Reproduction Study: Controlling the Risk of Conversational Search via Reinforcement Learning

Niek IJzerman
11318740

Jeroen van Wely
11289988

Jochem Soons
11327030

Abstract

1 Introduction

An important reason for the retrieval of unreliable results within modern IR is the vague or immature formulation of queries by users. Several techniques have been proposed to address this problem. Examples are query auto-complete (Hu et al., 2015) and query suggestion (Sordoni et al., 2015). In particular, conversational retrieval in which a system asks clarifying questions to the user is believed to be a key technique for future IR (Radlinski and Craswell, 2017). However, previous work on conversational retrieval often assumes a dedicated user that would provide responses to any question asked by a retrieval system. This assumption is over-optimistic as it neglects the fact that bad questions could be asked, which could drive the user away from using the IR system again.

To address this problem, Wang and Ai (Wang and Ai, 2021) propose a risk-aware conversational retrieval system that is able to control and balance the risk of retrieval reliability and user engagement. More precisely, the authors introduce a conversational retrieval system consisting of three components: (1) A result retrieval module that retrieves and ranks candidate results according to the current information requests and corresponding context, (2) A question retrieval module that finds or generates clarifying questions for users and (3) a risk decision module that decides whether the system should ask the question provided by the question retrieval module or directly show the documents provided by the result retrieval module. This risk decision module is the key to controlling and balancing the risk of asking poor questions or providing low-quality results to users.

The contributions of the authors are stated as following: (1) They propose a new risk-aware conversational agent which takes the risks into consideration when deciding to ask clarifying questions or answer the query given conversation context. (2) They introduce a Reinforcement Learning approach to train proposed agent without having annotated data for when to ask clarifying question and when to give answer. (3) They show that their risk-aware conversational search agent could improve both answer quality and user experience in interacting with the retrieval system.

2 Interactive user-agent model

In this section, we will discuss the interactive user-agent model proposed by Wang and Ai. The risk-aware conversation agent aims to answer a user's query question correctly. It will do so by entering into a conversation with the user. A conversation is constructed into turns. Each turn starts with the information provided by the user and ends with a reaction by the agent. The start of a conversation, turn one, is always the user's query question. From that point on, the agent can choose one out of two actions. Either asking a clarifying question to ensure the agent "understands" the question better or answering the query question with the current context information already available. Either one of these actions will end the current turn. When the agent asks a question, the user can react by answering the question or leaving the conversation. When the agent tries to answer the query question, the conversation is ended. Each turn will thus start with new context information, which is either the user's query question or the user's answer to the clarifying question asked by the agent in the previous turn. The rankers (section 2.1) will first rank the answers and clarifying questions based on the extended context information. The decision-maker (section 2.2) then uses these rankings to decide whether to ask a clarifying question or answer the user's query question. We will now go into more detail for each of the individual components of the model.

2.1 Answer and Question Reranker

As stated above, the goal of the answer and (clarifying) question reranker is to retrieve the best answer and question given the current context information. To demonstrate that the decision-maker (section 2.2) can work with any answer and (clarifying) question reranker, Wang and Ai present the model's performance using two different rerankers. These are the ParlAI bi-, and poly-encoder rerankers (Humeau et al., 2019). For both encoder implementations of the model, the same reranker structure is used with different sets of parameters. Which parameter settings have been used will be discussed in section 3.2.

If both encoders are compared, the poly-encoder tends to have the highest performance while maintaining reasonable efficiency. It uses a similar structure to the bi-encoder, where both the context and question/answer are encoded separately. However, it extends this system with an additional mechanism where the input context is jointly encoded with the question/answer encoding. Therefore, the bi-encoder is slightly faster. However, the performance is noticeably lower in both the original paper (Humeau et al., 2019), and the paper by Wang and Ai (Wang and Ai, 2021).

