UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Learning to Ask Conversational Questions by Optimizing Levenshtein Distance

Liu, Z.; Ren, P.; Chen, Z.; Ren, Z.; de Rijke, M.; Zhou, M.

DOI
10.18653/v1/2021.acl-long.438

Publication date
2021

Document Version
Final published version

Published in
The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing

License
CC BY

Link to publication

Citation for published version (APA):
Liu, Z., Ren, P., Chen, Z., Ren, Z., de Rijke, M., & Zhou, M. (2021). Learning to Ask Conversational Questions by Optimizing Levenshtein Distance. In C. Zong, F. Xia, W. Li, & R. Navigli (Eds.), *The 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: ACL-IJCNLP 2021 : proceedings of the conference : August 1-6, 2021* (Vol. 1, pp. 5638-5650). The Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.acl-long.438

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, P.O. Box 19185, 1000 GD Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:12 Oct 2026

Learning to Ask Conversational Questions by Optimizing Levenshtein Distance

Zhongkun Liu[1], **Pengjie Ren**[1*], **Zhumin Chen**[1*], **Zhaochun Ren**[1], **Maarten de Rijke**[2], **Ming Zhou**[3]
[1]School of Computer Science and Technology, Shandong University, China
[2]University of Amsterdam & Ahold Delhaize
[3]Sinovation Ventures, China
{liuzhongkun,renpengjie,chenzhumin, zhaochun.ren}@sdu.edu.cn
m.derijke@uva.nl; mingzhou926@hotmail.com

Abstract

Conversational Question Simplification (CQS) aims to simplify self-contained questions into conversational ones by incorporating some conversational characteristics, e.g., anaphora and ellipsis. Existing maximum likelihood estimation based methods often get trapped in easily learned tokens as all tokens are treated equally during training. In this work, we introduce a Reinforcement Iterative Sequence Editing (RISE) framework that optimizes the minimum Levenshtein distance through explicit editing actions. RISE is able to pay attention to tokens that are related to conversational characteristics. To train RISE, we devise an Iterative Reinforce Training (IRT) algorithm with a Dynamic Programming based Sampling (DPS) process to improve exploration. Experimental results on two benchmark datasets show that RISE significantly outperforms state-of-the-art methods and generalizes well on unseen data.

1 Introduction

Conversational information seeking (CIS) (Zamani and Craswell, 2020; Ren et al., 2021b) has received extensive attention. It introduces a new way to connect people to information through conversations (Qu et al., 2020; Gao et al., 2021; Ren et al., 2020). One of the key features of CIS is *mixed initiative* behavior, where a system can improve user satisfaction by proactively asking clarification questions (Zhang et al., 2018; Aliannejadi et al., 2019; Xu et al., 2019), besides passively providing answers (Croft et al., 2010; Radlinski and Craswell, 2017; Lei et al., 2020).

Previous studies on asking clarification questions can be grouped into two categories: conversational question generation (Duan et al., 2017) and conversational question ranking (Aliannejadi et al., 2019).

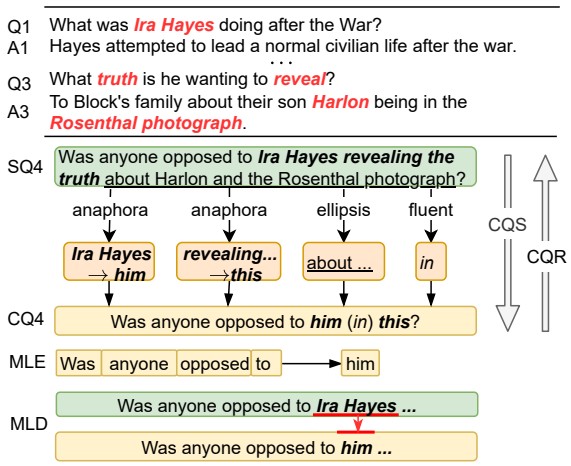

Figure 1: An example for Conversational Question Simplification and its reverse, Conversational Question Rewriting. Q1–A3 is the context, SQ4 is the self-contained question, and CQ4 is the conversational question.

The former directly generates conversational questions based on the dialogue context. However, the generated questions may be irrelevant and meaningless (Rosset et al., 2020). A lack of explicit semantic guidance makes it difficult to produce each question token from scratch while preserving relevancy and usefulness at the same time (Wang et al., 2018; Chai and Wan, 2020). Instead, the latter proposes to retrieve questions from a collection for the given dialogue context, which can usually guarantee that the questions are relevant and useful (Shen et al., 2018; Rosset et al., 2020). However, question ranking methods do not lead to a natural communication between human and machine (Pulman, 1995), as they neglect important characteristics in conversations, e.g., anaphora and ellipsis. As shown in Fig. 1, the self-contained question (SQ4) lacks these characteristics, which makes it look unnatural.

In this work, we study the task of *Conversa-*

* Corresponding authors.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 5638–5650
August 1–6, 2021. ©2021 Association for Computational Linguistics

tional Question Simplification (CQS). Given a dialogue context and self-contained question as input, CQS aims to transform the self-contained question into a conversational one by simulating conversational characteristics, such as anaphora and ellipsis. For example, in Fig. 1, four simplification operations are applied to obtain the conversational question (CQ4), which is context-dependent and superior to its origin one (SQ4) in terms of naturalness and conveying. The reverse process, i.e., Conversational Question Rewriting (CQR) (Elgohary et al., 2019; Voskarides et al., 2020) which rewrites CQ4 into SQ4, has been widely explored in the literature (Vakulenko et al., 2020; Yu et al., 2020). Although the proposed methods for CQR can be easily adopted for CQS, they do not always generate satisfactory results as they are all trained to optimize a maximum likelihood estimation (MLE) objective, which gives equal attention to generate each question token. Therefore, they often get stuck in easily learned tokens, i.e., tokens appearing in input, ignoring conversational tokens, e.g., him, which is a small but important portion of output.

To address the above issue, we propose a new scheme for CQS, namely *minimum Levenshtein distance* (MLD). It minimizes the differences between input and output, forcing the model to pay attention to contributing tokens that are related to conversational tokens, e.g., "Ira Hay" and "him" in Fig. 1. Therefore, MLD is expected to outperform MLE for CQS. However, MLD cannot be minimized by direct optimization due to the discrete nature, i.e., minimizing the number of discrete edits. We present an alternative solution, a **R**einforcement **I**terative **S**equence **E**diting (RISE) framework for the optimization of MLD.

