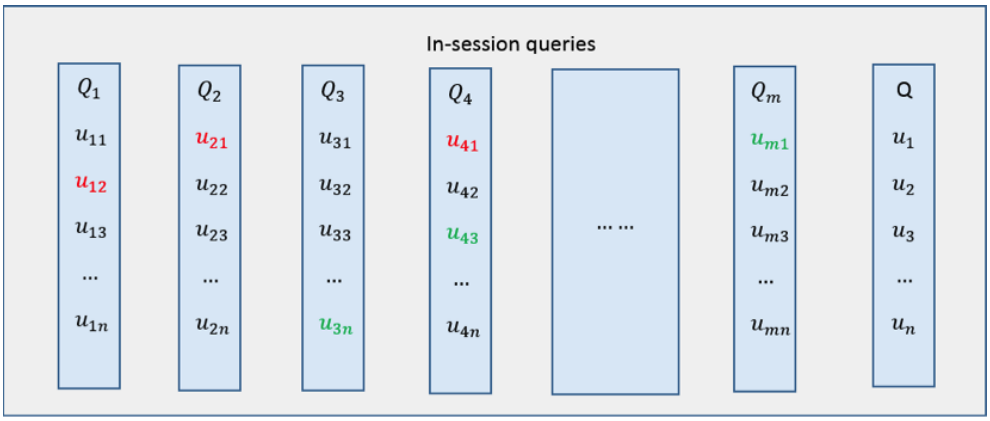

Figure 1: In-Session Queries Scenario
Red URLs are unsatisfied clicks, blue URLs are satisfied clicks.

3. $u_1, u_2, \ldots, u_n$ are the URLs in the impression page for the current query Q based on the current ranking algorithm;

Our goal is to re-rank the $u_1, u_2, \ldots, u_n$ of the current query based on the some signals (e.g. topic, domain) derived from in-session history queries and clickthrough data.

3.2 Fine-Grained Features

Previous in-session queries and clickthrough data can be viewed as the context information for the current query. We model the contextual information derived from users' in-session historical queries as a combination of a "global" component (i.e., query-level, they are not URL discriminative, including some sat/unsat clicks and URLs statistics information from in-session previous queries), and a "local" relevance component (i.e. URL-level, including the average distance of sat clicked URLs to the current URL, the least distance of a previous sat clicked URL to the current URL etc.), which is URL discriminative.

Table 2: Fine-Grained Features

Features	Level	Description	Sub Category
Semantic Features	URL-URL	The similarity between the urls of previous queries and the urls of current query	Satisfied
			Unsatisfied
	Query-URL	The similarity between the previous queries and the urls of current query	Satisfied
			Unsatisfied
	Weighted URL-URL	The weighted similarity between the urls of previous queries and the urls of current query	Satisfied
			Unsatisfied
Click-Based Features	URL Level	The domain and history click statistics on the current url of current query	History Clicks
			Domain Clicks
	Impression URL	The Sat/UnSat statistics on the in-session history queries	Clicked URLs
			Clicked Counts
			Last Click

The general approach is to implicitly represent this context information as a vector of features, and then train a ranker on these features to discover feature values indicative of relevance. Table 2 gives a different hierarchical structure about the features to represent the contextual information and users web search behaviors. Two categories are included: *Semantic features* and *Click-Based features*.

- Semantic features measure the similarity between previous queries, clicked URLs to the current query, and URLs from a semantic level. All of them are URL-level and can distinguish one URL from another.

- Click-based features encode the user interaction behaviors and content preferences in web search, e.g. satisfied click count, domain satisfied click count etc.

All these features are associated with users' satisfaction, which can give more information about user interests beyond content preferences.