2.2 Decision making model

The decision-maker model uses a Deep Q Network (DQN) to decide whether to answer the user's query question or ask a clarifying question. The DQN first uses a BERT encoder to encode the query question (q), context information (h), and the top-k ranked clarifying questions ($cq^1, ..., cq^k$) and answers ($a^1, ..., a^k$). Together with the top-k clarifying questions and answers scores ($s_{cq}^{1:k}$, and $s_{ans}^{1:k}$) respectively), these encodings are concatenated and given as input to a two-layer feed-forward neural network. We define the input by:

$$S = (q, h, cq^1, ..., cq^k, a^1, ..., a^k, s_{cq}^{1:k}, s_{ans}^{1:k}) \quad (1)$$

The network will output a prediction of the rewards for asking a clarifying question and answering the query question in the form of a 2×1 vector $y_{\text{pred}} = (r_{\text{ans}}, r_{cq})$. The Decision Maker DQN is represented as:

$$DQN(S) = W_2 \cdot \phi\left(W_1 \cdot S + b_1\right) + b_2 \quad (2)$$

Whith ϕ is a ReLU activation function.

2.3 Training

Rerankers. Both the question- and answer-reranker are pretrained on the Reddit dataset [bron]. They are then finetuned on the answer and question datasets, respectively. During tuning, the training set is divided into batches of 100 conversation-response pairs, of which only the true response is used as a positive sample. All other responses are used as negative samples. Both rerankers are trained to minimize the cross-entropy loss between the batch relevance label vector and the batch similarity dot product logits vector. The batch relevance label vector will have a single one corresponding to the true response. All other values will be zero. The batch similarity dot product logits vector is a vector containing the scores for each response. Such a score is computed by the Bi- and Poly-encoders.

Decision Maker. The DQN is trained via Reinforcement Learning by simulating the interactions between user and agent and defining a reward (r) for each action (a) made by the DQN. Note that a reward can also be negative and that an action is either answering the user's query question or asking a clarifying question. Additionally, each user is modeled with a bad question tolerance. This means that the agent can only ask so many bad questions in each turn before the user leaves and thus ends the conversation. The rewards for each action are defined as follows:

- When action a is to answer the user's query question, the conversation ends, and the DQN receives a reward r equal to the Mean Reciprocal Rank (MRR) of the top-k reranked answers by the answer-reranker. Because there is only one correct answer in the top-k reranking, the reward will therefore always be equal to $\frac{1}{r}$ where r is the rank of the true answer within the reranking. For example, with a batch size of 10, the maximum reward for answering will be 1, and the minimum will be $\frac{1}{10}$.
- When action a is to ask a clarifying question, and the correct question's rank r is lower than or equal to the bad question tolerance of the user, a reward will be given o $r_{cq} = 0.21$.
- When action a is to ask a clarifying question, and the correct question's rank r is higher then the bad question tolerance of the user, a penalty will be given of $p_{cq} = -0.79$.

Where r_{cq} and p_{cq} are two hyperparameters that are used as constants once they are set.

The goal of the DQN is to predict the reward r of taking action a given its input state S where S is previously defined in equation 1. It does this by minimizing the mean squared error (MSE) loss between the rewards prediction $y_{\text{pred}}\,(S)_a$ and its target $y_{\text{target}}\,(S)_a$ during training. The loss that is minimized is therefore defined as:

$$L = \text{MSE}\left(y_{\text{target}}\,(S)_a, y_{\text{pred}}\,(S)_a\right) \quad (3)$$

The predicted reward is defined by the output of the DQN:

$$y_{\text{pred}}\,(S)_{\text{a}} = DQN(S)_{\text{a}} \tag{4}$$

Note that the output of the DQN is a 2×1 vector $y_{\text{pred}} = (r_{\text{ans}}\,, r_{cq})$. The target reward is defined by:

$$y_{\text{target}}(S)_a = r_a + \gamma \cdot \max_{a'=ans,cq} DQN(S')_{a'} \tag{5}$$

Where r_a is the obtained reward by taking action a in state S, γ is a discount factor which controls how much we care about future actions, S' is the new state we are in after taking action a in the original state S. $\max_{a'=ans,cq} DQN(S')_{a'}$ is the maximum predicted reward by our DQN given the new state S' for any action a that leads to the highest predicted reward. Note that if our current action a ends the conversation, S' is a terminal state. The predicted reward of a terminal state is always 0 and therefore if S' is a terminal state: $\max_{a'=ans,cq} DQN(S')_{a'} = 0$. This would be the case for the actions that are answering the user's query question and exceeding the bad question tolerance of the user. In those cases, the target reward becomes equal to just the reward obtained by taking action a in state S:

$$y_{\text{target}}\,(S)_{\text{ans}} = r_{\text{ans}} \tag{6}$$

$$y_{\text{target}}\,(S)_{cq} = p_{cq} \tag{7}$$

Only taking the action of asking a clarifying question and the correct questions rank not exceeding the user's bad question tolerance will not lead to a terminal state and therefore:

$$y_{\text{target}}(S)_{cq} = r_{cq} + \sigma \cdot \max_{a'=ans,cq} DQN(S')_{a'} \tag{8}$$

