Improving Search Snippets in Context-aware Web Search Scenarios

Jia Chen, Jiaxin Mao, Yiqun Liu*, Min Zhang, Shaoping Ma

Department of Computer Science and Technology, Institute for Artificial Intelligence,
Beijing National Research Center for Information Science and Technology,
Tsinghua University, Beijing 100084, China

chenjia0831@gmail.com, maojiaxin@gmail.com,
{yiqunliu,z-m,msp}@tsinghua.edu.cn

Abstract. As an essential part in web search, search snippets usually provide result previews for users to either gather useful information or make click-through decisions. In complex search scenarios, users may need to submit multiple queries to search systems until their information needs are satisfied. As user intents tend to be ambiguous, incorporating contextual information for user modeling has been proved effective in many session-level tasks. Therefore, the generation of search snippets may also benefit from the integration of context information. However, to our best knowledge, most existing snippet generation methods ignore user interaction and focus merely on the query content. Whether it is useful of exploiting session contexts to improve search snippets still remains inscrutable. To this end, we propose a snippet generation model which considers session contexts. The proposed method utilizes the query sequence as well as users' interaction behaviors within a session to model users' session-level information needs. We also adopt practical log-based search data to evaluate the performance of the proposed method. Experiment results based on both expert annotation and user preference test show the effectiveness of considering contextual information in search snippet generation.

Keywords: Search snippets · User Intent · Web Search.

1 Introduction

Snippets are an essential part in **S**earch **E**ngine **R**esult **P**ages (SERPs). A search engine analyzes the content or metadata of a webpage and then extracts a small clip, i.e., a snippet, to help search users preview the landing page. Before clicking on a certain result, the user can only learn about the result content by its title or snippet. However, result titles are generally short and simple, usually directly

*Corresponding author

This work is supported by Natural Science Foundation of China (Grant No. 61622208, 61532011, 61672311) and National Key Basic Research Program (2015CB358700).

citing the titles of corresponding webpages. Moreover, some of them contain exaggerated or mendacious information, a.k.a. the *clickbaits* [17], which may lure users into clicking. In this scenario, search snippets are more credible for users to gain useful information and could be therefore essential for improving users' satisfaction.

Snippets may play a more important role in complex search scenarios. e.g., exploratory search. A user may not have a clear information need at the beginning. She needs to submit a few queries and interacts with some results to learn about the search task. In this scenario, she can obtain relevant information in the snippets and acquire effective query terms from them to reformulate her next query [9]. Sometimes query content may not fully reflect users' information needs, thus contextual information such as query history or clickthrough data should be exploited. However, query-biased snippets [6,7] are dominating in most commercial search engines nowadays. They are usually generated in the manner of selecting several relevant sentences in the result document. Few of them take the session contexts into consideration. Some previous studies propose to use pseudo or implicit relevance feedback to optimize the snippet generation [7,8]. These methods utilize the text statistics in the whole corpus rather than the contextual information within a session. Therefore, they may not successfully capture user intent in the current search tasks.

To address this issue, we propose a model to generate context-aware search snippets. Previous studies have shown great advantages of considering search contexts for session-level IR tasks. Inspired by [1], we adopt the *feedback memory mechanism* to encode session contexts. The mechanism represents users' session-level information needs based on the positive and negative feedbacks derived from their interactions. Based on the session-level representation, the model can rate each sentence in a candidate document with a relevance score. Then by considering different traits of good snippets, we design multiple methods to integrate the context-aware relevance scores into snippet generation and further evaluate their performances, respectively.

To summarize, the main contributions of our work are as follows:

- We present a novel extractive method which considers session contexts for context-aware search snippet generation.
- Experiment results based on both intrinsic and extrinsic evaluation validate mutually and indicate the effectiveness of the proposed model.

2 Related Work

2.1 Snippet Generation

Without loss of generality, snippet generation can be broadly categorized into two classes: *extractive summarization* and *abstractive summarization*. Extractive methods usually include analyzing the result page, selecting appropriate text content or media, and then forming a concise snippet to display on the SERP. Besides extracting sentences from a webpage, *paraphrasing* its content, a.k.a. abstractive summarization, is also available. Mainstream commercial search engines

usually adopt extractive methods due to the high cost of generating high-quality sentences for billions of result pages. Therefore, we do not consider abstractive methods for search snippet generation in this paper.

