Template-guided Clarifying Question Generation for Web Search Clarification

Jian Wang and Wenjie Li
Department of Computing, The Hong Kong Polytechnic University
jwanglvy@gmail.com,cswjli@comp.polyu.edu.hk

ABSTRACT

Clarification has attracted much attention because of its many potential applications especially in Web search. Since search queries are very short, the underlying user intents are often ambiguous. This makes it challenging for search engines to return the appropriate results that pertain to the users' actual information needs. To address this issue, asking clarifying questions has been recognized as a critical technique. Although previous studies have analyzed the importance of asking to clarify, generating clarifying questions for Web search remains under-explored. In this paper, we tackle this problem in a template-guided manner. Our objective is jointly learning to select question templates and fill question slots, using Transformer-based networks. We conduct experiments on MIMICS, a collection of datasets containing real Web search queries sampled from Bing's search logs. Our method is demonstrated to achieve significant improvements over various competitive baselines[1].

CCS CONCEPTS

• **Information systems** → **Web search engines**; • **Computing methodologies** → **Natural language generation**.

KEYWORDS

Search clarification; Joint learning; Template; Slot

ACM Reference Format:
Jian Wang and Wenjie Li. 2021. Template-guided Clarifying Question Generation for Web Search Clarification. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3459637.3482199

1 INTRODUCTION

Search queries are often short and ambiguous, which makes it difficult for search engines to identify the actual user intents [20]. Recently, search clarification has been recognized as an important technique to improve user search experience, especially in the interactive information seeking scenarios with "limited bandwidth" interfaces, such as on the speech-only and small-screen devices [2, 22]. This has motivated researchers to investigate possible approaches to clarify users' information needs by asking clarifying questions [10, 13, 15], which is of great significance in search.

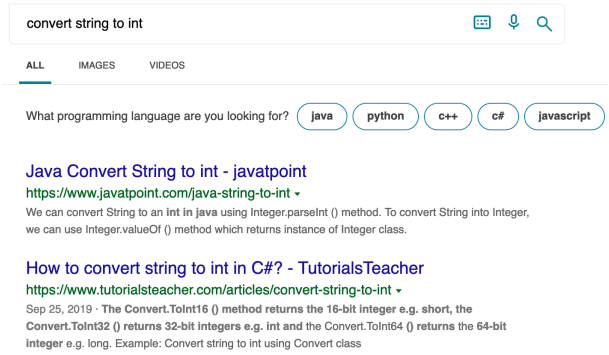

Figure 1: An example of asking for clarification in a Bing's search pane.

Recently, Aliannejadi et al. [1, 2] addressed clarification in the context of conversational search. They introduced public datasets (i.e., Qulac and ClariQ) and proposed an offline evaluation method. In a follow-up study, Hashemi et al. [7] developed a Guided Transformer model by leveraging multiple external sources for document retrieval and clarifying question selection. Rosset et al. [15] additionally considered to incorporate user past search behaviors. For clarification in the context of Web search, recent studies have shown that asking clarifying questions in order to better understand user queries gives a sense of confidence to users. It made a search engine look more intelligent [20]. Zamani et al. [22] further analyzed user interactions in Bing's search panes, and provided insights into user click bias and the impact of clarification on search experience. To facilitate the research progress in Web search clarification, Zamani et al. [21] lately introduced MIMICS, a collection of search clarification datasets containing real Web search queries sampled from Bing's search logs. To the best of our knowledge, MIMICS is the only publicly available dataset that focuses on Web search clarification. It is worth noting that MIMICS provides a unique resource in terms of size, realisticness, diversity, and clarification types, etc.