The DQN will start with a random exploration policy. Meaning it chooses its actions randomly given any state. Gradually the DQN will start choosing its actions not randomly anymore but start choosing them based on the highest predicted reward. The DQN is trained until it is converged.

3 Reproducing the results

Within this section we will address each part of the reproduction process that has been conducted. For each part, we will describe details and instructions from the paper and point out inconsistencies and mistakes we encountered when reproducing the research.

3.1 Datasets

Wang and Ai (Wang and Ai, 2021) use the MSDialog dataset (Qu et al., 2018), (Qu et al., 2019), (Yang et al., 2018) for their experiments. This dataset consists of question-answer conversations from an online forum for Microsoft products. This data is preprocessed as following:

1. The MSDialog data contains discussions where multiple agents are involved. Therefore, an assumption is made that all agents share the goal of correctly answering the user's question(s). Hence, the agents are merged into a single agent based to ensure that each conversation only has two participants (i.e. one user and one agent).
2. All consecutive responses from a user are merged into a single response within a conversation to ensure that all conversations are conducted in turns.
3. After the above preprocessing steps, a filtering step is performed to ensure that all conversations are between four and ten turns long.
4. In the MSDialog dataset, each turn has a binary label indicating if this reply is voted as the definitive answer by the community. These labels are used in the final filter step that ensures all conversations end in a correct final answer.

All the conversations filtered out within the preprocessing steps are used to train and test the question rerankers. These are conversations either not having a correct final answer or conversations that are too long or too short. We ran the preprocessing steps and found our datasets statistics to be comparable to that of the original paper:

Table 1: MSDialog and our dataset statistics

Item	MSDialog	Orig. & Repr.
conversations	35000	3762-0000
Max. turns	1700	10-00
Min. turns	3	4-0
Avg. turns	8.94	4.70-0.00

Additionally, there are two points of critique we would like to make concerning the handling of the data. Firstly, during training of the DQN, batch selection is performed deterministically. Each epoch uses the exact same batches in the exact same order. The authors may have implemented the training in this way to reduce the computational load as making the batch selection deterministic gives the opportunity to reuse the question and answer embeddings. After epoch one, all embeddings have been computed and stored in memory. Each epoch after this can then reuse these embeddings by loading them from memory. However, it is doubtful whether deterministic batches in combination with a batch size of 10 improves training and subsequently the quality of the model as it results in a very low diversity of negative samples. We will address this issue in section 5 as well.

Another point of concern is that during training, only the original answer from the conversation is considered correct. However, it might be the case that answers from other conversations could also be considered correct, especially because most conversations seem to be centered around the same general topic: Microsoft. Additionally, some answers might not be entirely correct but at least "more correct" than others.

3.2 Baselines

In the original paper, five models are used as baselines. The first three are **Q0A**, **Q1A**, and **Q2A**, which are models that answer the user's query question after 0, 1 and 2 question-answer turns, respectively. Additionally, **CtrPred** is used as a baseline as well. This risk-unaware model first employs a binary classifier to predict whether to answer the query or ask a clarifying question given the current information context. It then uses these classification results to use either the question- or answer-reranker to rank and select the top response. The code for CtxPred was not provided in the Github repository made publicly available by the authors. Therefore, we have not been able to reproduce the results for the CtxPred model. Finally, an **oracle model** is included that always knows when to answer the user's query question or ask a clarifying question. For all of these models, the same reranker and parameter settings have been used.

3.3 Evaluation metrics

The evaluations metrics used are recall@1, Mean Reciprocal Rank (MRR), and the decision-making accuracy. We will now briefly explain their definitions within the context of this paper.

The **recall@1** metric is defined as the frequency of conversations where the agent eventually gives the ground truth answer regardless of how many clarifying questions it has asked. Note that if the agent exceeds the bad-question-tolerance of the user, the conversation is terminated, and thus the agent can not answer the user's query question leading to a recall@1 score of 0. recall@1 is considered as the direct measure of the answer quality.