Numerous work have focused on extractive snippets generation. For example, Wang et al. [5] propose a *query-biased* webpage summarization method which introduces a scheme to rank candidate sentences in the result. Some other studies consider using user feedbacks to optimize snippet generation [7], e.g., Ko et al. [7] apply pseudo-relevance feedbacks and text summarization techniques to extract salient sentences for generating high-quality snippets. However, they utilize query contents as pseudo feedbacks and ignore user interactions. To better model user search intent, Ageev et al. [22] incorporate user mouse movement signals to learn *behavior-biased* snippets and open a new direction for improving search result presentation.

2.2 Snippet Evaluation

So far no standard metrics have been designed for search snippet evaluation. Therefore, NLG-summarization evaluation approaches are widely adopted. On one hand, some studies conduct snippet evaluation with *intrinsic* methods, that is, evaluating generated summaries by manually comparing them with human-created ones [12]. As factitious comparison is time-consuming, many evaluation agencies such as the **D**ocument **U**nderstanding **C**onference (DUC) adopt ROUGE [4] to access the quality of generated texts. The main idea of ROUGE is to compare the basic syntactic units (e.g. n-grams, word sequences, ordered pairs) with standard summaries generated by human experts and count the number of the overlapped ones. In our experiments, we also use ROUGE to evaluate the quality of the generated snippets.

Although *intrinsic* methods are convenient for evaluation, they lack user perception. Therefore, *extrinsic* evaluation with users involved is necessary. The core idea of extrinsic summarization evaluation is to determine the effect of summaries based on human perception tasks. For instance, one can examine the usefulness or the satisfaction of a snippet/summary with respect to her information needs or goals [13]. From this perspective, we conduct a user preference test to collect their preference on the paired snippets.

3 Extractive Context-aware Snippet Generation

In this section, we propose a context-aware snippet generation model based on the *feedback memory mechanism*. We start by elaborating how it encodes user information needs and preferences within a search session, and then introduce how to generate context-aware snippets based on session-level representations.

3.1 User Preference Modeling

To model session-level user intents, we adopt the feedback memory mechanism to represent the session context. As shown in Figure 1, users' interactions are

taken as inputs and will be converted into distributed representations. We further use encoding layers and attention mechanisms to embed user positive/negative feedbacks. Finally, we feed the feedback embeddings as well as the query representations into a recurrent neural network to represent the whole search session.

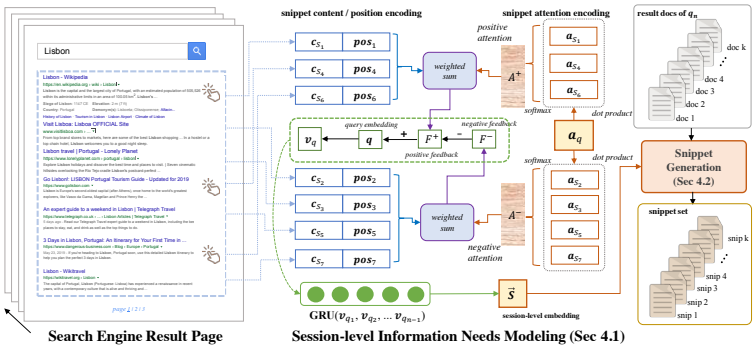


Fig. 1: Main framework of our model. Here shows an example of using clicked (1, 4, 6) and skipped (2, 3, 5, 7) snippets as the positive and negative feedbacks.

Positive/negative feedback snippets Here we collect user behaviors on each result snippet as the feedback signals. We use clicks as positive feedbacks and skips as negative ones. According to the cascade assumption [14], a user scans the result list from top to bottom and examines the result right after the last clicked one to determine whether to end this query session. Therefore, we consider all the unclicked result which rank higher or just one position lower than the last clicked result as skipped results. For a given query q and its result documents $\mathcal{D} = \{d_1, ...d_i, ...d_n\}$, the positive/negative feedback snippets $\mathcal{S}^+/\mathcal{S}^-$ can be represented as $\mathcal{S}^+ = \{\mathcal{S}_p | p \in \mathcal{C}\}, \mathcal{S}^- = \{\mathcal{S}_p | p \leq max(\mathcal{C}) + 1, p \notin \mathcal{C}\}$, where $\mathcal{S}_p$ denotes the snippet content of d_p, and $\mathcal{C}$ is the set that contains the positions of all the clicked documents. The whole framework consists of a snippet content encoding layer, a position encoding layer, and an attention layer to represent the positive/negative feedback memory $\mathcal{F}^+/\mathcal{F}^-$. These three components will be introduced in the next, respectively.