In this paper, we focus on the task of automatically asking clarifying questions for Web search clarification, which is relatively less explored. Figure 1 shows an example in response to the query "*convert string to int*". The search engine proactively asks the question "*What programming language are you looking for?*" for clarification. This happens because when given such an ambiguous query, the top-ranked search engine result pages (SERP) are so diverse that the search engine itself is not confident to tell whether the user is looking for string-to-integer conversion method for Java, Python, or

[1]Our code is available at https://github.com/iwangjian/TG-ClariQ.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia
© 2021 Association for Computing Machinery.
ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482199

any other programming language. In view of the nature of this task, it is a big challenge for a sequence-to-sequence method [4, 17] to generate clarifying questions directly as it can hardly well capture the intra-semantics of each SERP and the inter-patterns between different SERPs, which are crucial for what is to be clarified. Alternatively, if the task is formulated as selecting clarifying questions in a retrieval manner, the bottleneck is to select the most appropriate one from a large pool of question candidates.

After a closer look, we observe that most clarifying questions follow a few types of templates according to their purposes like disambiguation, comparison, asking for preference, or asking for sub-topic information. For example in Figure 1, "*What (.+) are you looking for?*" is a question template for disambiguation, and the full question can be completed by filling in the slot[2] (.+) with "*programming language*". Our statistical analysis on the MIMICS [21] dataset reveals that the common question templates, such as "*What (do you want | would you like) to know about (.+)?*" and "*(What | Which) (.+) are you looking for?*" etc. can actually match with over 95% of the clarifying questions. Motivated by this observation, we propose a simple yet effective template-guided clarifying question generation model, which employs Transformer [18] networks to enable deep interactions between user queries and SERP contents. Our goal is jointly learning to select the question template from a list of template candidates and fill in the question slot from a slot vocabulary.

The main contributions of this paper are summarized in two folds. (1) Our method is simple yet effective to address the challenge of asking clarifying questions for Web search clarification. It achieves superior performance compared to competitive baselines. (2) Different from generating clarifying questions end-to-end or selecting clarifying questions from a large pool of candidate questions, our method is more practical to be applied in online Web search since both the question templates to be selected and the question slots to be filled are in a very limited size.

2 METHODOLOGY

2.1 Problem Formulation

Let $Q = \{q_1, q_2, \cdots, q_n\}$ be the set of user queries, for each query $q_i (1 \leq i \leq n)$, let $S_{q_i} = \{s^1_{q_i}, s^2_{q_i}, \cdots, s^m_{q_i}\}$ denote the top-m search engine result pages (SERP) in response to q_i, where the content of each $s^j_{q_i} (1 \leq j \leq m)$ is the snippet of the Web page returned by the search engine. Given a user query q_i and SERP snippets S_{q_i}, the task of Web search clarification is to automatically ask a clarifying question c_i with the intention of clarifying the user's ambiguous information need.

2.2 Template-guided Question Generation

In this paper, we frame the task of asking clarifying questions for Web search clarification as the unified problem of question template selection and slot generation, with the objective of jointly learning to select the question template from a list of template candidates and fill in the question slot from a slot vocabulary. Inspired by the architecture of Guided-Transformer [7], we investigate a **T**emplate-**G**uided **Clari**fying **Q**uestion generation model, called **TG-ClariQ**, the architecture of which is illustrated in Figure 2. We use different text encoders to learn representations of user queries, SERP snippets, and question templates. We then employ Transformer [18] based networks to enable deep interactions between user queries and SERP snippets, as well as to capture relationships between SERP snippets and candidate question templates, which highlight the important information that needs to be clarified.

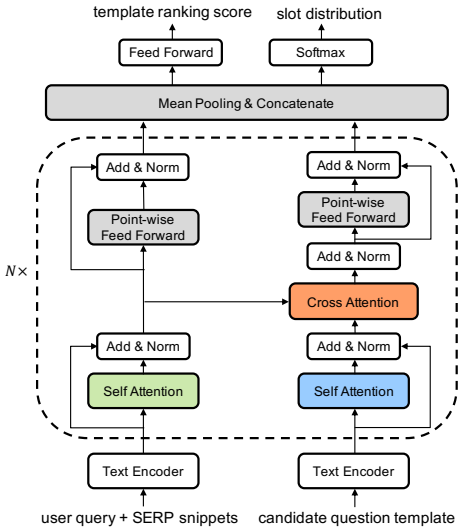

Figure 2: Overview of the TG-ClariQ architecture.