3.3 Semantic Features Calculation

Most search engines resort to semantic methods in matching web documents with search queries, because a single concept can often be expressed using different words or lexicon. Here many of our features involve measuring the similarity of two queries. To accurately measure the similarity between the previous queries, clicked URLs and the current query, current URLs, we calculate three distances. (The definition of Q_t, Q and u_{it}, u_j is in Section 3.1):

1. $dis(Q_t, Q)$: the distance between query Q_t and Q.
2. $dis(u_{it}, u_j)$: the distance between i-th URL of query Q_t and j-th URL of query Q.
3. $dis(Q_t, u_j)$: the distance between query Q_t and j-th URL of query Q.

To calculate the three distances, we assess three different semantic models: XCode, DSSM [2], and CDSSM [3]. While the specific training algorithms of each model are beyond the scope of this paper, we just briefly review them.

XCode: XCode (developed internally) is trained based on massive data (Query, URL, Title, Docs etc.), to map every word, query and URL into a latent intent space of fixed size. Each URL, query and word has a unique 512-bit XCode, so we can measure the similarity in this intent space.

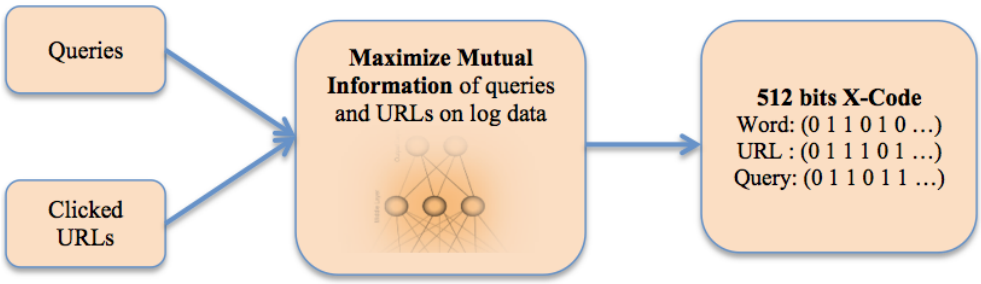

Figure 2: XCode Training

DSSM: Deep Structured Semantic Model is a latent semantic model, which provides a way to encode the contextual information at a semantic level. It uses a feed-forward neural network to map high-dimensional sparse text (i.e., query, document) into low-dimensional latent features in a semantic space. The first layer is for word hashing, which converts the term vector to a letter-trigram vector, to handle an unlimited vocabulary to enhance the scalability of the model; then the word-hashed features are projected through multiple layers of non-linear projections, to form the features in a semantic space. Ultimately, we can measure the similarity of queries and URLs in the semantic space.

CDSSM: Convolutional Deep Structure Semantic Model can capture the important word n-gram level contextual structures, while DSSM treats them as a bag of words (n-gram). Their differences are 1) It has a sliding window to capture the local contextual information for a word. 2) Then the convolutional layer projects the information within the sliding window to a local contextual feature vector. 3) The max pooling layer extracts the most salient local features to form a global contextual feature vector. We measure the similarity in this semantic space.

These three models share one basic idea - first project the text (query, URL, doc) into a semantic space, then measure the distance in that space. Table 3 summarizes the semantics of the three models.

Table 3: Model Semantics Comparison

Model	Distance	Range	Semantics
XCode	Hamming Distance	[0, 512]	0 is the most similar; 512 is the least similar
DSSM	Cosine Distance	[-1, 1]	1 is the most similar; -1 is the least similar
CDSSM	Cosine Distance	[-1, 1]	1 is the most similar; -1 is the least similar

3.4 Click-Based Features

To learn users search preference, we also employ some click-based features, and correlate users' feedback (i.e., domain clicks and URL clicks) with their satisfaction. Besides giving some distribution information about users clicks, the satisfaction information derived from the click-based features (Table 4) provides strong evidence of users' preference.