We formulate RISE as a Hierarchical Combinatorial Markov Decision Process (HCMDP) consisting of an editing Markov Decision Process (MDP) to predict multiple edits for all tokens in the self-contained question, e.g., '**K**eep (K)' to keep a token, and a phrasing MDP to predict a phrase if the edit is '**I**nsert (I)' or '**S**ubstitute (S)'. We only have the self-contained and conversational question pairs in the dataset while the demonstrations of the editing iterations are lacked. Thus, we cannot train each editing iteration of RISE with teacher forcing. To this end, we devise an Iterative Reinforce Training (IRT) algorithm that allows RISE to do some exploration itself. The exploration can be rewarded according to its Levenshtein distance (LD) with the demonstrated conversational question. Traditional exploration methods like ϵ-sampling (Sutton and Barto, 1998) neglect the interdependency between edits for all tokens, resulting in poor exploration. Thus, we further introduce a Dynamic Programming based Sampling (DPS) process that adopts a Dynamic Programming (DP) algorithm to track and model the interdependency in IRT. Experiments on the CANARD (Elgohary et al., 2019) and CAsT (Dalton et al., 2019) datasets show that RISE significantly outperforms state-of-the-art methods and generalizes well to unseen data.

2 Conversational Question Simplification: From maximum likelihood estimation to minimum Levenshtein distance

2.1 CQS

Given a dialogue context C representing the previous conversation utterances and the self-contained clarification question candidate $x = \{x_1, \ldots, x_{|x|}\}$ to be asked next (e.g., from a conversational question ranking model), the goal of Conversational Question Simplification (CQS) is to reformulate question x to a conversational question $y = \{y_1, \ldots, y_{|y|}\}$ by simulating conversational characteristics, e.g., anaphora and ellipsis. A target conversational question $y^* = \{y_1^*, \ldots, y_{|y^*|}^*\}$ is provided during the training phase.

2.2 Maximum likelihood estimation for CQS

A commonly adopted paradigm for tasks similar to CQS, e.g., CQR, is to model the task as a conditional sequence generation process parameterized by θ, which is usually optimized by MLE:

$$\begin{aligned}\mathcal{L}_\theta &= -\log p_\theta(y^*|x, C) \\ &= -\sum_{t=1}^{|y^*|} \log p_\theta(y_t^*|y_{<t}^*, x, C),\end{aligned} \tag{1}$$

where y^* is the target question and $y_{<t}^*$ denotes the prefix $y_1^*, y_2^*, \ldots, y_{t-1}^*$. As we can see, MLE gives equal weight to each token and falls in easily learned tokens, the overwhelming duplicate tokens between x and y, while underestimating subtle differences of tokens related to conversational characteristics.

2.3 Minimum Levenshtein distance for CQS

Inspired by Arjovsky et al. (2017), to minimize the distance between two distributions, we propose

to minimize the LD between the target question y^* and the model output y so as to leverage the high overlap between x and y and focus on subtle different tokens:

$$\mathcal{L}_\theta = LD(y, y^*). \quad (2)$$

Unfortunately, it is impossible to directly optimize Eq. 2 because the LD between y and y^* is the minimum number of single-token edits (insertions, deletions or substitutions) required to change y into y^*, which is discrete and non-differentiable.

3 RISE

To optimize MLD in Eq. 2, we devise the Reinforcement Iterative Sequence Editing (RISE) framework, which reformulates the optimization of MLD as a Hierarchical Combinatorial Markov Decision Process (HCMDP). Next, we first describe our HCMDP formulation of RISE. We then detail the modeling of each ingredient in RISE. Finally, we present the training process of RISE.

3.1 HCMDP formulation for RISE

RISE produces its output y by iteratively editing x with four types of edit, i.e., 'K' to keep a token, '**D**elete (D)' to delete a token, 'I' to insert a phrase (a sequence of tokens) after a token, and 'S' to substitute a phrase by a new one. If a token is predicted as 'I' or 'S', we need to further predict a corresponding phrase. Note that we only predict one phrase for successive 'S' edits. We formulate RISE as a Hierarchical Combinatorial Markov Decision Process (HCMDP) consisting of (1) an *editing* MDP to predict multiple edits for all tokens, and (2) a *phrasing* MDP to predict a phrase if the edit is 'I' or 'S'.

The editing MDP can be formulated as a tuple $\langle \mathcal{S}^e, \mathcal{A}^e, \mathcal{T}^e, \mathcal{R}, \pi^e \rangle$. Here, $s_t^e \in \mathcal{S}^e$ denotes the question at t-th iteration y^t together with the context C, i.e., $s_t^e = (y^t, C)$. Note that $s_0^e = (x, C)$. $a_t^e = [a_{t,1}^e, a_{t,2}^e, \ldots, a_{t,|y^t|}^e] \in \mathcal{A}^e$ is a combinatorial action consisting of several interdependent edits. The number of edits corresponds to the length of y^t. For example, in Fig. 2, a_t^e = ['K', 'K', 'K', 'K', 'S', 'S', 'K', 'K']. In our case, the transition function $\mathcal{T}^e$ is deterministic, which means that the next state s_{t+1}^e is obtained by applying the predicted actions from both the editing MDP and phrasing MDP to the current state s_t^e. $r_t \in \mathcal{R}$ is the reward function, which estimates the joint effect of taking the predicted actions from both the editing and phrasing MDPs. π^e is the editing policy network.

The phrasing MDP can be formulated as a tuple $\langle \mathcal{S}^p, \mathcal{A}^p, \mathcal{T}^p, \mathcal{R}, \pi^p \rangle$. Here, $s_t^p \in \mathcal{S}^p$ consists of the current question y^t, the predicted action from the editing MDP a_t^e, and the context C, i.e., $s_t^p = (y^t, a_t^e, C)$. $a_t^p = [a_{t,1}^p, a_{t,2}^p, \ldots] \in \mathcal{A}^p$ is also a combinatorial action, where $a_{t,i}^p$ denotes a phrase from a predefined vocabulary and i corresponds to the index of the 'I' or 'S' edits, e.g., in Fig. 2, '$a_{t,1}^p$ = him' is the predicted phrase for the first 'S' edit. The length of the action sequence corresponds to the number of 'I' or 'S' edits. The transition function $\mathcal{T}^p$ returns the next state s_{t+1}^p by applying the predicted actions from the phrasing MDP to the current state s_t^p. $r_t \in \mathcal{R}$ is the shared reward function. π^p is the phrasing policy network.