Snippet content encoding & position encoding A snippet is regarded as a sequence of words it contains $\mathcal{S}_p = \{x_1^p, ...x_j^p ..., x_{|\mathcal{S}_p|}^p\}$. We apply the standard **G**ated **R**ecurrent **U**nits (GRUs) [15] to encode a snippet into a fixed size vector:

$$\mathbf{w}_j^p = \mathbf{E}_c^T \cdot \mathbf{x}_j^p, \tag{1}$$

$$\mathbf{c}_{\mathcal{S}_p} = \mathbb{GRU}_c(\mathbf{w}_1^p, ..., \mathbf{w}_j^p ..., \mathbf{w}_{|\mathcal{S}_p|}^p), \tag{2}$$

here $\mathbf{E}_c \in \mathbb{R}^{V \times l}$ is the word embedding matrix, $\mathbf{x}_j^p$ and $\mathbf{w}_j^p \in \mathbb{R}^V$ represent the one-hot vector and the distributed vector (embedding) for the j-th word of the snippet at position p, respectively. V is the vocabulary size and l is the

dimension of word embeddings. $\mathbf{c}_{\mathcal{S}_p}$ is the output of the last hidden layer state in GRU, which represents the content of a snippet.

The position of a snippet also influences users' perception of its relevance to a certain extent, hence we encode the ranking position of snippets by $\mathbf{pos}_p = \mathbf{E}_{pos}^T \cdot \mathbf{p}$, where $\mathbf{E}_{pos}$ is the learnable embedding matrix for all ranking positions and $\mathbf{p}$ denotes the one-hot presentation for position p. In this paper, we only consider the top 15 results for a query.

Attention mechanism We combine the text embedding and position embedding of a snippet with an attention mechanism and use the weighted combinations as the positive/negative feedback memory $\mathcal{F}^+/\mathcal{F}^-$. For a specific query, we learn the positive attention $\mathcal{A}^+$ and the negative attention $\mathcal{A}^-$ separately. For a query $q = \{x_1^q, ...x_j^q, ..., x_{|q|}^q\}$ and a snippet $\mathcal{S}_p = \{x_1^p, ...x_j^p ..., x_{|\mathcal{S}_p|}^p\}$, we also embed the words first and then use GRUs to encode them into two fixed-length vectors $\mathbf{a}_q$ and $\mathbf{a}_{\mathcal{S}_p}$ respectively:

$$\mathbf{w}_j^q = \mathbf{E}_q^T \cdot \mathbf{x}_j^q, \tag{3}$$

$$\mathbf{w}_j^p = \mathbf{E}_{a_\mathcal{S}}^T \cdot \mathbf{x}_j^p, \tag{4}$$

$$\mathbf{a}_q = \mathbb{GRU}_q(\mathbf{w}_1^q, ..., \mathbf{w}_j^q ..., \mathbf{w}_{|q|}^q), \tag{5}$$

$$\mathbf{a}_{\mathcal{S}_p} = \mathbb{GRU}_{a_\mathcal{S}}(\mathbf{w}_1^p, ..., \mathbf{w}_j^p ..., \mathbf{w}_{|\mathcal{S}_p|}^p). \tag{6}$$

The attention distribution of a query q on a snippet is given by the dot product of these two vectors. We can represent the positive or negative attention $\mathcal{A}^+/\mathcal{A}^-$ with *softmax* function over positive or feedback snippets $\mathcal{S}^+/\mathcal{S}^-$: $\mathcal{A}_\mathcal{S}^{+/-} = softmax(\mathbf{a}_q^T \mathbf{a}_{\mathcal{S}_p} | \mathcal{S}_p \in \mathcal{S}^{+/-})$.

Feedback memory mechanism We then concatenate the snippet content embeddings and position embeddings together and feed them into a dense layer. The output vectors of this layer will be weighted by the attention distribution to calculate the feedback memory:

$$\mathcal{F}^{+/-} = \sum_{\mathcal{S}_p \in \mathcal{S}^{+/-}} \mathcal{A}_\mathcal{S}^{+/-}(\mathbf{W}[\mathbf{c}_{\mathcal{S}_p} \oplus \mathbf{pos}_p] + \mathbf{b}_F), \tag{7}$$

where $\mathcal{F}^+$ and $\mathcal{F}^-$ represent users' positive preferences and negative preferences within the session, respectively. $\mathbf{W}$ and $\mathbf{b}_F$ are the weight matrix and the bias of the dense layer. "$\oplus$" denotes the vector concatenation.