The **MRR** metric also does not consider the amount of clarifying questions proceedings the agent's answer. Note here that recall@1 evaluates the top answer, while MRR evaluates the entire reranking of length k. The ground truth is always present in this reranking, and therefore, the minimum MRR score for answering is $\frac{1}{k}$. However, suppose the agent exceeds the bad-question-tolerance of the user. In that case, the conversation is terminated, and thus the agent can not answer the user's query, which leads to an MRR score of 0. MRR, therefore, punishes the asking of bad questions more severely compared to answering the user's query question with a wrong answer. Thus MRR should not be used as a primary evaluation metric because it is biased to answer the user's query question.

The **decision error** measures the frequency of the agent making the wrong decision, where a wrong decision can either be exceeding the bad question tolerance of the user or answer the user's query question with the wrong answer. It, therefore, mainly measures the ability of the model to avoid risks.

3.4 Hyperparameters & Technical details

The parameters used to obtain our results are presented in table 2. We used the same hyperparameters settings as in the original paper.

Table 2: Hyperparametersettings

Hyperparameter	Abbr.	Value
Good question reward	r_{cq}	0.21
Bad question penalty	p_{cq}	-0.79
Learning rate	lr	10^{-4}
Discount factor	γ	0.79
Regularization weight	λ	10^{-2}
Batch size	$batch\_size$	$10/100$

Additionally, what was not entirely clear from the original paper is that the ablation study (results in table 5 of the original paper) has been performed with a batch size of 100. All other experiments use a batch size of 10.

The implementations for the bi- and poly encoder are from ParlAI. The Agent (DQN) and User are written by the authors from scratch using Pytorch. We ran the code on the Lisa GPU server provided by the University of Amsterdam (see https://userinfo.surfsara.nl/systems/lisa for more information).

3.5 Running the code

When we tried to run the code for the first time, we immediately experienced quite some errors in the code. Firstly, to finetune the encoders, dependencies psutil, typing-extensions and egg-subword had to be installed additionally to the packages provided by the authors in their GitHub repository. Secondly, we had to adjust the code in metric.py to avoid None value errors. Thirdly, in train_model.py within the directory conversationalQA-master/ParlAI/examples it was neccesarry to add the path to the ParlAI directory, which was not clear from the authors' instructions. Fourthly, we ran into dictionary size mismatch errors when the pretrained models were loaded in. Although the authors provided a fix for this issue within their GitHub, it did not provide a full solution. We did notice via a GitHub forum that other researchers encountered the same issue and that the authors did provide the full solution within that thread. This fix, however, seems not to be updated in the authors' GitHub repository.

Training and evaluation the actual models by calling run_sampling.py resulted in some complications as well. Firstly, we ran into a type error in agent.py as the targets for the model were ints while floats where expected. Secondly, the path to the ParlAI directory needed to be added run_sampling.py. Thirdly, the weight decay of the score agent was set to zero while it needed to be 0.01 to be in line with the paper. Fourthly, we encountered index out of range errors while updating the memory within the code which was fixed as well.

3.6 Inspection of code pipeline

While the fixes mentioned in section 3.5 allowed us to train and evaluate the model, we also inspected the code pipeline more closely to determine if it was aligned with the pipeline stated in the paper. This led to some interesting insights. First of all, the tolerance parameter was being initialized but not used within the code. Therefore, adjusting the tolerance parameter did not have any effect on the model training. Instead, all experiments were effectively ran with a tolerance of 0. **@jochem even kijken of het volgende stuk klopt na al die aanpassingen die je nog had gedaan en of er nog wat bij moet**. Secondly, the evaluation of all models (i.e. original risk aware conversational agent and baseline models Q0A, Q1A, etc.) was done in parallel. At the same time, if the authors' risk aware conversational agent had closed a conversation (by either giving back the results or exceeding tolerance/patience) the conversation of the baseline models was also ended and a penalty was given to these models. Due to the fact that most conversations naturally ended in less than two turns, the evaluation scores of the Q1A and particularly Q2A were poor (see results of paper in table 3). However, as this approach of evaluation and comparison of the baseline models against the proposed risk aware model is unfair, we choose to adjust the pipeline such that baseline models were not punished disproportionate when the risk aware model had already finished its conversation. Subsequently, this made the comparison of results between models fair.