RISE tries to maximize the expected reward:

$$J(\theta) = E_{a_t^e \sim \pi^e, a_t^p \sim \pi^p}[r_t], \quad (3)$$

where θ is the model parameter which is optimized with the policy gradient:

$$\nabla J(\theta) = E_{a_t^e \sim \pi^e, a_t^p \sim \pi^p}[r_t(\nabla \log \pi^e(a_t^e|s_t^e) + \nabla \log \pi^p(a_t^p|s_t^p))], \quad (4)$$

Next, we will show how to model $\pi^e(a_t^e|s_t^e)$, $\pi^p(a_t^p|s_t^p)$, and r_t.

3.2 Policy networks

We implement the editing and phrasing policy networks (π^e and π^p) based on BERT2BERT (Rothe et al., 2020) as shown in Fig. 2. The editing policy network is implemented by the encoder to predict combinatorial edits, and the phrasing policy network is implemented by the decoder to predict phrases.

3.2.1 Editing policy network

We unfold all tokens of the utterances in the context into a sequence $C = (w_1, \ldots, w_c)$, where w_i denotes a token and we add "[SEP]" to separate different utterances. Then the context and input question in t-th iteration are concatenated with "[SEP]" as the separator. Finally, we feed them into the encoder of BERT2BERT to obtain hidden representations for tokens in question $H^t = (h_1^t, \ldots, h_{|y^t|}^t)$ and apply a linear layer with parameter W^e to predict a_t^e:

$$\pi^e(a_t^e|s_t^e = (y^t, C)) = \text{softmax}(W^e H^t). \quad (5)$$

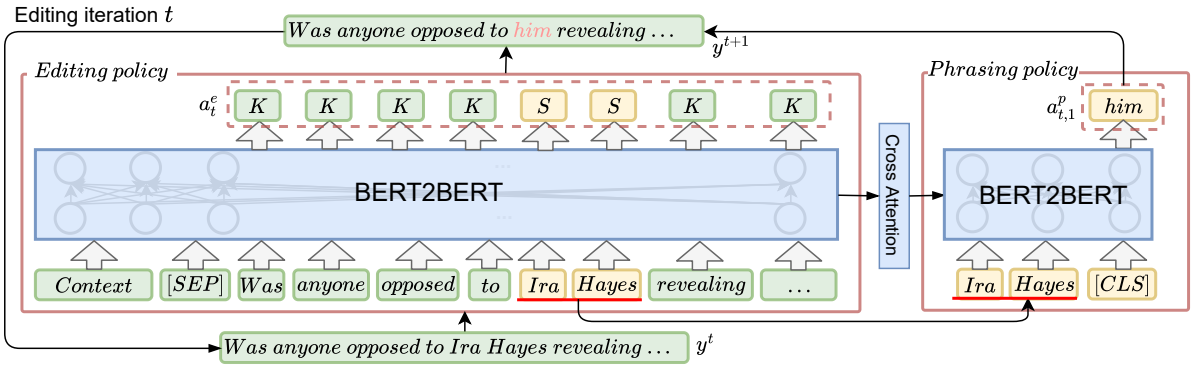

Figure 2: Architecture of our policy network. A combinatorial of all tokens edits is predicted by editing policy, and for each 'I' or 'S' edit, a phrase will be predicted by phrasing policy.

3.2.2 Phrasing policy network

We first extract the spans corresponding to the 'I' or 'S' edits from the question. If the edit is 'I', the question span $span_i^t$ consists of tokens before and after this insertion, i.e., $span_i^t = [y_j^t, y_{j+1}^t]$; if the edit is 'S', the question span $span_i^t$ consists of successive tokens corresponding to the 'S' edit, i.e., $span_i^t = [y_j^t, \ldots, y_k^t]$, where $a_{t,j:k}^e =$'S' and $a_{t,k+1}^e \neq$ 'S'. We only predict once for successive 'S' edits, e.g., in Fig. 2, the phrase 'him' is predicted to substitute question span ["Ira", "Hayes"].

For the i-th 'I' or 'S' edit with a question span $span_i^t$, we concatenate the span and "[CLS]" token as input tokens, and feed them into the decoder of BERT2BERT to obtain a hidden representation of "[CLS]" token s_i^t. We obtain S^t by concatenating each s_i^t and predict the phrases for all 'S' and 'I' edits by a linear layer with parameter W^p:

$$\pi^p(a_t^p|s_t^p) = \text{softmax}(W^p S^t). \tag{6}$$

3.3 Reward R

We devise the reward r_t to estimate the effect of taking the joint action (a_t^e, a_t^p) by encouraging actions that can result in low LD values between y^{t+1} and y^*, i.e., minimizing Eq. 2. Besides, we discourage those actions to achieve same y^{t+1} with extra non 'K' edits:

$$\begin{aligned} r_t = &\frac{1}{1+LD(y^{t+1},y^*)} \times \\ &\left(l - \sum_t (a_t^e \neq \text{'K'}) + 1\right), \\ l = &LD(y^t, y^*) - LD(y^{t+1}, y^*), \end{aligned} \tag{7}$$

where $\frac{1}{1+LD(y^{t+1},y^*)}$ will reward actions that result in low LD values between y^{t+1} and y^* and $(l - \sum_t (a_t^e \neq$ 'K'$))$ will punish those actions with unnecessary non 'K' edits.

3.4 Training

To train RISE, we need training samples in the form of a tuple $(s_t^e, a_t^e, s_t^p, a_t^p, r_t)$. However, we only have $(y^0 = x, y^*)$ in our dataset. Traditional exploration methods like ϵ-greedy sampling sample edits for all tokens independently, ignoring the interdependency between them. Instead, we devise an Iterative Reinforce Training (IRT) algorithm to sample an edit for each token by considering its future expectation, i.e., sampling $a_{t,i}^e$ based on expectation of $a_{t,:i-1}^e$ from $i = |y^t|$ to 1. We maintain a matrix M^t for this expectation based on both y^t and y^*, which is computed by a Dynamic Programming based Sampling (DPS) process due to the exponential number of edit combinations of $a_{t,:i}^e$. The details of IRT are provided in Alg. 1; it contains a DPS process that consists of two parts: computing the matrix M^t (line 4–8) and sampling actions (a_t^e, a_t^p) (line 10) based on M^t.