Table 4: Click-Based Features

Level	Category	Features	Semantics
URL Level	History Click	clickcount_url_total clickcount_url_sat clickcount_url_unsat	How many sat/unsat clicks for the current url in the previous queries in this session
	Domain Click	domain_click_total domain_click_sat domain_click_unsat	How many sat/unsat clicks for the domain of current url in the previous queries in the same session
Impression Level	Clicked URL	sat_click_urls unsat_click_urls	Some statistics from in-session history queries
	Clicked Count	sat_click_count unsat_click_count	

4 Experiments

The ultimate goal is to incorporate these context features derived from in-session context into the current ranker to re-rank the results, to improve users' search experience. We conducted two experiments. The first method is to manually label some data. Although manually labeled data is of good quality, it is prohibitively expensive to scale up for a real-world search engine. The second approach is by automatics derivation of training data from search logs. Here we conduct our experiments offline over a large set of real user queries and clicked log data from a search engine.

4.1 Re-ranking Experiment

We prepare the experimental data from the raw clickthrough log data, and create feature data from search sessions. We use the original rank position of a document returned by the baseline ranker as a feature, and append the new contextual features at the end. Since the new model incorporates original search engine's ranking, we call it a *reranking* model. The performance of *reranking* methods can be evaluated by whether the ranker promotes the search results of satisfied clicks to higher positions, and demotes the search results of unsatisfied clicks to lower positions.

We took 15 days of raw log data to generate the feature data, one week as training data, one day as validation data, and one week as test data. For our evaluation, we measure our performance using the mean average precision (MAP) of the re-ranked lists. This is the mean of the average precision attained for each of the queries across the results retrieved before re-ranking (for the baseline) and after re-ranking. The change in the performance from the baseline ranking can more clearly illustrate performance difference. Here are the metrics we mainly care about:

1. TimeToSuccess (sec): The dwelling time to meet the first Sat Click in the session.
2. Session Success Position: Defined by the position of the first Sat Click in the session.
3. TimeToSuccess from Quickback (sec): The dwelling time to success from a unsatisfied query in the session.

4. SatClick Rate: The Satisfied clicks in the current ranking.
5. UnSatClick Rate: The Unsatisfied clicks in the current ranking.

4.2 Results

We first experimented with different models of re-ranking the baseline results. Figure 3 reports the performance of individual features. Interestingly, DSSM trained from {Query, Title} pair is the best one among them, followed by CDSSM trained from {Query, Title}. We will interpret why CDSSM is not better than DSSM in the Discussion section.

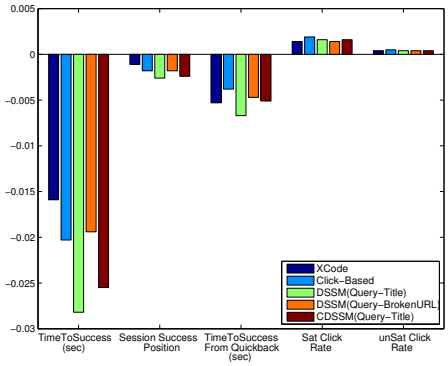

Figure 3: Ranking of individual features

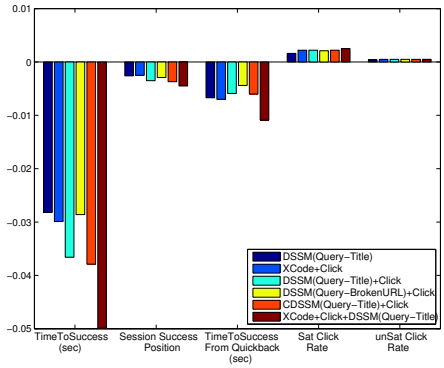

Figure 4: Ranking of combined features

Figure 4 shows the performance for the combined features. Interestingly, semantic features from CDSSM (Query-Title) model combined with click-based features is the best group. Furthermore, the combination of XCode features, Click-based features and DSSM (Query-Title) model can gain extra improvement again. We can see that semantic features and click-based features can provide different context information to the web search task.

