Attention-based Hierarchical Neural Query Suggestion

Wanyu Chen
Science and Technology on Information Systems
Engineering Laboratory
National University of Defense Technology
Changsha, China
wanyuchen@nudt.edu.cn

Fei Cai*
Science and Technology on Information Systems
Engineering Laboratory
National University of Defense Technology
Changsha, China
caifei@nudt.edu.cn

Honghui Chen
Science and Technology on Information Systems
Engineering Laboratory
National University of Defense Technology
Changsha, China
caifei@nudt.edu.cn

Maarten de Rijke
Informatics Institute
University of Amsterdam
Amsterdam, The Netherlands
derijke@uva.nl

ABSTRACT

Query suggestions help users of a search engine to refine their queries. Previous work on query suggestion has mainly focused on incorporating directly observable features such as query co-occurrence and semantic similarity. The structure of such features is often set manually, as a result of which hidden dependencies between queries and users may be ignored. We propose an Attention-based Hierarchical Neural Query Suggestion (AHNQS) model that combines a hierarchical structure with a session-level neural network and a user-level neural network to model the short- and long-term search history of a user. An attention mechanism is used to capture user preferences. We quantify the improvements of AHNQS over state-of-the-art recurrent neural network-based query suggestion baselines on the AOL query log dataset, with improvements of up to 21.86% and 22.99% in terms of MRR@10 and Recall@10, respectively, over the state-of-the-art; improvements are especially large for short sessions.

CCS CONCEPTS

• **Information systems** → **Query suggestion**;

KEYWORDS

Neural methods for information retrieval, Query suggestion

1 INTRODUCTION

Modern search engines offer query suggestions to help users express information need effectively. Previous work on query suggestion, such as probabilistic models and learning to rank techniques, mainly relys on features indicating dependencies between queries and users, such as clicks and dwell time [2, 7]. However, the structure

*Corresponding author.

of those dependencies is usually modeled manually. As a result, hidden relationships between queries and a user's behavior may be ignored. Consequently, recurrent neural network (RNN) based approaches have been proposed to tackle these challenges. A query log can be treated as sequential data that can be modeled to predict the next input query. However, existing neural based methods only consider so-called current sessions (in which a query suggestion is being generated) as the search context for query suggestion [8].

We propose an Attention-based Hierarchical Neural Query Suggestion (AHNQS) model that applies a user attention mechanism inside a hierarchical neural structure for query suggestion. The hierarchical structure contains two parts: a session-level RNN and a user-level RNN. The session-level RNN captures queries in the current session and is used to model the user's short-term search context to predict their next query. The user-level RNN captures the past search sessions for a given user and is applied to model the user's long-term search behavior to output a user state vector representing their preferences. We use the hidden state of the session-level RNN as the input to the user-level RNN; the user state of the latter is then used to initialize the first hidden state of the next session-level RNN.

In addition, we apply an attention mechanism inside the hierarchical structure that is meant to capture a user's preference towards different queries in a session. This addition is based on the assumption that different queries in the same session may express different aspects of the user's search intent [1], e.g., queries with subsequent click behavior are more likely to represent the user's information need than those without. An attention mechanism can automatically assign different weights for hidden states of the queries in the session-level RNN. The attentive hidden states together compose the session state, which is used as input for the user-level RNN.

We compare the performance of AHNQS against a state-of-the-art query suggestion baseline and variants of RNN based query suggestion methods using the AOL query log. In terms of query suggestion ranking accuracy we establish improvements of AHNQS over the best baseline model of up to 21.86% and 22.99% in terms of MRR@10 and Recall@10, respectively.

Our contributions in this paper are: (1) We tackle the challenge of query suggestion in a novel way with neural network based method. (2) We propose AHNQS, which adopts a hierarchical structure containing a user attention mechanism to better capture the user's search intent. (3) We analyse the impact of session length on
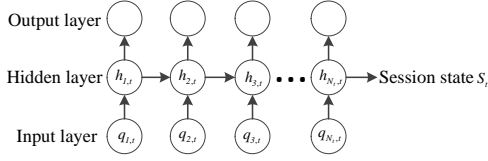
Figure 1: Structure of the NQS model.

query suggestion performance and find that AHNQS consistently yields the best performance, especially with short search contexts.