3.4.1 Computing the matrix M^t

Given (y^t, y^*) with length m and n, we maintain a matrix $M^t \in \mathbb{R}^{(m+1)\times(n+1)}$ (including '[SEP]', see the upper right part in Fig. 3) where each element $M_{i,j}^t$ tracks the expectation of $a_{t,:i}^e$ to convert $y_{:i}^t$ to $y_{:j}^*$:

$$\begin{aligned} M_{i,j}^t &= E_{p_{i,j}(a_{t,i}^e)}[E_{p(a_{t,:i-1}^e)}\pi_{y_{:i}^t -> y_{:j}^*}(a_{t,:i}^e)] \\ &= E_{p_{i,j}(a_{t,i}^e)} \Bigg[\pi^e(a_{t,i}^e|y^t, C) \times \\ &\quad \begin{cases} M_{i-1,j-1}^t, & \text{if } a_{t,i}^e = \text{'K'} \\ M_{i-1,j}^t, & \text{if } a_{t,i}^e = \text{'D'} \\ M_{i,j-1}^t, & \text{if } a_{t,i}^e = \text{'I'} \\ M_{i-1,j-1}^t, & \text{if } a_{t,i}^e = \text{'S'} \end{cases} \Bigg], \end{aligned} \tag{8}$$

where $a^e_{t,:i}$ is the combinational edits for tokens $y^t_{:i}$ and $\pi^e(a^e_{t,i}|y^t, C)$ is calculated by Eq. 5 (see the upper left part in Fig. 3). $M^t_{0,0}$ is initialized to 1. We will first introduce $p_{i,j}(a^e_{t,i})$ and then introduce $\pi_{y^t_{:i}->y^*_{:j}}(a^e_{t,:i})$ in Eq. 8.

Traditional sampling methods sample each edit $a^e_{t,i}$ independently, based on model likelihood $\pi^e(a^e_{t,i}|y^t, C)$. Instead, we sample each edit with probability $p_{i,j}(a^e_{t,i})$ based on edits expectation M^t, which is modeled as:

$$p_{i,j}(a^e_{t,i}) = \frac{1}{Z^t_{i,j}}\pi(a^e_{t,i}|y^t, C)\times \begin{cases} M^t_{i-1,j-1}, \text{ if } a^e_{t,i} = \text{'K'} \\ M^t_{i-1,j}, \text{ if } a^e_{t,i} = \text{'D'} \\ M^t_{i,j-1}, \text{ if } a^e_{t,i} = \text{'I'} \\ M^t_{i-1,j-1}, \text{ if } a^e_{t,i} = \text{'S'}, \end{cases} \tag{9}$$

where $Z^t_{i,j}$ is the normalization term. We give an example on computing $M^t_{1,2}$ in the bottom part of Fig. 3. For edit 'I' in $M^t_{1,2}$, its probability is 1, and its value is $\pi^e(a^e_{t,i} = \text{'I'}|y^t, C) \times M^t_{1,1} = 0.008$. For the other edits, the probability is 0. Therefore, $M^t_{1,2} = 0.008$.

$\pi_{y^t_{:i}->y^*_{:j}}(a^e_{t,:i})$ is the probability of conducting edits $a^e_{t,:i}$ to convert $y^t_{:i}$ to $y^*_{:j}$:

$$\pi_{y^t_{:i}->y^*_{:j}}(a^e_{t,:i}) = \pi^e(a^e_{t,i}|y^t, C)\times \begin{cases} \pi_{y^t_{:i-1}->y^*_{:j-1}}(a^e_{t,:i-1}), \text{ if } a^e_{t,i-1} = \text{'K'} \\ \pi_{y^t_{:i-1}->y^*_{:j}}(a^e_{t,:i-1}), \text{ if } a^e_{t,i-1} = \text{'D'} \\ \pi_{y^t_{:i}->y^*_{:j-1}}(a^e_{t,:i}), \text{ if } a^e_{t,i} = \text{'I'} \\ \pi_{y^t_{:i-1}->y^*_{:j-1}}(a^e_{t,:i-1}), \text{ if } a^e_{t,i-1} = \text{'S'}, \end{cases} \tag{10}$$

To convert $y^t_{:i}$ to $y^*_{:j}$, we need to make sure that y^t_i can convert to y^*_j and that $y^t_{:i-1}$ can convert to $y^*_{:j-1}$, which can be calculated recursively. Note that we only allow 'S' and 'D' for y^t_i when $y^t_i \neq y^*_j$ and 'K' and 'I' for y^t_i when $y^t_i = y^*_j$. And $M^t_{i-1,j-1} = E_{p(a^e_{t,:i-1})}\pi_{y^t_{:i-1}->y^*_{:j-1}}(a^e_{t,:i-1})$.

3.4.2 Sampling (a^e_t, a^p_t)

We sample (a^e_t, a^p_t) based on matrix M^t by backtracking from $i = m, j = n$. For example, as shown in the upper right in Fig. 3, we backtrack along the blue arrows. In this truncated sample, we start from $M^t_{7,6}$, sample an edit 'K' to keep 'revealing' based on $p_{7,6}(a^e_{t,7})$ in Eq. 9, and move to $M^t_{6,5}$. Then, we sample 'S' to substitute 'Ira Hayes' to 'him' and move to $M^t_{4,4}$. Finally, we sample 'K' in $[M^t_{4,4}, M^t_{3,3}, M^t_{2,2}M^t_{1,1}, M^t_{0,0}]$ to keep ['to', 'opposed', 'anyone', 'Was', '[SEP]']. Therefore, we can obtain a^e_t = [K, K, K, K, K, S, S, K], a^p_t = ['him']. Note that we obtain a^p_t by merging all corresponding tokens y^*_j as the phrase for each 'I' edit and successive 'S' edits and we only substitute once. The backtracking rule can be formulated as:

$$M^t_{i,j} \rightarrow \begin{cases} M^t_{i-1,j-1}, \text{if } a^e_{t,i} \in [\text{'K'}, \text{'S'}] \\ M^t_{i-1,j}, \text{if } a^e_{t,i} = \text{'D'} \\ M^t_{i,j-1}, \text{if } a^e_{t,i} = \text{'I'}. \end{cases} \tag{11}$$