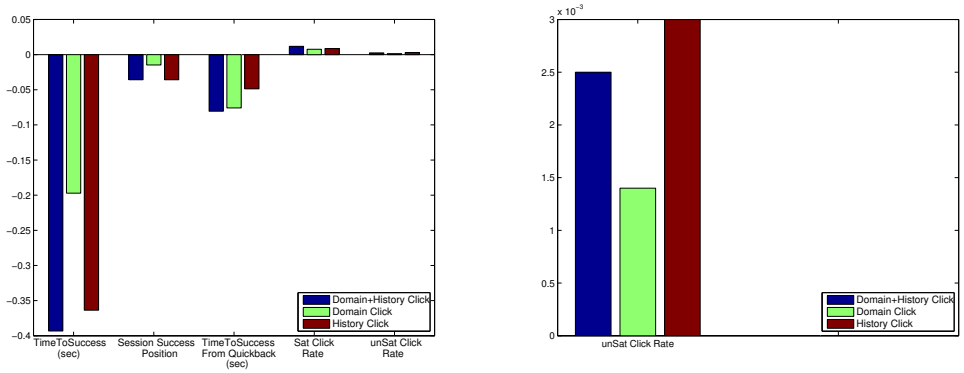

Figure 5: Ranking performance of click-based features: Domain Clicks (green) v.s. History Clicks (brown); The left is click-based features; the right is a magnified figure for unSat Click Rate.

Figure 3 shows that click-based features can also offer fairly good performance, even better than XCode features, and more importantly, it is easy to obtain and compute. Recall that the URL-level click-based features have domain click and history click features in Table 4. Besides, we conduct further analysis for click-based features to interpret where the model succeeds and fails, and find domain clicks and history clicks serve different roles in the personalized ranking task: domain clicks can decrease the unsatisfied click rate and increase the sat click rate, while history clicks can improve the TimeToSuccess and Session Success Position. This confirms with our intuition that if a URL appears in the current query, which has received a satisfied click in the previous queries, the impression will be easy to succeed. Figure 5 compares two kinds of features.

Table 5: The new ranking results, the left side is old ranking, the right side is new ranking.

Query 6: the dangerous days of daniel x Altrelda wiki	Query 6: the dangerous days of daniel x Altrelda wiki
http://en.wikipedia.org/wiki/The_Dangerous_Days_of_Daniel_X	http://fanon.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X
http://fanon.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X	http://en.wikipedia.org/wiki/The_Dangerous_Days_of_Daniel_X
http://www.amazon.com/The-Dangerous-Days-Daniel-X/dp/0316119709	http://danielx.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X_(novel)
http://danielx.wikia.com/wiki/The_Dangerous_Days_of_Daniel_X_(novel)	http://www.goodreads.com/series/49946-daniel-x
http://danielx.wikia.com/wiki/Daniel_X	http://danielx.wikia.com/wiki/Daniel_X
http://www.goodreads.com/series/49946-daniel-x	http://www.amazon.com/The-Dangerous-Days-Daniel-X/dp/0316119709
http://en.wikipedia.org/wiki/Daniel_X:_Watch_the_Skies	http://www.jamespatterson.com/books_danielX.php
http://www.jamespatterson.com/books_danielX.php	http://en.wikipedia.org/wiki/Daniel_X:_Watch_the_Skies

Let us revisit the example at the beginning of this paper. We run our ranker to produce re-ranked order as shown in Table 5. Since the Wikipedia domain has 5 unsatisfied clicks in the query 2, the new ranker demotes the Wikipedia page, and promotes the wikia page in query 6. This wikia page is sat-clicked at query 6, perfectly meeting the user's needs.