2 APPROACH

2.1 Session-level RNNs for query suggestion

As in [10], session-level RNNs are our starting point. Here in the neural based query suggestion model (NQS), queries in the current session are taken as sequential input and used to output the probability of being the next query for the query suggestion candidates.

Formally, we assume that a query session $Session_t$ contains N_t queries, denoted as $Session_t = (q_{1,t}, q_{2,t}, q_{3,t}, \ldots, q_{N_t,t})$. As shown in Fig. 1, for generating the input vector of the network, we use a 1-of-N encoding of q_i, i.e., the vector length equals the number of unique queries V and only the coordinate corresponding to the i-th query is one, the others are zero. We choose to use the Gated Recurrent Unit (GRU) [3] as our non-linear transformation. The hidden state h_n can be calculated by using the previous hidden state h_{n-1} and the candidate update state $\widehat{h}_n$:

$$h_n = (1 - u_n)h_{n-1} + u_n\widehat{h}_n, \tag{1}$$

where the update gate u_n can be generated by:

$$u_n = \sigma(I_u q_{n,t} + H_u h_{n-1}). \tag{2}$$

The candidate update state $\widehat{h}_n$ is calculated by:

$$\widehat{h}_n = \tanh(I q_{n,t} + H(r_n \cdot h_{n-1})), \tag{3}$$

where the reset gate r_n is:

$$r_n = \sigma(I_r q_{n,t} + H_r h_{n-1}) \tag{4}$$

where $I, I_u, I_r \in \mathbb{R}^{d_h \times V}$, $H, H_u, H_r \in \mathbb{R}^{d_h \times d_h}$, and d_h is the number of dimension of the hidden state. The H matrices are used to keep or forget the information in h_{n-1}.

We use $RNN_{session}$ and RNN_{user} to denote the GRU function. The final hidden state of a session-level RNN is used to indicate the session state, $S_t = h_{N_t,t}$. The output of the session-level RNN are the scores of query suggestion candidates being predicted as the next query: $s_{n,t} = g(h_{n,t})$, where $g(\cdot)$ is the active function of the output layer, which can be a softmax or tanh depending on the loss function of the neural network. We generate the query suggestion list in the test set according to the scores of query candidates.

We choose a pairwise loss function that forces positive query suggestion samples to be ranked higher than the negative ones. Actually, there are several pairwise ranking loss functions, including cross-entropy and TOP1 [5]. In the field of recommender systems, TOP1 has proved to outperform others, so we set:

$$Loss = \frac{1}{N_S} \cdot \sum_{j=1}^{N_S} \sigma(s_{j,t} - s_{i,t}) + \sigma(s_{j,t}^2), \tag{5}$$

where $s_{j,t}$ and $s_{i,t}$ denote the score of a negative query candidate and a ground truth query, respectively.

2.2 Hierarchical user-session RNN for query suggestion

Clearly, the NQS model only models the short-term search context. Here, we model the long-term search behavior of a given user with a user-level RNN, producing the hierarchical NQS model (HNQS).
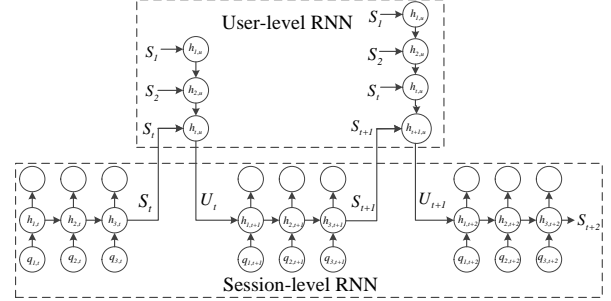

Figure 2: Structure of the HNQS model.

We assume that a user u has N_u query sessions, $User_u = (Session_{1,u}, Session_{2,u}, \ldots, Session_{N_u,u})$. In a user-level RNN, the input is the session state S_t and the hidden state can be calculated as:

$$h_{n,u} = RNN_{user}(h_{n-1,u}, S_{n,u}), \tag{6}$$

where $S_{n,u}$ is the session state of the n-th query session of user u, which is equal to the last hidden state of the session-level RNN.