Algorithm 1: Training Process of RISE

Input: The origin data $\mathcal{D} = \{(x, y^*)\}$, the number of samples L;
Output: The model parameters θ;

while *not coverage* **do**
  Sample (y^t, y^*) from $\mathcal{D}$;
  $M^t_{0,0} = 1$;
  for *i in* 0,..., m **do**
    for *j in* 0,..., n **do**
      Compute $M^t_{i,j}$ according to Eq. 8;
    end
  end
  Sample a^e_t, a^p_t according to Eq. 11 ;
  Apply a^e_t, a^p_t to obtain y^{t+1} ;
  Obtain r_t according to Eq. 7 ;
  Update θ according to Eq. 4 ;
  Add (y^{t+1}, y^*) to D.
end

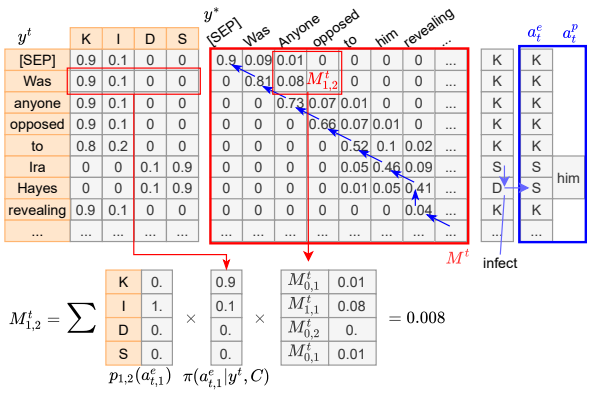

Figure 3: The DPS process consists of computing matrix M (red box) and sampling (a^e_t, a^t_p) (blue arrows and box).

3.5 Inference

During inference, RISE iteratively edits x until it predicts 'K' edits for all tokens or it achieves the

maximum iteration limit. For example, for editing iteration t in Figure 2, it predicts 'S' for 'Ira' and 'Hayes' to substitute it to 'him' and 'K' for other tokens, which results in 'Was anyone opposed to him revealing . . . ' as output. The output in iteration t is the input of iteration $t + 1$. The actual editing iteration times vary with different samples.

4 Experiments

4.1 Datasets

As with previous studies (Elgohary et al., 2019; Yu et al., 2020; Vakulenko et al., 2020; Lin et al., 2020a), we conduct experiments on the CANARD[1] (Elgohary et al., 2019) dataset, which is a large open-domain dataset for conversational question answering (with over 30k training samples). Each sample in the CANARD dataset includes a conversational context (historical questions and answers), an self-contained question, and its corresponding conversational question under the context. The questions always have clear answers, e.g., 'Did he win the lawsuit?' We follow the CANARD splits for training and evaluation.

In addition, we evaluate the model performance on the CAsT[2] dataset (Dalton et al., 2019), which is built for conversational search. Different from CANARD, its context only contains questions without corresponding answers. Besides, most questions in the CAsT dataset are exploring questions to explore relevant information, e.g., 'What about for great whites?' Since the CAsT dataset only contains 479 samples from different domains compared to CANARD, we use it for testing.

4.2 Evaluation metrics

Following Su et al. (2019); Xu et al. (2020), we use BLEU-1, BLEU-2, BLEU-3, BLEU-4 (Papineni et al., 2002), ROUGE-L (Lin, 2004), and CIDEr (Vedantam et al., 2015) for automatic evaluation. BLEU-n and ROUGE-L measure the word overlap between the generated and golden questions. CIDEr measures the extent to which important information is missing. Elgohary et al. (2019); Lin et al. (2020a); Xu et al. (2020) have shown that automatic evaluation has a high correlation with human judgement on this task, so we do not conduct human evaluation in this paper.

[1] http://canard.qanta.org
[2] http://www.treccast.ai

4.3 Baselines

We compare with several recent state-of-the-art methods for this task or closely related tasks:

- **Origin** uses the original self-contained question as output.
- **Rule** (Yu et al., 2020) employs two simple rules to mimic two conversational characteristics: anaphora and ellipsis.
- **QGDiv** (Sultan et al., 2020) uses RoBERTa (Liu et al., 2019) with beam search (Wiseman and Rush, 2016) for generation.
- **Trans++** (Vakulenko et al., 2020) predicts several word distributions, and combines them to obtain the final word distribution when generating each token.
- **QuerySim** (Yu et al., 2020) adopts a GPT-2 (Radford et al., 2019) model to generate conversational question.

We also found some methods from related tasks. But they do not work on this task for various reasons. For example, due to the lack of labels needed for training, we cannot compare with the methods proposed by Rosset et al. (2020) and Xu et al. (2020). Su et al. (2019) propose a model that can only copy tokens from input; it works well on the reverse task (i.e., CQR), but not on CQS.

4.4 Implementation details

We use BERT2BERT for the modeling of the editing and phrasing parts (Rothe et al., 2020), as other pretrained models like GPT-2 (Radford et al., 2019) cannot work for both. The hidden size is 768 and phrase vocabulary is 3461 following (Malmi et al., 2019). We use the BERT vocabulary (30,522 tokens) for all BERT-based or BERT2BERT-based models. We use the Adam optimizer (learning rate 5e-5) (Kingma and Ba, 2015) to train all models. In particular, we train all models for 20,000 warm-up steps, 5 epochs with pretrained model parameters frozen, and 20 epochs for all parameters. For RISE, the maximum editing iteration times is set to 3. We use gradient clipping with a maximum gradient norm of 1.0. We select the best models based on the performance on the validation set. During inference, we use greedy decoding for all models.

4.5 Results

We list the results of all methods on both CANARD and CAsT in Table 1. From the results, we have two main observations.

First, RISE significantly outperforms all base-

Table 1: Overall performance (%) on CANARD and CAsT. **Bold face** indicates the best results in terms of the corresponding metrics. Significant improvements over the best baseline results are marked with * (t-test, $p < 0.01$). Note that we denote BLEU-n as B-n and ROUGE-L as R-L.