Session-level Embedding As the output $\mathcal{F}^+/\mathcal{F}^-$ capture user preference on a query, we further utilize it to present the session-level contexts. We first combine the query embeddings with feedback memories and then feed the combined vectors into a session-level GRU:

$$\mathbf{v}_{q_k} = \mathbf{q}_k + \mathcal{F}^+ - \mathcal{F}^-, \tag{8}$$

$$\boldsymbol{S} = \mathbb{GRU}_s(\mathbf{v}_{q_1}, ...\mathbf{v}_{q_k}, ...\mathbf{v}_{q_K}). \tag{9}$$

Training Objective Function The entire architecture is trained end-to-end [1]. Here we select the log-likelihood of decoding the last query $q_s = \{x_1, ...x_j, ..., x_{|q_s|}\}$

[1] For all datasets, we train the network for 200 epochs because the training losses have converged stably within 200 epochs.

within the session from the session representation $\boldsymbol{S}$ as the objective function. For lack of golden context-aware snippets for each result, the last query is used as a surrogate training target because it implies a user's current search intention. This training function can be formulated as follows:

$$\log \mathcal{L}(M) = \sum_{x_j \in q_s} \log P_M(x_j | x_1 : x_{j-1}, \boldsymbol{S}). \tag{10}$$

3.2 Snippet Generation

Candidate Scoring After training, we use Equation 11 to assign a relevance score for each candidate sentence. For a sentence $s = \{x_1, ...x_j, ..., x_{|s|}\}$, the score is represented as the probability of decoding a sentence from a given session-level representation $\boldsymbol{S}$:

$$score(s) = \prod_{j, x_j \in s} P(x_j | x_1 : x_{j-1}, \boldsymbol{S}). \tag{11}$$

Then we can design vaious snippet generation approaches based on these scores.
Simple Sorting (SS) A simple method is to directly rank the sentences by their relevance scores in the descending order and select those with the highest relevance scores. The process of Simple Sorting can be formulated as:

$$Snippet_{ss} = \arg\max_{\Omega'} \sum_{s \in \Omega'} S_{ss}(s), \ s.t. \sum_{s \in \Omega'} |s| \leq L, s \in \Omega, \tag{12}$$

$$S_{ss}(s) = score(s) + \sum_{i=1, x_i \in q_s}^{n} N(x_i) \cdot TFIDF(x_i), \tag{13}$$

where $S_{ss}(s)$ denotes the score of a sentence s, x_i is the i-th word in the query q_s, $N(x)$ represents the appearance number of a query word x in the sentence s, Ω is the set of candidate sentences, Ω' represents the selected sentences, and L is the word limit for a snippet. According to previous work [3], the length of a snippet has a great impact on the snippet quality. So we strictly unify the lengths of all the generated snippets.
N-tuple Sorting (NS) Snippets should be understandable and readable. So far it is unknown whether a single sentence could convey clear and integrated ideas to readers. Therefore, we cut each document into n-sentence subsequences and then rank them according to their summed scores in Equation 13. Due to the word limit, we only consider the case of $n = 2, 3$ here.
Diversified Sorting (DS) To avoid redundancy, we try to increase the diversity of snippets by the **M**aximal **M**arginal **R**elevance (MMR) [16] of candidate sentences. We first select the sentence with the highest Simple Sorting score and then sequentially choose sentences with the highest marginal relevance score S_{ds} in the remaining candidate set until exceeding the word limit. The marginal relevance score S_{ds} is defined as follows:

$$S_{ds}(s_i^t) = \beta \cdot S_{ss}(s_i^t) - (1 - \beta) \cdot \{ \max_{j < i, i \leq |C^t|} sim(\mathbf{s}_i^t, \mathbf{s}_j^t) \}, \tag{14}$$

Here s_i^t denotes the i-th sentence in the candidate set when selecting the t-th sentence in snippet and β is a hyper-parameter ($\beta \in [0.1, 0.5]$). $sim(\cdot)$ is the

Table 1: Statistics for each dataset.

Specific\Dataset	SD_1	SD_2	SQE	UPT
# *sessions*	115	246	200	99
# *queries*	486	795	727	358
# *top15 web pages*	18,244	51,042	–	–
# *pages for annotation*	–	–	632	880

⋆ Note that SQE is short for the dataset used in **S**nippet **Q**uality **E**valuation (§4.2) and UPT is short for the dataset used in **U**ser **P**reference **T**est (§4.3)

cosine similarity of two vectors, and $\mathbf{s}_i^t$ is the distributed representation of s_i^t, which is generated by max-pooled GloVe [2] vectors of each word in s_i^t.

