Information Retrieval Journal (2021) 24:322–346
https://doi.org/10.1007/s10791-021-09395-3

Strong natural language query generation

Binsheng Liu[1] · **Xiaolu Lu**[2] · **J. Shane Culpepper**[1]

Received: 24 June 2020 / Accepted: 2 July 2021 / Published online: 15 July 2021

Abstract
In this paper, we propose a novel query generation task we refer to as the Strong Natural Language Query (SNLQ) problem. The key idea we explore is how to best learn document summarization and ranker effectiveness jointly in order to generate human-readable queries which capture the information need conveyed by a document, and that can also be used for refinding tasks and query rewriting. Our problem is closely related to two well-known retrieval problems—known-item finding and strong query generation—with the additional objective of maximizing query informativeness. In order to achieve this goal, we combine state-of-the-art abstractive summarization techniques and reinforcement learning. We have empirically compared our new approaches with several closely related baselines using the MS-MARCO data collection, and show that the approach is capable of achieving substantially better trade-off between effectiveness and human-readability than have been reported previously.

Keywords Known-item finding · Generative natural language models · Retrieval performance trade-offs

1 Introduction

Effective query (re-)formulation is a fundamental research problem which has been studied extensively in Information Retrieval (IR) for several decades (Callan and Connell 2001; Lee and Croft 2012; Azzopardi et al. 2007; Dang and Croft 2010; Xue and Croft 2013; Bendersky et al. 2012; He et al. 2016; Belkin et al. 1993; Bailey et al. 2017). While many users are surprisingly good at formulating short keyword queries to find relevant documents that satisfy their information needs, automated methods capable of generating similarly "good" queries remain an elusive research goal. A "good" query should be readable,

✉ J. Shane Culpepper
shane.culpepper@rmit.edu.au

Binsheng Liu
binsheng.liu@rmit.edu.au

Xiaolu Lu
xiaolu.lu@microsoft.com

[1] RMIT University, Melbourne, Australia

[2] Microsoft, Melbourne, Australia

capture core elements of the underlying information need, and be effective (find relevant information in a target document collection—a collection which the user may have little or no information about). From a user's perspective, the query should also be short, informative, and effective. That is, the user can easily rationalize word choices and intent of their query re-formulations (or suggestions) in an information seeking session in order to achieve the best outcomes, and finding methods that are capable of mimicking this behavior can be used to improve interactions between humans and machines.

This type of "good" query, also known as a *transparent query* (Muramatsu and Pratt 2001), are easy to understand, and positively influence retrieval performance for the users (Koenemann and Belkin 1996; Thomas et al. 2019; Muramatsu and Pratt 2001). As noted by Muramatsu and Pratt (2001), when a search system provides a conspicuous query rewriting process, users have clues on how best to reformulate their own queries rather than trying to guess what a system is doing surreptitiously. While previous work (Muramatsu and Pratt 2001) describes how such a transformation process might be operationalized, recent transformer-based models have shown substantial improvements when applied to human text generation and query effectiveness (Dai and Callan 2019). Transformers are now readily available that contain deep linguistic capabilities (e.g. co-references) (Clark et al. 2019) which were pre-trained using web-scale text collections.

In this work, we revisit the problem of automatic query formulation from a related angle—document summarization. Remarkable progress has been made in the NLP community in text summarization and Natural Language Generation (NLG) in the last five years, albeit with different goals. More specifically, these new approaches generate human readable text from documents that capture the most salient points of the target document. However, the summary may not be an effective query to find the original document or other similar but relevant documents in the collection. In IR, generating queries to find a specific document is called the *known-item finding problem* (Ogilvie and Callan 2003; Azzopardi and de Rijke 2006), or more formally as the *strong query problem* (Kumar et al. 2011). A strong query by definition is a query that uniquely identifies a document, and was originally explored for the rank aggregation problem. Such queries are highly effective for the document refinding task, and provide a theoretical model that can also be adapted to model effectiveness for our key task—generating effective queries for a collection. However, strong queries heavily favor rare terms and often have no conceptual overlap with the user's information need. So, a variation on this problem is the focus in this work. The key idea is to impose additional linguistic constraints to favor query formulations that are more human readable.