As shown in Fig. 2, we use the final hidden state of the user-level RNN to denote the user state, $U_t = h_{t,u}$, that contains the information about the search behavior from a user's past sessions and thus can be applied in the session-level RNN. In HNQS, the session-level RNN is initialized with a user state as follows: $h_{0,t+1} = \tanh(W \cdot U_t + b_0)$. We choose to use only the initialization strategy for session-level RNN with user state U_t, instead of transporting the user information from U_t throughout the whole session-level RNN including initialization, updating and output. The GRU unit has both long and short term memory [6] and can automatically transport the user state information within the network, which leads to a better performance when combined with our initialization strategy.

2.3 Attention-based hierarchical RNN for query suggestion

We assume that submitted queries that trigger subsequent click behavior have a better expression of the user's search intent. We hypothesize that queries in a session should have different weights to reflect the user's information need and employ an attention mechanism on top of the HNQS model to capture the user's preference for different queries in a session and then aggregate the representations of informative queries.

Fig. 3 shows how we update the user-level RNN in AHNQS as follows: $h_{t,u} = RNN_{user}(h_{t-1,u}, C_t)$, where $C_t = \sum_{j=1}^{M} \alpha_{jt} h_{j,t}$ is the attentive representation of the session state, a weighted sum of the hidden states $h_{j,t}$ from the session-level RNN, where α_{jt} is the normalized attention score for the j-th query in session $Session_t$, which is interpreted as the contribution of the query to the preference of the user:

$$\alpha_{jt} = \frac{\exp(e_{jt})}{\sum_{k=1}^{M} \exp(e_{kt})}, \tag{7}$$

where $e_{jt} = h_{t-1,u}^{\mathrm{T}} W_a h_{jt}$ is the initial attention score computed with user state $h_{t-1,u}$ and hidden state h_{jt} in a session-level RNN,
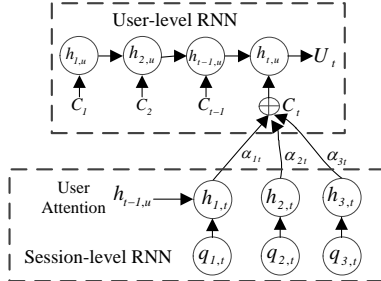
Figure 3: Structure of the AHNQS model.

Table 1: Dataset statistics.

Variable	Training	Test
# Queries	1,545,543	576,817
# Unique queries	61,641	33,519
# Sessions	166,414	67,716
# Users	23,308	19,255
Average # queries per session	9.28	8.52
Average # sessions per user	7.14	3.52

where the parameters $\mathbf{W}_a$ can be jointly trained with all the other components of AHNQS as the attention mechanism allows the gradient of the loss function to be backpropagated.

Thus, AHNQS combines a hierarchical user-session RNN and an attention mechanism for query suggestion; the hierarchical structure models the user's short and long-term search behavior, while the attention mechanism captures the user's query preference.

3 EXPERIMENTS

Research questions. (RQ1) Do the hierarchical structure and attention mechanism incorporated in HNQS and AHNQS help to improve the performance of the neural query suggestion model and beat the state-of-the-art method? **(RQ2)** What is the impact on query suggestion performance of session length, i.e., short vs. medium length vs. long sessions?

Model summary. As AHNQS is based on a neural network to capture the dependencies between queries and users, we compare it with the state-of-the-art neural models for query suggestion. As an aside, Sordoni et al. [10] incorporate a neural query suggestion model as a feature into a learning to rank approach and thus belongs to the feature engineering approaches we do not compare with. We consider the following baselines for comparison: (1) ADJ: original co-occurrence based query suggestion method [7]; (2) NQS: a simple session-based RNN method for query suggestion [5], §2.1. In addition, we consider: (3) HNQS: a hierarchical structure with user-session level RNN for query suggestion, §2.2; (4) AHNQS: an attention-based hierarchical RNN model for query suggestion, §2.3.

Datasets and parameters. We use the AOL query log [9] and preprocess the dataset following [4]. Queries are separated into sessions by 30 minutes of inactivity. We remove queries with less than 20 occurrences and keep sessions with length larger than 5 as well as users with at least 5 sessions to provide sufficient user-session information. The training set consists all but the last 30 days in the search history; the test set consists of the last 30 days in the log after filtering out queries that do not exist in the training set. Table 1 details the statistics of the dataset used.

Table 2: Parameters used for each model.