4 Experiments and Evaluation

In this section, we introduce the experimental setups as well as the results of the two evaluation methods: snippet quality evaluation and user preference test.

4.1 Data Collection and Processing

We extract search sessions from two real-world search logs recorded in 20170903 and 20180401 by *Sogou.com* (a major Chinese commercial search engine). The log data contains user ID, query word, search results, and click data. We adopt a standard 30-minute gap to split queries submitted by the same user into sessions. Sessions that are too short or too long ($\leq$2 or $\geq$7 queries) are filtered because long sessions contain more noises but only account for little proportions while short sessions contain limited context information. We adopt `jieba` [2] for Chinese word segmentation. The webpage corpus is then fed into GloVe [2] to obtain word embeddings. Finally, the two log collections are organized into two datasets (SD_1 and SD_2) as shown in the first and second column of Table 1.

4.2 Snippet Quality Evaluation

We first use ROUGE [4] metrics to intrinsically measure the snippet quality. Then we collect reference snippets by user annotation and compare the generated snippets against the reference snippets for extrinsic evaluation.

Annotation process We recruit 25 participants aged 19-24 to annotate the reference snippets. All of the participants are familiar with the basic operations of web search. Annotation data are randomly extracted from the two datasets (80 from SD_1 and 120 from SD_2). Basic statistics of this dataset (**SQE**) is presented in the third column of Table 1. Each participant needs to complete 40 tasks. In each task, the previous queries and result hyperlinks in the last query are presented to annotators. They need to read the result document and select the sentences that are most suitable for being included in the snippet with

[2] https://github.com/yanyiwu/cppjieba

regard to user intent. The selected sentences should be pasted into an input box with a length limit of 110-165 Chinese characters (i.e., the average length of baseline snippets). All webpages receive five annotations and each of them will be regarded as a ground truth. On average, each participant spends 200 minutes to annotate 132 documents with a reward of about $8/h.

Baseline methods Few studies have focused on the context-aware summarization, so we take some typical summarization algorithms as baselines. We choose seven automatic summarizers implemented in the open-source tool SUMY [3], listed as *Luhn* [18], *Latent Semantic Analysis (LSA)* [19], *LexRank* [10], *TextRank* [11], *SumBasic* [20], *KL-Sum* [21] and *Reduction*. Another pseudo relevance feedback based method [7] is taken as the typical query-biased system for comparison (denoted by *PRF*). In addition, the snippets crawled from the *Sogou* search engine are also used as a baseline (denoted by *SE*). Note that although abstractive methods have developed a lot with the emergence of sophisticated deep learning models, they are not adaptive for commercial Web search engines, so we only consider the state-of-the-art extractive methods in this paper.

Overall evaluation results We denote our approaches as FMN_{SS}, FMN_{NS} and FMN_{DS}, respectively. The results are presented in Table 2.

Table 2: Intrinsic evaluation results of each model. All values are $F_{1.2}$-scores, where $*$ and † indicate a statistically significant improvement over the strongest baseline *SE* at $p < 0.05/0.01$ level, respectively.

Sys.\Met.	RG-1	RG-2	RG-3	RG-4	RG-L	RG-W	RG-S	RG-SU
Luhn	0.236	0.167	0.148	0.136	0.215	0.101	0.119	0.122
KL-Sum	0.239	0.1663	0.147	0.136	0.215	0.101	0.118	0.122
Sumbasic	0.240	0.169	0.149	0.137	0.218	0.103	0.120	0.124
TextRank	0.242	0.171	0.151	0.139	0.220	0.103	0.122	0.125
PRF	0.361	0.209	0.151	0.111	0.309	0.120	0.155	0.161
SE(Sogou)	0.361	0.204	0.161	0.138	0.298	0.130	0.149	0.154
FMN_{NS}	0.352	0.216	0.185^{*}	$0.167^{\dagger}$	0.283	0.131	0.157	0.161
FMN_{DS}	0.353	0.218	0.189^{*}	$0.172^{\dagger}$	0.291	0.135	0.160	0.165
FMN_{SS}	**0.385** +6.48%	**0.259**† +26.54%	**0.226**† +40.15%	**0.205**† +48.34%	**0.314** +5.36%	**0.148*** +13.25%	**0.183**† +22.68%	**0.188**† +21.73%

⋆⋆ Note that RG is short for ROUGE. Here we only present the results of four systems with highest performances in SUMY: *Luhn*, *KL-Sum*, *SumBasic* and *TextRank*.