4 Results and Analysis

Explain what experiments we have reproduced. Why did we not reproduce the other experiments? We need to say here that we did not do 5 fold cross-validation.
Present, discuss and compare Tables 3 and 4. Shall we extend our table 4 with the 2-tolerance column as well? **niet echt nodig**
Additionally, explain what was wrong with the code with regard to these two experiments. **dit uitleggen moet al in de reproducing the results sectie gebeuren anders komt het verschil van die resultaten uit het niks**
Also, discuss training time and that it took much longer as stated in the paper.

5 Extension: BM25 negative sampling

While the authors state that conversations to sample negative answers and questions from are chosen randomly from the dataset, these conversations are picked by deterministically looping over the train and test set. Furthermore, the procedure to sample negative answers and questions from these conversations is deterministically as well. Subsequently, the batches used for training are the same for each epoch. As deterministically sampling negative training instances might not be a desired property, we have implemented a sampling framework that utilizes the BM25 algorithm to sample negatives. By using this framework we do no longer sample negative questions and answers from conversations for training randomly and locally, but construct our negatives globally by taking the best

Table 3: Comparison of all models using poly-encoder. The original paper's results are presented in red. The reproducibility results are presented in blue. Numbers in bold mean the results is the best excluding the oracle. † indicates $p < 0.01$ statistical significance over the best among baseline models.

Users	0 -tolerance			1-tolerance			2 -tolerance		
Models	R@1/100	MRR	Dec. err	R@1/100	MRR	Dec. err	R@1/100	MRR	Dec. err
Q0A	0.775-0.000	**0.8398**-0.000	0.1975-0.000	0.7475-0.000	0.8398-0.000	0.1975-0.000	0.7475-0.000	0.8398-0.000	0.1975-0.000
Q1A	0.7525-0.00	**0.8044**-0.00	0.1225-0.000	0.7850-0.000	0.8494-0.000	0.0650-0.000	0.8200-0.000	0.8723-0.000	0.225-0.000
Q2A	0.0075-0.000	**0.0075**-0.000	0.9925-0.000	0.0075-0.000	0.0075-0.000	0.9925-0.000	0.0075-0.000	0.0075-0.000	0.9925-0.000
CtxPred	0.7400-0.000	0.7960-0.000	0.1275-0.000	0.7850-0.000	0.8530-0.000	0.0575-0.000	0.8200-0.000	0.8723-0.000	0.0225-0.000
Ours	**0.7775**-0.000	0.8305-0.000	**0.0975**†-0.000	**0.7875**-0.000	**0.8530**-0.000	**0.0575**†-0.000	**0.8200**†-0.000	**0.8723**†-0.000	**0.0225**†-0.000
Oracle	0.8575-0.000	0.9139-0.000	0-0	0.8650-0.000	0.9169-0.000	0-0	0.8925-0.000	0.9324-0.000	0-0

Table 4: Comparison of all models using bi-encoder. The original paper's results are presented in red. The reproducibility results are presented in blue. Numbers in bold mean the result is the best excluding oracle. † means $p < 0.05$ statistical significance over the Q0A baseline.

Users	0 -tolerance			1-tolerance		
Models	R@1/100	MRR	Dec. err	R@1/100	MRR	Dec. err
Q0A	0.6725-0.0000	**0.7981**-0.0000	0.2425-0.0000	0.6725-0.0000	0.7981-0.0000	0.2425-0.0000
Q1A	0.7200-0.0000	0.7784-0.0000	0.1400-0.0000	**0.7700**-0.0000	**0.8352**-0.0000	0.0725-0.0000
Q2A	0.0075-0.0000	**0.0075**-0.0000	0.9925-0.0000	0.0075-0.0000	0.0075-0.0000	0.9925-0.0000
CtxPred	0.7200-0.0000	0.7784-0.0000	0.1350-0.0000	0.7675-0.0000	0.8327-0.0000	0.0700-0.0000
Ours	**0.7375**†-0.0000	**0.7982**-0.0000	**0.1225**†-0.0000	0.7600-0.0000	0.8337-0.0000	**0.0650**†-0.0000
Oracle	0.8200-0.0000	0.8898-0.0000	0-0	0.8600-0.0000	0.9156-0.0000	0-0

matching BM25 conversations compared to the true positive conversation and substract negative questions and answers randomly from these best matching BM25 conversations. Subsequently, the created batches contain more informative negatives that are semantically closer related to the true positive conversation. We expect that this change of sampling negatives improves the distinctiveness of the model for positive and negative instances and therefore leads to better scores. In table ... we present our results retrieved from our BM25 negative sampling pipeline.