Model	Batch	Dropout	Learning rate	Momentum
NQS	50	0.5	0.01	0.0
HNQS	50	0.1	0.1	0.0
AHNQS	50	0.1	0.1	0.0

We use GRUs as the RNN units and optimize the neural models using TOP1 loss function and AdaGrad with momentum for 20 epochs. The number of hidden units is set to 100 in all cases and we use dropout regularization. We optimize the hyperparameters by running 100 experiments at randomly selected points of the parameter space. Optimization is done on a validation set, which is partitioned from the training set with the same procedure as the test set. We summarize the best performing parameters in Table 2.

Training and evaluation. As the lengths of sessions are different and our goal is to capture how a session evolves over time, traditional methods to form batches in natural language processing are not suitable for our query suggestion task. We thus use parallel mini-batches with the identifications of users and sessions following [5]. We evaluate the models by providing queries in a session one by one and measure the ranking performance of query suggestions with MRR and Recall on the test set.

4 RESULTS AND DISCUSSION

Overall performance. To answer **RQ1**, we examine the query suggestion performance of the baselines as well as the HNQS and AHNQS models. Table 3 shows the results. ADJ outperforms NQS,

Table 3: Performance of query suggestion models. The results by the best baseline and the best performer in each column are underlined and in boldface, respectively. Statistical significance of pairwise differences of HNQS and AHNQS vs. the best baseline) is determined by a t-test ($\blacktriangle$/$\blacktriangledown$ for α = .01).

Model	Recall@10	MRR@10
ADJ	.7072	.6922
NQS	.6444	.6148
HNQS	.8138$^{\blacktriangle}$	.7874$^{\blacktriangle}$
AHNQS	**.8618**$^{\blacktriangle}$	**.8514**$^{\blacktriangle}$

with 9.74% and 12.58% improvements in terms of Recall@10 and MRR@10, respectively. This may be because that the NQS model (without knowing about individual users) fails to capture information from the past search history. HNQS shows significant improvements over ADJ, with Recall@10 improved 15.07% and MRR@10 improved 13.75%. The hierarchical structure effectively incorporates a given user's previous search behavior and thus improves accuracy. The best performance is obtained by AHNQS, which outperforms ADJ by 21.86% (Recall@10) and 22.99% (MRR@10), and HNQS by 5.9% (Recall@10) and 8.13% (MRR@10). The latter difference indicates that attention can strengthen the model's ability to rank query suggestion candidates effectively.

To determine the impact of the hierarchical structure and attention mechanism, we consider a sample session and user, and compare the hidden states of an RNN in NQS and HNQS, respectively (Fig. 4a and 4b), as well as the hidden states of an RNN in HNQS and AHNQS, respectively (Fig. 4c and 4d). The lighter the

(a) session-level RNN in NQS.   (b) session-level RNN in HNQS.

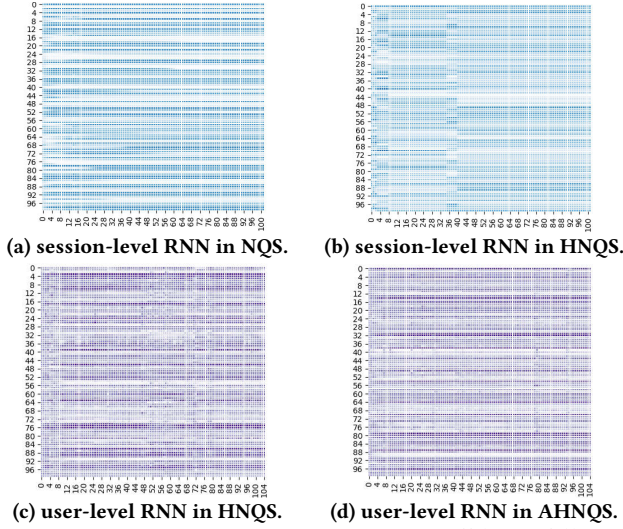(c) user-level RNN in HNQS.   (d) user-level RNN in AHNQS.