According to the results in Table 2, we have the following main findings:

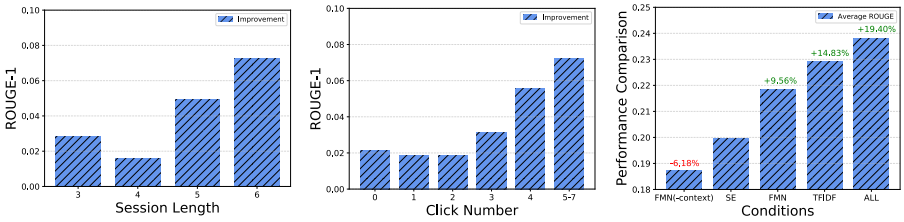
- Generally, unsupervised summarization methods have the worst performance.
- By taking query content into consideration, *PRF* and *SE* achieve better performances than the query-independent baselines by a large margin.
- Systems that leverage the feedback memory mechanism to model search context achieve better performances in all metrics, especially in ROUGR-3/4. Such an improved performance on long phrase matching suggests a successful capture of users' session-level information needs.
- Among all models, FMN_{SS} that combines context-aware relevance score with TF-IDF term has the best overall performances. Especially, it achieves a

[3] https://github.com/miso-belica/sumy

48.34% improvement over the *SE* baseline in ROUGE-4. Note that the models that utilize N-tuple Sorting (FMN_{NS}) and Diversified Sorting (FMN_{DS}) perform worse than FMN_{SS}. This indicates that forcing the algorithm to select consecutive sentences for the coherence and readability does not necessarily improve the snippet quality. Sentence-level consistency is enough for readers to get the main idea of the search result.

To sum up, the comparison results suggest that (1) the current query is crucial for search snippet generation; (2) the context information within a search session is helpful for snippet generation and that (3) our proposed method FMN_{SS} outperforms all the baseline systems.

Performances across session lengths & clicks To investigate the impact of context information, we further analyze the model performances across sessions with different lengths and click numbers. Figure 2(a) shows the recall improvements of FMN_{SS} over *SE* in ROUGE-1 across different session lengths. We can observe that the improvement of FMN_{SS} over *SE* is generally larger when a session is longer (e.g., length = 5 or 6). Also from Figure 2(b), we can find that our model can better exploit feedback information when there are more clicks in a session. Note that there is a drop in sessions with four queries, this may be caused by sparse click signals to be exploited by the feedback memory mechanism. Generally, our model can perform better by exploiting more clicks.
Ablation study To verify the results achieved by our model, we further compare the performance of each component in average ROUGE against the strongest baseline *SE* in Figure 2(c). We delete the context information by feeding empty query history and clicks into the network and test the system performance. We can find that there is a huge drop in overall performance (+9.56% to -6.18% compared to *SE*) for FMN without context information. This reveals that context information is crucial for user modeling and snippet generation. Both the context information captured by FMN and the TF-IDF term contribute to system performance. The combination of all techniques has the highest improvement of 19.40% over *SE*, which indicates the effectiveness of our model.



(a) Impro. w.r.t. session len. (b) Improv. w.r.t. clicks (c) Ablation Study

Fig. 2: System performance in different conditions. Figure (a) and (b) show the performances improvement of FMN_{SS} over *SE* in ROUGE-1 w.r.t. session lengths and click numbers. Figure (c) is the result of system ablation study.

4.3 User Preference Test

After evaluating the quality of the snippets generated by our models, we further explore whether people prefer the context-aware snippets by conducting a user preference test. To prepare data, we randomly extract 99 sessions from SD_1 and SD_2 to construct the annotation dataset. We then crawl the SERPs of the last queries in sessions from a search engine to expand the search results and generate snippets for them respectively. The annotation dataset is organized as **UPT**.

User perception on snippets is subtle, so it is both human- and time-consuming to conduct the side-by-side user preference test. We also pre-survey the user preferences among the baseline systems by collecting the annotation results in a small scope. The results have suggested that the *SE* system is much better than other baselines. So we only conduct a large-scale user preference test that compares the FMN_{SS} (the best of our models) with the *SE* (the best baseline).