Text Encoder. The inputs contain two text sequences. The first part is the user query plus the corresponding SERP snippets. We concatenate the tokens of user query and SERP snippets using a separator `[SEP]`, and adopt `[CLS]` as the start token for the concatenated sequence, which is in consistent with the input to BERT [6]. The second part is the candidate clarifying question template, which is processed in the same way. We then feed each input sequence into an individual text encoder to obtain its hidden representation. The text encoder can be a recurrent neural network (e.g., LSTM [8]) or a BERT [6] encoder. We implement both varieties of text encoders in our experiments and provide analysis in Section 3.2.

Transformer-based Interaction. Our Transformer-based interaction is similar to the Guided-Transformer [7], with the key difference that our cross-attention can be directly computed without using the Bayes rule. Let $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_n\}$ be the output hidden representations generated by the text encoder from the concatenated sequence of user query and SERP snippets, we feed $\mathbf{H}$ to a self-attention layer to capture what information in the SERP snippets is important to emphasize with respect to the user query. It is crucial because the information needs to be clarified should be determined by looking into whether SERP snippets pertain to the user query, and whether their topics are similar or diverse. Concretely, the self-attention layer is computed based on three matrices, the query weight matrix $\mathbf{W}_Q$, the key weight matrix $\mathbf{W}_K$, and the value weight matrix $\mathbf{W}_V$. We multiply $\mathbf{H}$ to these three matrices to

[2]We use "slot" in this paper to denote an entity word or a noun phrase that appears in a clarifying question in response to the user query.

obtain the query matrix $\mathbf{Q}$, the key matrix $\mathbf{K}$, and the value matrix $\mathbf{V}$, respectively. The self-attention is then computed by:

$$\mathbf{Z} = \text{softmax}(\frac{\mathbf{Q} \times \mathbf{K}^T}{\sqrt{d}})\mathbf{V} \quad (1)$$

where d is the dimension of the vector. We add the representations $\mathbf{H}$ inputted to and the representations $\mathbf{Z}$ outputted from the self-attention layer, and proceed to layer normalization [3]. The output representations are denoted as $\mathbf{Z}_S$. Similarly, the encoded question template sequence representations are passed to another self-attention layer, followed by the addition operation and layer normalization, producing the output representations $\mathbf{Z}_T$.

To identify the actual information that needs to be clarified, we incorporate a cross-attention layer to enable interactions between candidate clarifying question templates and the concatenated sequence of user query and SERP snippets. The computation of cross-attention is similar to that of self-attention formulated by Eq. (1), with the differences that the query source comes from $\mathbf{Z}_T$, while the key source and the value source both come from $\mathbf{Z}_S$. On top of the self-attention outputs, the cross-attention layer helps to determine which question template is preferred. Next after the cross-attention layer are the two point-wise feed forward layers with ReLU activation functions, followed by the addition and layer normalization.

In general, such an architecture can be stacked to N layers for learning more complex representations, where N is a hyper-parameter. Finally, the last layer outputs two sets of representations in relation to two input sequences. We apply mean pooling to each of them and concatenate into a shared representation.

Training and Inference. We adopt the multi-task learning strategy to train our model. The first task is question template selection, and the second task is question slot generation. For the first task, we feed the shared representation to a fully-connected feed forward neural network (FFNN). We take the correct question template and k negative question templates as training examples. For the second task, we feed the shared representation to another FFNN, followed by a softmax operator. The overall loss function is given by:

$$L = \beta_1 L_{temp} + \beta_2 L_{slot} \quad (2)$$

where both loss functions L_{temp} and L_{slot} are defined using cross entropy. β_1, β_2 are hyper-parameters. Note that the output of template selection is binary (positive vs. negative), and optimizing L_{temp} is equivalent to point-wise learning to rank (LTR). The output of slot generation is a one-hot distribution over the slot vocabulary, and optimizing L_{slot} is similar to training a typical generator.

During inference, we take all question templates as candidate inputs for each user query. The template with the highest score is selected as the predicted clarifying question template. Meanwhile, we select the slot with the highest probability from the slot vocabulary as the generated question slot. The complete clarifying question is formed by filling the slot into the corresponding position of the selected question template.