Figure 4: Visualizing hierarchical structure ((a) and (b)) and attention mechanism ((c) and (d)). The lighter the area in the plot, the more important the information is.
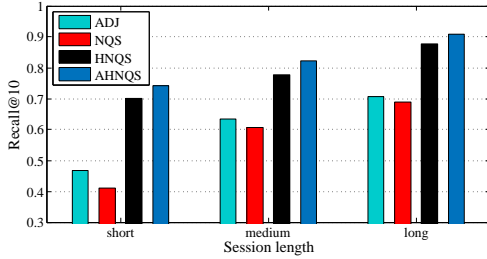

Figure 5: Performance with different session lengths.

area in the plot, the more important the information is. In Fig. 4a and 4b, the session contains 102 queries (x-axis); the number of hidden units is 100 (y-axis). Compared with Fig. 4a, we can see that the hierarchical structure modifies the user's search intent especially at the first positions in a session in Fig. 4b. There is a fluctuation around the 36th to 40th queries in Fig. 4b, which may be due to the fact that we use a GRU unit inside the session-level RNN to transport the user state information within the network.

Turning to the attention mechanism (Fig. 4c and 4d), we select a user with 105 sessions (x-axis) and the number of hidden units in the user-level RNN is set to 100 (y-axis). Compared with Fig. 4c, the user's preference towards different information is more equally distributed inside the user-level RNN in Fig. 4d. Moreover, going from left to right there are fewer abrupt shifts form high interest (light) to low interest (dark) areas, or vice versa, in Fig. 4d than in Fig. 4c: the attention mechanism can help to describe a user's long-term search preferences towards different topics.

Impact of current session length. For **RQ2**, we expect the current session length to have an impact on the performance of query suggestion models. We report separate results for short (2 queries), medium (3 or 4 queries) and long current sessions (at least 5 queries) on the test set in Fig. 5.

As the session length increases, the performance in terms of Recall@10 of all query suggestion models improves and AHNQS always achieves the highest scores. As for baselines, ADJ outperforms NQS; the margin between them shrinks as the session length increases. For short current sessions, HNQS and AHNQS yield larger

improvements than for medium and long current sessions. This is due to the fact that when predicting a user's search intent at the first positions of a session, the hierarchical structure within RNN models can provide effective information from a user's past search history and thus can improve the accuracy for query suggestion (compared to ADJ and NQS).

For MRR a similar picture emerges (not shown). AHNQS shows bigger improvements over HNQS, of 11.23%, 7.13% and 8.74% in terms of MRR@10, for short, medium and long sessions, respectively, vs. improvements of 5.88%, 5.97% and 3.61% for Recall@10. This confirms our intuition about attention mechanisms, i.e., that they help to improve precision.

5 CONCLUSIONS AND FUTURE WORK

We have proposed an attention-based hierarchical neural query suggestion model (AHNQS) that combines a hierarchical user-session RNN with an attention mechanism. The hierarchical structure can model both the user's short-term and long-term search behavior effectively, while the attention mechanism captures a user's preference towards certain queries over others. The experimental results also confirm the effectiveness of the proposed model for query suggestion across sessions with various lengths. As to future work, we plan to evaluate our model on other datasets so as to verify its robustness. We also want to investigate the performance of AHNQS when combining semantic similarity with the hierarchical structure, i.e., different encoding methods for input queries.

REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *ICLR'14*.
[2] Wanyu Chen, Fei Cai, Honghui Chen, and Maarten de Rijke. 2017. Personalized query suggestion diversification. In *SIGIR'17*. ACM, 817–820.
[3] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, et al. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP'14*. ACL, 1724–1734.
[4] Jiafeng Guo, Xueqi Cheng, Gu Xu, and Xiaofei Zhu. 2011. Intent-aware query similarity. In *CIKM'11*. ACM, 259–268.
[5] Balazs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, et al. 2016. Session-based recommendations with recurrent neural networks. In *ICLR'16*.
[6] Sepp Hochreiter and Juergen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9, 8 (Nov 1997), 1735–1780.
[7] Chien-Kang Huang, Lee-Feng Chien, and Yen-Jen Oyang. 2003. Relevant term suggestion in interactive web search based on contextual information in query session logs. *J. Am. Soc. Inf. Sci. Technol.* 54, 7 (May 2003), 638–649.
[8] Kezban Dilek Onal et al. 2018. Neural information retrieval: At the end of the early years. *Information Retrieval Journal* (2018). To appear.
[9] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. 2006. A picture of search. In *InfoScale'06*. ACM, Article No. 1.
[10] Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, et al. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *CIKM '15*. ACM, 553–562.