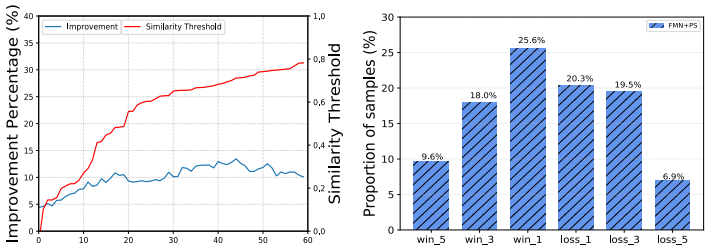
We recruit 45 participants aged from 17 to 25 to annotate their preferences for each pair of snippets. For each task, a search session and several pairs of snippets for the results of the last query in a session are presented to the participants. Similar to the snippet quality evaluation, the participants should take a glance at the query sequence and consider the search user's information needs. Then they need to click on the result page and skim the whole page from top to bottom. A pair of snippets for the page is presented vertically in the annotation page. The snippets share the same stylesheet as the original SERP but contain different contents generated by corresponding models. Modern web search engines may highlight some keywords in the snippet, which can influence the user's preference on the snippets. In this study, we only focus on the snippet content so we disable the keyword highlight feature. In addition, we also remove all the vertical results to avoid the influence of webpage layout and multimedia elements. Each participant should choose one snippet she prefers. The position of *SE* and testing snippets are randomly shuffled to avoid the position bias. In addition, to ensure the annotation quality, we impose a minimum annotation time by disabling the submit button for 15 seconds after the participant has entered the annotation page.

Table 3: The win/loss/tie cases for FMN_{SS} compared to *SE* in different comparison conditions. Improvements of FMN_{SS} over *SE* are calculated by $\frac{N_{win}-N_{loss}}{N_{win}+N_{loss}}$.

Conditions	Win	Loss	Tie	Improv.
All(100%)	448	410	26	+4.43%
Top 80%	384	311	24	+10.50%
Top 60%	290	229	19	**+11.75%**
Top 40%	191	156	12	+10.08%
Top 20%	95	79	8	+9.20%

Results and analysis Users' intents and information needs may shift within a search session. To this end, we calculate the cosine similarities between the max-

pooled GloVe vectors of the last query and previous queries in each session and sort these sessions by the similarities in descending order. We then analyze the results of the preference test for five conditions, where top 20%, 40%, 60%, 80%, and all sessions are considered. The numbers of win/loss/tie cases of sessions across the five conditions are shown in Table 3. Figure 3(a) depicts the system improvements when filtering different proportions (from 0 to 60%) of sessions with an increasing similarity threshold. Although FMN_{SS} model only has a 4.43% improvement over *SE* on all sessions, we find that once filtering some sessions with sharp intent shifts, the improvement will soon rise to over 10%. This can be explained by: feedbacks in previous search rounds can benefit those queries under the same topic to a greater extent. Intent shift may affect the system effectiveness thus should be considered in complex scenarios.



(a) System improvement across various filtering conditions.

(b) User preference distribution on FMN_{SS}.

Fig. 3: System performance w.r.t. filtering conditions and user preference distribution on FMN_{SS}. Note that on the right win_n represents $N_{win} - N_{loss} = n$ for FMN_{SS} in a case and $loss\_n$ represents $N_{win} - N_{loss} = -n$.

The distribution of users' preferences over the FMN_{SS} model and the baseline is shown in Figure 3(b). We can observe that the distribution is skewed towards the "win" side, which indicates that the participants prefer the snippets generated by FMN_{SS} over *SE*. Note that few cases have five consistent labels (e.g. the proportions of win_5 or $loss\_5$ are less than 10%), showing that user preference are diverse. To further ensure the robustness of evaluation results, we calculate average proportion scores and boolean scores of each case for *SE* and FMN_{SS} in top 60% and all session conditions. The proportion score is the percentage of "win" within five labels. The boolean score is the majority vote of all preference judgments on a single pair. As shown in Table 4, all scores of the FMN_{SS} model are higher than those of *SE*. The improvements in the proportion scores in both top 60% and all sessions condition are statistically significant. These results further demonstrate that users do prefer the snippets generated by the FMN_{SS} model, especially in search scenarios without sharp intent shifts.

Table 4: The average proportion and boolean scores of SE and FMN_{SS} in all and top 60% sessions conditions (* indicates a statistical significant improvement over SE with an independent t-test at $p < 0.01$, and σ^2 denotes the variance).

Score\Model	***SE***	**FMN**$_{SS}$	σ^2
Boolean(All)	0.4779	**0.5221**	0.2495
Percentage(All)	0.4804	**0.5196***	0.0825
Boolean(Top 60%)	0.4412	**0.5588***	0.2465
Percentage(Top 60%)	0.4683	**0.5317***	0.0800

5 Discussions and Conclusions

In this paper, we propose a novel model which adopts feedback memory mechanism to model users' session-level information needs and generate context-aware snippets. We further evaluate the proposed method via an automatic summary evaluation and a user preference test. Experiment results in both evaluation show the expressive power of our methods.