6 Conclusion

7 Further Research

Overall, incorporating risk using Reinforcement Learning (RL) within conversational search is a very intriguing idea. We think there are many possibilities concerning extending Wang Ai's research. It would be interesting to analyze the performance of this model on other datasets. Currently, the MSDialog dataset is limited to the subject of Microsoft products only. Using a dataset that contains multiple different topics might be interesting.Additionally, it might be interesting to make a model performance comparison between different RL algorithms. Finally, We have tried to extend the research by incorporating BM25 negative sampling. However, this method of sampling is based on sparse information. We think the incorporation of negative sampling using dense retrieval could improve the performance of the model.

8 Acknowledgment

We would like to thank Mohammad Aliannejandi and Evangelos Kanoulas for the opportunity, guidance, and supervision during this reproducibility work. We would also like to extend our gratitude to Zhenduo Wang and Qingyao Ai. They have been very helpful during the complete process of conducting this research.

References

Wenyan Hu, Xiaodi Zhang, Alvaro Bolivar, and Randall Scott Shoup. 2015. Predictive algorithm for search box auto-complete. US Patent 8,990,240.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring. *arXiv preprint arXiv:1905.01969*.

Chen Qu, Liu Yang, W Bruce Croft, Johanne R Trippas, Yongfeng Zhang, and Minghui Qiu. 2018. Analyzing and characterizing user intent in information-seeking conversations. In *The 41st international acm sigir conference on research & development in information retrieval*, pages 989–992.

Chen Qu, Liu Yang, W Bruce Croft, Yongfeng Zhang, Johanne R Trippas, and Minghui Qiu. 2019. User intent prediction in information-seeking conversations. In *Proceedings of the 2019 Conference on Human Information Interaction and Retrieval*, pages 25–33.

Table 5: Ablation study results using poly-encoder as reranker. The original paper's results are presented in red. The reproducibility results are presented in blue. Numbers in bold mean the result is the best excluding oracle. † and ‡ indicates $p < 0.1$ and $p < 0.01$ statistical significance over the best of encoded text and score models.

Users	0 -tolerance			1-tolerance			2-tolerance		
Models	R@1/100	MRR	Dec. err	R@1/100	MRR	Dec. err	R@1/100	MRR	Dec. err
Encoded text	0.3919-0.0000	0.4882-0.0000	0.3236-0.0000	0.4378-0.0000	0.5284-0.0000	0.2056-0.0000	0.4611-0.0000	0.5594-0.0000	0.1631-0.0000
Score	0.4419-0.0000	0.5428-0.0000	0.2456-0.0000	0.4519-0.0000	0.5517-0.0000	0.2350-0.0000	0.4483-0.0000	0.5505-0.0000	0.2261-0.0000
Encoded text + Score	**0.4461**-0.0000	**0.5435**-0.0000	**0.2375**†-0.0000	**0.4656**‡-0.0000	**0.5620**‡-0.0000	**0.1830**‡-0.0000	**0.4788**‡-0.0000	**0.5781**‡-0.0000	**0.1519**‡-0.0000

Filip Radlinski and Nick Craswell. 2017. A theoretical framework for conversational search. In *Proceedings of the 2017 conference on conference human information interaction and retrieval*, pages 117–126.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *proceedings of the 24th ACM international on conference on information and knowledge management*, pages 553–562.

Zhenduo Wang and Qingyao Ai. 2021. Controlling the risk of conversational search via reinforcement learning. In *Proceedings of the Web Conference 2021*, pages 1968–1977.

Liu Yang, Minghui Qiu, Chen Qu, Jiafeng Guo, Yongfeng Zhang, W Bruce Croft, Jun Huang, and Haiqing Chen. 2018. Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. In *The 41st international acm sigir conference on research & development in information retrieval*, pages 245–254.