	CANARD (%)						CAsT (%) (unseen)					
Method	**B-1**	**B-2**	**B-3**	**B-4**	**R-L**	**CIDEr**	**B-1**	**B-2**	**B-3**	**B-4**	**R-L**	**CIDEr**
Origin	54.7	47.0	40.6	35.3	70.9	3.460	75.9	69.2	62.9	57.6	85.0	5.946
Rule	55.0	47.0	40.2	34.8	70.5	3.420	78.0	71.4	65.3	60.0	86.1	6.220
Trans++	84.3	77.5	72.1	67.5	84.6	6.348	76.0	64.3	54.8	47.2	76.5	4.258
QGDiv	85.2	78.6	73.3	68.9	85.2	6.469	75.9	65.3	56.7	59.6	78.0	4.694
QuerySim	83.1	78.5	74.5	71.0	82.7	6.585	80.6	75.3	70.2	65.5	83.3	6.345
RISE	**86.3***	**80.5***	**75.6**	**71.6***	**86.2***	**6.759**	**85.1***	**78.4**	**72.2**	**66.8**	**87.8***	**6.543**

lines on both datasets. Specifically, RISE outperforms the strongest baseline QuerySim by ˜4% in terms of ROUGE-L. The reason is that RISE enhanced by DPS has a better ability to emphasize conversational tokens, rather than treating all tokens equally.

Second, RISE is more robust, which generalizes better to unseen data of CAsT. The results of the neural methods on CANARD are much better than those on CAsT. But, RISE is more stable than the other neural models. For example, RISE outperforms QuerySim by 0.6% in BLEU-4 on CANARD, while 1.3% on CAsT. The reason is that RISE learns to cope with conversational tokens only, while other models need to generate each token from scratch.

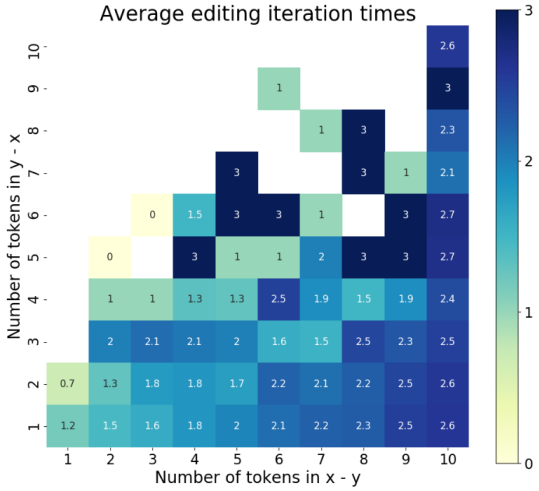

Figure 4: Average number of editing iteration of RISE conditioned on number of tokens in x - y and y - x.

5 Analysis

5.1 Ablation study

To analyze where the improvements of RISE come from, we conduct an ablation study on the CANARD and CAsT datasets (see Table 2). We consider two settings:

- **-DPS.** Here, we replace DPS by ϵ-greedy sampling ($\epsilon = 0.2$) (Sutton and Barto, 1998).
- **-MLD.** Here, we replace MLD by MLE in RISE.

The results show that both parts (DPS and MLD) are helpful to RISE as removing either of them leads to a decrease in performance. Without MLD, the performance drops a lot in terms of all metrics, e.g., 3% and 7% in BLEU-4 on CANARD and CAsT, respectively. This indicates that optimizing MLD is more effective than optimizing MLE. Besides, MLD generalizes better on unseen CAsT as it drops slightly in all metrics, while with MLE, we see a drop of 10% in BLEU-1.

Without DPS, the results drop dramatically, which indicates that DPS can do better exploration than ϵ-greedy and is of vital importance for RISE. For example, -DPS tends to sample more non 'K' edits (RISE vs -DPS: 10% vs 22% on CANARD), which is redundant and fragile. The performance of -DPS is even worse than Origin in CAsT in BLEU-4. This may be because CAsT is unseen.

5.2 Editing iterations

To analyze the relation between the number of editing iterations of RISE and the editing difficulty, we plot a heatmap in Fig. 4, where the deeper color represents a larger number of editing iterations. The x-axis denotes the number of tokens shown in input x but not shown in output y and the y-axis denotes the number of tokens shown in y but not in x.

As the number of different tokens between x and y increases, the number of editing iterations increases too. For example, when the y-axis is 1, as the x-axis ranges from 1 to 10, the number of

Table 2: Ablation study (%) on CANARD and CAsT.

	CANARD (%)						CAsT (%) (unseen)					
Method	**B-1**	**B-2**	**B-3**	**B-4**	**R-L**	**CIDEr**	**B-1**	**B-2**	**B-3**	**B-4**	**R-L**	**CIDEr**
Origin	54.7	47.0	40.6	35.3	70.9	3.460	75.9	69.2	62.9	57.6	85.0	5.946
-DPS	67.5	56.4	47.3	39.9	73.9	3.743	80.9	70.0	60.6	53.3	81.2	4.713
-MLD	85.2	78.6	73.3	68.9	85.2	6.469	75.9	65.3	56.7	59.6	78.0	4.694
RISE	**86.3**	**80.5***	**75.6***	**71.6***	**86.2***	**6.759***	**85.1***	**78.4***	**72.2***	**66.8***	**87.8***	**6.543***

editing iterations increases from 1.2 to 2.6 because more 'D' edits are needed. We also found that when the x-axis is between 3 and 7 and the y-axis is between 1 and 4, only 1–2 editing iterations are needed. Usually, this is because RISE only needs 1 or 2 successive 'S' edits for simulating anaphora.

5.3 Influence of the number of editing iterations

The overall performance of RISE improves as the number of editing iterations increases. RISE achieves 70.5% in BLEU-4 in the first iteration (even worse than QuerySim in Table 1) but 71.5% and 71.6% in the second and third iterations. This shows that some samples are indeed more difficult to be directly edited into conversational ones, and thus need more editing iterations.

Even though it will not hurt the performance a lot, more editing iterations are not always helpful. About 5% of the samples achieve worse BLEU-4 scores as the number of editing iterations increases. For example, RISE edits 'where did humphrey lyttelton go to school at?' into 'where did he go to school at?' in the first iteration, which is perfect. But RISE continues to edit it into 'where did he go to school?' in the second iteration, which is undesirable. This is because RISE fails to decide whether to stop or continue editing.