To sum up, our work has the following advantages. Firstly, compared to existing snippet generation methods which mainly rely on exact matching with query terms, our methods are better at capturing semantic features. Secondly, the model utilizes previous queries and clicks to model users' session-level information needs, which can boost its performance in multi-query sessions. Last but not least, the proposed model has a low inference latency and needs not any human labels to train thus can be easily adopted in commercial search engines. However, since our model is feedback-based, it may not adapt to sessions with dramatic intent shifts. Intent detection and search task identification methods should be explored for further improvement for context-aware snippets.

References

1. Wu B, Xiong C, Sun M, et al. Query suggestion with feedback memory network[C]//Proceedings of the 2018 World Wide Web Conference. International World Wide Web Conferences Steering Committee, 2018: 1563-1571.
2. Pennington J, Socher R, Manning C. Glove: Global vectors for word representation[C]//Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014: 1532-1543.
3. Maxwell D, Azzopardi L, Moshfeghi Y. A study of snippet length and informativeness: Behaviour, performance and user experience[C]//Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, 2017: 135-144.
4. Lin C Y. Rouge: A package for automatic evaluation of summaries[C]//Text summarization branches out. 2004: 74-81.
5. Wang C, Jing F, Zhang L, et al. Learning query-biased web page summarization[C]//Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM, 2007: 555-562.
6. Penin T, Wang H, Tran T, et al. Snippet generation for semantic web search engines[C]//Asian Semantic Web Conference. Springer, Berlin, Heidelberg, 2008: 493-507.

7. Ko Y, An H, Seo J. An effective snippet generation method using the pseudo relevance feedback technique[C]//Annual ACM Conference on Research and Development in Information Retrieval: Proceedings of the 30 th annual international ACM SIGIR conference on Research and development in information retrieval. 2007, 23(27): 711-712.
8. Sun J T, Shen D, Zeng H J, et al. Web-page summarization using clickthrough data[C]//Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 2005: 194-201.
9. Chen, J., Mao, J., Liu, Y., Zhang, M., & Ma, S. (2019, September). Investigating Query Reformulation Behavior of Search Users. In China Conference on Information Retrieval (pp. 39-51). Springer, Cham.
10. Erkan G, Radev D R. Lexrank: Graph-based lexical centrality as salience in text summarization[J]. Journal of artificial intelligence research, 2004, 22: 457-479.
11. Mihalcea R, Tarau P. Textrank: Bringing order into text[C]//Proceedings of the 2004 conference on empirical methods in natural language processing. 2004: 404-411.
12. Edmundson H P. New methods in automatic extracting[J]. Journal of the ACM (JACM), 1969, 16(2): 264-285.
13. Tombros A, Sanderson M, Gray P. Advantages of query biased summaries in information retrieval[C]//SIGIR. 1998, 98: 2-10.
14. Chuklin A, Markov I, Rijke M. Click models for web search[J]. Synthesis Lectures on Information Concepts, Retrieval, and Services, 2015, 7(3): 1-115.
15. Chung J, Gulcehre C, Cho K H, et al. Empirical evaluation of gated recurrent neural networks on sequence modeling[J]. arXiv preprint arXiv:1412.3555, 2014.
16. Carbonell J G, Goldstein J. The use of MMR, diversity-based reranking for reordering documents and producing summaries[C]//SIGIR. 1998, 98: 335-336.
17. Kumar V, Khattar D, Gairola S, et al. Identifying clickbait: A multi-strategy approach using neural networks[C]//The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. ACM, 2018: 1225-1228.
18. Luhn H P. The automatic creation of literature abstracts[J]. IBM Journal of research and development, 1958, 2(2): 159-165.
19. Steinberger J, Jezek K. Using latent semantic analysis in text summarization and summary evaluation[J]. Proc. ISIM, 2004, 4: 93-100.
20. Vanderwende L, Suzuki H, Brockett C, et al. Beyond SumBasic: Task-focused summarization with sentence simplification and lexical expansion[J]. Information Processing & Management, 2007, 43(6): 1606-1618.
21. Haghighi A, Vanderwende L. Exploring content models for multi-document summarization[C]//Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, 2009: 362-370.
22. Ageev M, Lagun D, Agichtein E. Improving search result summaries by using searcher behavior data[C]//Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval. ACM, 2013: 13-22.