It should be noted if there are two or more slots that need to be filled in a question template, our model can be extended by adding additional slot generation layers and designing extra strategies to determine the order of slot filling. Due to the single slot nature of the dataset, we leave this as a direction for future investigation.

3 EXPERIMENTS

3.1 Experimental Setup

Dataset. We use the MIMICS [21] dataset to conduct experiments. MIMICS is a collection of search clarification datasets sampled from the Bing's search logs, consisting of 3 subsets (MIMICS-Click, MIMICS-ClickExplore, and MIMICS-Manual). To enable quality evaluation of clarifying questions, we preprocess MIMICS-Click and MIMICS-ClickExplore by extracting <*query, clarifying question*> pairs with the "High" engagement level. For MIMICS-Manual, we extract <*query, clarifying question*> pairs with the "Good" question label. Each pair is associated with at most top-10 SERP snippets returned by the Bing's search API. We filter the pairs when the corresponding question templates occur less than 50 occurrences in the entire collection. After preprocessing, we obtain a total of 40,508 unique samples, and divide them into 38,508/1000/1000 samples for training/validation/testing, respectively. We obtain 8 question templates in total, which cover all clarifying questions in the samples. The statistics are summarized in Table 1 below.

Table 1: Statistics of the clarifying question templates extracted from the MIMICS dataset. Each (.+) denotes the position where a slot value needs to be filled.

ID	Clarifying question template	#train	#dev	#test
T1	select one to refine your search	12,000	325	308
T2	what (do you want | would you like) to know about (.+)?	10,662	230	233
T3	(which | what) (.+) do you mean?	8,607	147	151
T4	(what | which) (.+) are you looking for?	4,645	130	127
T5	what (do you want | would you like) to do with (.+)?	1,988	89	105
T6	who are you shopping for?	300	40	37
T7	what are you trying to do?	227	30	29
T8	do you have any (specific | particular) (.+) in mind?	79	9	10

Implementation Details. We compare two text encoders, i.e., LSTM and BERT. For LSTM, we use pre-trained 300-d Glove [12] word vectors as initialized word embeddings. The hidden size is 256. For BERT, we use the pre-trained BERT-base model (12 layers, 768 dimensions, 12 heads and 110M parameters). The slot vocab is built by counting the slot values of the clarifying questions in the dataset plus two extra special slots, `<EMPTY>` and `<QUERY>`. The `<EMPTY>` indicates no slot value is required in the question template, e.g., T6 and T7 in Table 1. The `<QUERY>` means that the slot value should be copied from user query tokens. The slot vocab size is 258. The number of stacked layers N is 3. The number of negative samples k is 2. The dropout ratio is 0.1 in all hidden layers. The length of the concatenated sequence of user query and SERP snippets is limited to 512 (i.e., the BERT maximum sequence length). The hyper-parameters β_1 and β_2 are set to 1.0. We use the Adam optimizer [9] with the initial learning rate of 2×10^{-5}, the L_2 weight decay of 0.01, learning rate warm-up over the first 5000 steps, and the linear decay of the learning rate. The batch size is set to 8 due to memory constraints. The other hyper-parameters are selected based on validation performance.

Baseline Methods. We validate the effectiveness of TG-ClariQ in selecting correct question templates and determining correct slot values. To this end, we first compare with several methods that extract clarifying questions from a large pool of candidate questions

(denoted as clarifying question selection, CQS). For fair comparison, we evaluate the templates selected by our model and the templates that match with the questions selected by those methods. The choices on slot values are checked as well. We also examine how these methods work when they directly select clarifying question templates (denoted as clarifying template selection, CTS). Since our method resembles to template-based generation, we also compare with two methods that generate clarifying questions end-to-end (denoted as clarifying question generation, CQG). The baseline methods are described as follows:

- **BM25**: For each user query and the corresponding SERP snippets, we retrieve the clarifying question (or template) using BM25 [14].
- **RankNet** and **LambdaMART**: We consider the task of CQS or CTS as a ranking problem, where a list of candidate questions (or templates) are ranked and the one with the highest rank is chosen. We use RankNet [5] and LambdaMART [19] as two LTR baselines. The concatenated BERT representations of user query and SERP snippets are fed as the input features.
- **BERT**: We fine-tune the pre-trained BERT model [6] with an additional feed forward neural network for output prediction. The clarifying question (or template) with the highest prediction score among the list of candidates is chosen.
- **Seq2Seq-LSTM/+Copy**: We take user query and SERP snippets as input and adopt sequence-to-sequence (Seq2Seq) with attention [4] to generate the clarifying question. Both encoder and decoder are LSTM [8] based. We further incorporate the copy mechanism [16] to enhance Seq2Seq generation.
- **Seq2Seq-Transformer+Copy**: We replace the LSTM based encoder/decoder in **Seq2Seq-LSTM+Copy** with a more advanced Transformer [18] based encoder/decoder.

Evaluation Metrics. For CQS and CTS, we use accuracy as the evaluation metric, which is computed by template match. We also use mean reciprocal rank (MRR) with ranking cut-off of @3 to evaluate the quality of the top-3 ranked questions (or templates). For CQG, we use BLEU [11] and entity F1 as evaluation metrics. BLEU calculates the n-gram overlaps between generated questions and gold questions. Entity F1 is computed by micro-averaging precision and recall over slot values (i.e., targets for clarifying). It evaluates the performance of generating proper entities to complete clarification.

3.2 Results and Discussion

The experimental results are reported in Table 2. According to the performance of clarifying question template selection, all the CQS baseline methods are weak. It is because they lack deep interactions between user query and SERP snippets, making them difficult to learn complex representations to identify the information to be clarified. All these baseline methods except for BM25, achieve significant improvements in CTS. This is not surprising. The number of question candidates and the number of question template candidates differ significantly and many questions share the same template. This characteristic makes CTS an easier task compared to CQS. However, the performance of BM25 in CTS is greatly disappointed. After careful thought, we realize that BM25 is a word-level retrieval method, while the templates seldom contain the words common to either user query or SERP snippets. Overall, we believe that question templates are essential for effective clarification. Compared to all baselines, our models achieve statistically significant improvements. For instance, TG-ClariQ-LSTM is superior to RankNet, LambdaMART, and even comparable to BERT, while TG-ClariQ-BERT achieves 6.8% relative improvement in accuracy compared to fine-tuned BERT in CTS. All these observations verify the effectiveness of our models, in which self-attention and cross-attention together encourage adequate information interactions between user query, SERP snippets and candidate templates.

Table 2: Experimental results. * denotes statistically significant improvements compared to all baselines ($p < 0.005$).

	Methods	Accuracy	MRR@3	BLEU	Entity F1
CQS	BM25	0.355	0.399	n.a.	0.414
	RankNet	0.308	0.384	n.a.	0.203
	LambdaMART	0.490	0.564	n.a.	0.214
	BERT	0.394	0.440	n.a.	0.356
CTS	BM25	0.095	0.191	n.a.	n.a.
	RankNet	0.323	0.455	n.a.	n.a.
	LambdaMART	0.564	0.621	n.a.	n.a.
	BERT	0.676	0.794	n.a.	n.a.
CQG	Seq2Seq-LSTM	n.a.	n.a.	45.30	0.166
	Seq2Seq-LSTM+Copy	n.a.	n.a.	52.64	0.495
	Seq2Seq-Transformer+Copy	n.a.	n.a.	55.37	0.546
	TG-ClariQ-LSTM	0.659	0.791	55.05	0.682
	TG-ClariQ-BERT	**0.722***	**0.827***	**60.49***	**0.788***

In comparison to the CQG baselines, our models achieve noticeably higher BLEU scores and entity F1 scores. Although it might not be strictly comparable in BLEU, we report BLEU scores here to show that our models are prone to generate the same clarifying questions as the ground truths. Notably, our models significantly outperform Seq2Seq models in entity F1. Seq2Seq models are incapable of generating right entity words from a large word vocabulary, since they are difficult to well capture the intra-semantics of each SERP snippet and the inter-pattern between different SERP snippets. By contract, our models employ specific supervision for slot generation by using multi-task learning and generate entity words from a limited size of slot vocabulary, which is proven to be effective. It is also interesting to note that F1 of BM25 is even better than BERT in CQS. When the slot value is a part of the query, the method relying on word-matching or word copy is advantageous. This indirectly confirms the importance to the copy mechanism and is consistent with the observations over the CQG methods.