5 Discussion and Conclusion

We have studied how to use the latent semantic deep learning models to learn the semantic features in the contextual-ranking problem, and also learned how each kind of features may be used in isolation or in combination to contribute relevance gains. Through a large-scale analysis of real clicked search logs, we found that the DSSM model trained on {Query, Title} model is the best one for individual features, slightly higher than CDSSM. We expected that CDSSM would be better than DSSM, but that turned out to not be the case. Some potential reasons follow. One, in our experiment, the DSSM model was trained on a very large dataset, while CDSSM can only train on a small dataset with a up-bounded size (i.e. 1M queries for web search); the size of the training dataset might affect the prediction ability of the model. Two, due to time constraint, not all parameters (e.g. sliding window size) of the CDSSM model in the experiment were optimally tuned. Thus, the CDSSM model warrants further exploration.

Our experimental results show significant improvement for the web ranking by employing the sophisticated deep learning models to derive the semantic features. Currently all of our experiments were conducted offline over real clicked-log data and we are going to verify the gain in online experiments.

To summarize, we studied the problem of contextual ranking in personalized web search. We employed deep learning models to measure the semantic similarity between the in-session contextual information (i.e., previous queries, clicked URLs) and the current query, the current URLs. The experimental results demonstrated significant improvement from our models to the ranking of a major commercial search engine. Furthermore, we showed that domain clicks and history clicks serve different roles in personalized search, where domain clicks can effectively decrease the unsatisfied click rate.

Acknowledgments

We would thank Wayne Xiong for the help during XCode generation, and Byungki Byun, Jianfeng Gao for CDSSM Training, and Anthony Gitter for assistance preparing the manuscript. JS was supported by DARPA Grant FA8750-13-2-0039.

References

[1] Agichtein, E., Brill, E., & Dumais, S. (2006). 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*, pp. 19-26. ACM.

[2] Huang, P. S., He, X., Gao, J., Deng, L., Acero, A., & Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pp. 2333-2338. ACM.

[3] Gao, J., Pantel, P., Gamon, M., He, X., Deng, L., & Shen, Y. (2014). Modeling interestingness with deep neural networks. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

[4] Xiang, B., Jiang, D., Pei, J., Sun, X., Chen, E., & Li, H. (2010,). Context-aware ranking in web search. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pp. 451-458. ACM.

[5] Cao, H., Jiang, D., Pei, J., He, Q., Liao, Z., Chen, E., & Li, H. (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*, pp. 875-883. ACM.

[6] Cao, H., Hu, D. H., Shen, D., Jiang, D., Sun, J. T., Chen, E., & Yang, Q. (2009). Context-aware query classification. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 3-10. ACM.

[7] Cao, H., Jiang, D., Pei, J., Chen, E., & Li, H. (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*, pp. 191-200. ACM.

[8] Shen, X., Tan, B., & Zhai, C. (2005). Context-sensitive information retrieval using implicit feedback. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 43-50. ACM.

[9] Fox, S., Karnawat, K., Mydland, M., Dumais, S., & White, T. (2005). Evaluating implicit measures to improve web search. *ACM Transactions on Information Systems (TOIS)*, 23(2), 147-168.

[10] Huang, X., Peng, F., An, A., & Schuurmans, D. (2004). Dynamic web log session identification with statistical language models. *Journal of the American Society for Information Science and Technology*, 55(14), 1290-1303.

[11] Piwowarski, B., & Zaragoza, H. (2007). Predictive user click models based on click-through history. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pp. 175-182. ACM.

[12] White, R. W., Bailey, P., & Chen, L. (2009). Predicting user interests from contextual information. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp. 363-370. ACM.

[13] Sontag, D., Collins-Thompson, K., Bennett, P. N., White, R. W., Dumais, S., & Billerbeck, B. (2012). Probabilistic models for personalizing web search. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pp. 433-442. ACM.

[14] Matthijs, N., & Radlinski, F. (2011). Personalizing web search using long term browsing history. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pp. 25-34. ACM.

[15] Teevan, J., Dumais, S. T., & Horvitz, E. (2005). Personalizing search via automated analysis of interests and activities. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 449-456. ACM.

[16] Tan, B., Shen, X., & Zhai, C. (2006). Mining long-term search history to improve search accuracy. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 718-723. ACM.