5.4 Case Study

In Table 3 we present two examples of the output of RISE. We present the context, the original self-contained question, the target conversational question, and the output of RISE in the n-th iteration, denoted as 'Context', 'Question', 'Target' and 'Rewrite#n', respectively. We have two main observations. First, it is helpful to edit iteratively. As shown in Example 1, RISE first replaces 'Abu' as 'he' in the first iteration and then deletes 'bakr' in the second iteration, which simulates anaphora by editing twice. In Example 2, RISE simulates ellipsis by deleting multiple words and achieves poor grammar after the first iteration but corrects this by deleting some of the leftover words. RISE may have learned to check the grammar and remove redundant words.

Table 3: Examples generated by RISE on CANARD. Here, 'Question' means the self-contained question, and 'Target' means the desired conversational question. 'Rewrite#n' denotes the output of RISE in n-th iteration.

Example 1	1. At Tabuk the standard of the army was entrusted to Abu Bakr.
Context	2. Where was Tabuk located? 3. Tabuk on the Syrian border.
Question	What did Abu Bakr do during the expedition of Tabuk?
Rewrite#1	What did he bakr do during expedition?
Rewrite#2	What did he do during expedition?
Target	What did abu bakr do during the expedition?
Example 2	1. When did Clift start his film career?
Context	2. His first movie role was opposite John Wayne in Red River, which was shot in 1946 and released in 1948.
Question	Did Montgomery Clift win any awards for any of his films?
Rewrite#1	Did he win any awards for and?
Rewrite#2	Did he win any awards?
Target	Did he win any awards for any of his films?

Second, RISE can simulate more conversational characteristics than human, and sometimes it can achieve a better result, sometimes not. As we can see, RISE results a better conversational question by additionally simulating anaphora for 'Abu Bakr' in Example 1. However, RISE leaves out necessary information in Example 2. Here, RISE tries to simulate conversational characteristics as much as

possible, where the result may be uncontrollable. In future work, we will add a discriminator to check the necessary information.

6 Related work

Studies on asking conversational question can be divided into two categories: *conversational question generation* and *conversational question ranking*.

Conversational question generation aims to directly generate conversational questions conditioned on the dialogue context (Sultan et al., 2020; Ren et al., 2021a). Zamani et al. (2020) and Qi et al. (2020) define a question utility function to guide the generation of conversational questions. Nakanishi et al. (2019); Jia et al. (2020) incorporate knowledge with auxiliary tasks. These methods may generate irrelevant questions due to their pure generation nature.

Conversational question ranking (Aliannejadi et al., 2019) retrieves questions from a collection based on the given context, so the questions are mostly relevant to the context. Kundu et al. (2020) propose a pair-wise matching network between context and question to do question ranking. Some studies also use auxiliary tasks to improve ranking performance, such as Natural Language Inference (Kumar et al., 2020) and relevance classification (Rosset et al., 2020). The retrieved questions are often unnatural without considering the conversational characteristics, e.g., anaphora and ellipsis.

CQS rewrites the retrieved self-contained questions into conversational ones by incorporating the conversational characteristics. Existing applicable methods for CQS are all MLE based (Xu et al., 2020; Yu et al., 2020; Lin et al., 2020b; Vakulenko et al., 2020), which often get stuck in easily learned tokens as each token is treated equally by MLE. Instead, we propose a MLD based RISE framework to formulate CQS as a HCMDP, which is able to discriminate different tokens through explicit editing actions, so that it can learn to emphasize the conversational tokens and generate more natural and appropriate questions.

7 Conclusion

In this paper, we have proposed a minimum Levenshtein distance (MLD) based Reinforcement Iterative Sequence Editing (RISE) framework for Conversational Question Simplification (CQS). To train RISE, we have devised an Iterative Reinforce Training (IRT) algorithm with a novel Dynamic Programming based Sampling (DPS) process. Extensive experiments show that RISE is more effective and robust than several state-of-the-art CQS methods. A limitation of RISE is that it may fail to decide whether to stop or continue editing and leave out necessary information. In future work, we plan to address this issue by learning a reward function that considers the whole editing process through adversarial learning (Goodfellow et al., 2014).

Code

To facilitate the reproducibility of the results, we share the codes of all methods at `https://github.com/LZKSKY/CaSE_RISE`.

Acknowledgments

We thank the reviewers for their valuable feedback. This research was partially supported by the National Key R&D Program of China with grant No. 2020YFB1406704, the Natural Science Foundation of China (61972234, 61902219, 62072279), the Key Scientific and Technological Innovation Program of Shandong Province (2019JZZY010129), the Tencent WeChat Rhino-Bird Focused Research Program (JR-WXG-2021411), the Fundamental Research Funds of Shandong University, and the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, `https://hybrid-intelligence-centre.nl`.

All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

References

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, SIGIR 2019*, pages 475–484.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Zi Chai and Xiaojun Wan. 2020. Learning to ask more: Semi-autoregressive sequential question generation under dual-graph interaction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 225–237.

W Bruce Croft, Donald Metzler, and Trevor Strohman. 2010. *Search engines: Information retrieval in practice*, volume 520. Addison-Wesley Reading.

Jeffrey Dalton, Chenyan Xiong, and Jamie Callan. 2019. Cast 2019: The conversational assistance track overview. In *Proceedings of the 28th Text REtrieval Conference, TREC 2019*, pages 13–15.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017*, pages 866–874.

Ahmed Elgohary, Denis Peskov, and Jordan L. Boyd-Graber. 2019. Can you unpack that? Learning to rewrite questions-in-context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 5917–5923.

Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and challenges in conversational recommender systems: A survey. *arXiv preprint arXiv:2101.09459*.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv arxiv arXiv:1406.2661*.

Xin Jia, Wenjie Zhou, Xu Sun, and Yunfang Wu. 2020. How to ask good questions? Try to leverage paraphrases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 6130–6140.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*.

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 and Knowledge Management, CIKM 2020*, pages 2093–2096.

Souvik Kundu, Qian Lin, and Hwee Tou Ng. 2020. Learning to identify follow-up questions in conversational question answering. In *Proceedings of the 58th Conference of the Association for Computational Linguistics, ACL 2020*, pages 959–968.

Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining, WSDM 2020*, pages 304–312.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, ACL 2002*, pages 74–81.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020a. Conversational question reformulation via sequence-to-sequence architectures and pretrained language models. *arXiv preprint arXiv:2004.01909*.

Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. 2020b. Query reformulation using query history for passage retrieval in conversational search. *arXiv preprint arXiv:2005.02230*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 5053–5064.

Mao Nakanishi, Tetsunori Kobayashi, and Yoshihiko Hayashi. 2019. Towards answer-unaware conversational question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering, MRQA@EMNLP 2019*, pages 63–71.

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 2002*, pages 311–318.

Stephen G Pulman. 1995. Anaphora and ellipsis in artificial languages. *Natural Language Engineering*, 1(3):217–234.

Peng Qi, Yuhao Zhang, and Christopher D. Manning. 2020. Stay hungry, stay focused: Generating informative and specific questions in information-seeking conversations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 25–40.

Chen Qu, Liu Yang, Cen Chen, Minghui Qiu, W. Bruce Croft, and Mohit Iyyer. 2020. Open-retrieval conversational question answering. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020*, pages 539–548.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

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, CHIIR 2017*, pages 117–126.

Pengjie Ren, Zhumin Chen, Christof Monz, Jun Ma, and Maarten de Rijke. 2020. Thinking globally, acting locally: Distantly supervised global-to-local knowledge selection for background based conversation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI*, pages 8697–8704.

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Evangelos Kanoulas, Christof Monz, and Maarten de Rijke. 2021a. Conversations with search engines: Serp-based conversational response generation. *ACM Transactions on Information Systems (TOIS), 2021*.

Pengjie Ren, Zhongkun Liu, Xiaomeng Song, Hongtao Tian, Zhumin Chen, Zhaochun Ren, and Maarten de Rijke. 2021b. Wizard of search engine: Access to information through conversations with search engines. In *Proceedings of the 44rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2021*.

Corbin Rosset, Chenyan Xiong, Xia Song, Daniel Campos, Nick Craswell, Saurabh Tiwary, and Paul N. Bennett. 2020. Leading conversational search by suggesting useful questions. In *Proceedings of the Web Conference, WWW 2020*, pages 1160–1170.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2020. Leveraging pre-trained checkpoints for sequence generation tasks. *Trans. Assoc. Comput. Linguistics*, 8:264–280.

Ying Shen, Yang Deng, Min Yang, Yaliang Li, Nan Du, Wei Fan, and Kai Lei. 2018. Knowledge-aware attentive neural network for ranking question answer pairs. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020*, pages 901–904.

Hui Su, Xiaoyu Shen, Rongzhi Zhang, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance rewriter. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 22–31.

Md. Arafat Sultan, Shubham Chandel, Ramón Fernandez Astudillo, and Vittorio Castelli. 2020. On the importance of diversity in question generation for QA. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 5651–5656.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Svitlana Vakulenko, Shayne Longpre, Zhucheng Tu, and Raviteja Anantha. 2020. Question rewriting for conversational question answering. *arXiv preprint arXiv:2004.14652*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, CVPR*, pages 4566–4575.

Nikos Voskarides, Dan Li, Pengjie Ren, Evangelos Kanoulas, and Maarten de Rijke. 2020. Query resolution for conversational search with limited supervision. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 921–930.

Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018. Learning to ask questions in open-domain conversational systems with typed decoders. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*, pages 2193–2203.

Sam Wiseman and Alexander M. Rush. 2016. Sequence-to-sequence learning as beam-search optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016*, pages 1296–1306.

Jingjing Xu, Yuechen Wang, Duyu Tang, Nan Duan, Pengcheng Yang, Qi Zeng, Ming Zhou, and Xu Sun. 2019. Asking clarification questions in knowledge-based question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*, pages 1618–1629.

Kun Xu, Haochen Tan, Linfeng Song, Han Wu, Haisong Zhang, Linqi Song, and Dong Yu. 2020. Semantic role labeling guided multi-turn dialogue rewriter. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020*, pages 6632–6639.

Shi Yu, Jiahua Liu, Jingqin Yang, Chenyan Xiong, Paul N. Bennett, Jianfeng Gao, and Zhiyuan Liu. 2020. Few-shot generative conversational query rewriting. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, SIGIR 2020*, pages 1933–1936.

Hamed Zamani and Nick Craswell. 2020. Macaw: An extensible conversational information seeking platform. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2020*, pages 2193–2196.

Hamed Zamani, Susan T. Dumais, Nick Craswell, Paul N. Bennett, and Gord Lueck. 2020. Generating clarifying questions for information retrieval. In *Proceedings of the Web Conference 2020, WWW 2020*, pages 418–428.

Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational

search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018*, pages 177–186.

Appendix

For reproducibility for all reported experimental results, we report the following information. The average running time for RISE, QuerySim, Trans++, QGDiv, -MLD, -DPS are 15 hours, 5 hours, 9.5 hours, 9 hours, 9 hours, 15 hours, respectively. The number of parameters in RISE, Trans++, QGDiv, -MLD, -DPS are 221M and the number of parameters in QuerySim is 125M. We list the validation performance on CANARD in Table. 4, as only CANARD is used for validation. As we can see, it has high correlation to test performance on CANARD. We use this script [3] for evaluation.

Table 4: Overall performance (%) on validation set of CANARD. Note that we denote BLEU-n as B-n and ROUGE-L as R-L.

	CANARD (%)					
Method	**B-1**	**B-2**	**B-3**	**B-4**	**R-L**	**CIDEr**
Trans++	86.5	80.3	75.4	71.3	86.2	6.704
QGDiv	87.0	80.9	75.9	61.8	86.8	6.786
QuerySim	83.9	79.7	75.9	72.5	83.2	6.737
-DPS	67.2	55.9	46.8	39.4	74.3	3.745
-MLD	87.0	80.9	75.9	61.8	86.8	6.786
RISE	88.0	82.6	78.3	74.6	87.5	7.050

For reproducibility for experiments with hyperparameter search, we report the following information. The hyperparameter for RISE is the max editing iteration times. We search it in range of 1 to 5 and find 3 can perform best on BLEU-4. The results in range of 1 to 5 on BLEU-4 are 70.5%, 71.5%, 71.6%, 71.6% and 71.6%, respectively.

[3] https://github.com/Maluuba/nlg-eval