4 CONCLUSION

In this paper, we investigate a template-guided clarifying question generation method for Web search clarification, with the objective of jointly learning to select appropriate question templates and fill with slot values. Extensive experiments are conducted on a collection of datasets comprised of real Web search queries, which provide many insights of the task and demonstrate the effectiveness of our method. We intend to further explore how to generate the answer options that are paired with the clarifying questions.

ACKNOWLEDGMENTS

The work described in this paper was supported by Research Grants Council of Hong Kong (15207920, 15207821) and National Natural Science Foundation of China (62076212).

REFERENCES

[1] Mohammad Aliannejadi, Julia Kiseleva, Aleksandr Chuklin, Jeff Dalton, and Mikhail Burtsev. 2020. ConvAI3: Generating Clarifying Questions for Open-Domain Dialogue Systems (ClariQ). *arXiv preprint arXiv:2009.11352* (2020).

[2] Mohammad Aliannejadi, Hamed Zamani, Fabio Crestani, and W Bruce Croft. 2019. Asking clarifying questions in open-domain information-seeking conversations. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 475–484.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer Normalization. *arXiv preprint arXiv:1607.06450* (2016).

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to Rank Using Gradient Descent. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*. 89–96.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. 4171–4186.

[7] Helia Hashemi, Hamed Zamani, and W Bruce Croft. 2020. Guided Transformer: Leveraging Multiple External Sources for Representation Learning in Conversational Search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1131–1140.

[8] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Vaibhav Kumar, Vikas Raunak, and Jamie Callan. 2020. Ranking Clarification Questions via Natural Language Inference. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM)*. 2093–2096.

[11] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics (ACL)*. 311–318.

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[13] Sudha Rao and Hal Daumé III. 2018. Learning to Ask Good Questions: Ranking Clarification Questions using Neural Expected Value of Perfect Information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. 2737–2746.

[14] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Proceedings of The Third Text REtrieval Conference, TREC (NIST Special Publication, Vol. 500-225)*. 109–126.

[15] Corbin Rosset, Chenyan Xiong, Xia Song, Daniel Campos, Nick Craswell, Saurabh Tiwary, and Paul Bennett. 2020. Leading Conversational Search by Suggesting Useful Questions. In *Proceedings of The Web Conference 2020*. 1160–1170.

[16] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*. 1073–1083.

[17] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*. 3104–3112.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems (NeurIPS) 2017, December 4-9, 2017, Long Beach, CA, USA*. 5998–6008.

[19] Qiang Wu, Christopher JC Burges, Krysta M Svore, and Jianfeng Gao. 2010. Adapting Boosting for Information Retrieval Measures. *Information Retrieval* 13, 3 (2010), 254–270.

[20] Hamed Zamani, Susan Dumais, Nick Craswell, Paul Bennett, and Gord Lueck. 2020. Generating Clarifying Questions for Information Retrieval. In *Proceedings of The Web Conference 2020*. 418–428.

[21] Hamed Zamani, Gord Lueck, Everest Chen, Rodolfo Quispe, Flint Luu, and Nick Craswell. 2020. Mimics: A Large-scale Data Collection for Search Clarification. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management (CIKM)*. 3189–3196.

[22] Hamed Zamani, Bhaskar Mitra, Everest Chen, Gord Lueck, Fernando Diaz, Paul N Bennett, Nick Craswell, and Susan T Dumais. 2020. Analyzing and Learning from User Interactions for Search Clarification. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1181–1190.