A variation of this problem is the focus in this work, where we impose additional constraints that also capture important natural language properties and improve system performance. The most effective solutions for this problem are commonly biased towards rare terms in the queries generated, which are rarely representative of human generated queries, are not accurate summaries of the document, and often not easily understandable. That is, they are effective but not informative. Our goal is to generate strong queries which are readable and informative. If a generated document summary accurately captures the information need(s) satisfied by the document and is also a strong query, dramatic improvements in query suggestion / reformulation tools would be possible in the future, which have historically used user query logs and click graphs to make suggestions, rather than statistical properties of the documents relative to the entire collection (Cai and de Rijke 2016). While not explored in the original work of Kumar et al., the strong query problem can be easily extended to maximize effectiveness when more than one relevant document exists. In such a scenario, the model is conditioned using multiple query-document pairs.

1.1 Problem formulation

Strong queries for a document are the shortest queries that rank a target document at the topmost position (Kumar et al. 2011). Strong queries can play an important role in understanding the performance of retrieval models, and can be used for a variety of related tasks, such as plagiarism detection. We extend this important problem by adding two additional constraints: (1) the queries should be informative; (2) the queries should not use esoteric terms (be easy to understand). We will refer to this problem the *strong natural language query* (SNLQ) problem.

Let $\mathscr{M}$ be a retrieval model and x be a target document. Formally, the problem studied in this paper is to construct a model to generate queries from a given document x: $x \mapsto y$, where y is a generated query such that: (1) $\mathscr{M}(y, x)$ ranks x at the topmost position (2) y should be readable and (3) y should also be informative to the user. Note that, our problem does not necessarily require the shortest query as there is a tension between retrieval effectiveness and quality of the summary, and short queries have the additional benefit of being more efficient to process by a search engine.

Contributions

- We propose a novel query generation task—SNLQ. This task has the dual objective of maximizing readability, as is common in natural language processing, and retrieval effectiveness, which is often the main objective in information retrieval systems. That is, the task improves transparency when reformulating a user query but also ensures that the queries are effective when used for retrieval by the underlying search engine.
- Our solution combines transformer-based abstractive summarization techniques and reinforcement learning over multiple objectives—summarization, informativeness, and retrieval effectiveness.
- Experiments to empirically validate the quality of our new approach using the MS-MARCO dataset show that we are able to leverage models learned from an abstractive summarizer to generate effective and readable queries for any document in the collection. We also consider how best to evaluate the quality of such queries, which can be very difficult to do in an automated way.

2 Related work

There are four lines of work which are most closely related to our own: known-item finding, query generation, document summarization and interpretability of query reformulation. In this section, we discuss the connections between our work and the existing literature.

2.1 Known-item finding

Known-Item finding (or re-finding) is the task of re-finding a previously seen item. This is a common behavior exhibited by frequent users of commercial web search engines (Teevan 2007). These observations motivated the work of Hauff et al. (2012) who wished to explore this behavior further in a laboratory environment, which is often difficult given the lack of personalized search logs for privacy reasons. In order to construct their test collection, crowdsourcing and automatic query generation approaches were combined to

simulate user behavior. The automatic generation framework used in this work was initially proposed by Azzopardi and de Rijke (2006) and consisted three sampling-based methods: popular (POP), discriminative (DIS) and uniform, which simulated the process of human generated re-finding queries with "false memory". Follow-up work by Azzopardi et al. (2007) applied these models on six different languages to further validate the effectiveness of these approaches. This work differs from the SNLQ problem in two ways: (1) there were no guarantees of query informativeness, and (2) their goal was not to generate a *strong query* which ranks the target document at the topmost position.

Improving search effectiveness of the known-item finding task has also been explored in past work. The most relevant to our work is the strong query problem as proposed by Kumar et al. (2011). A *strong query* is defined as the shortest query which ranks a document at the highest position. The task of strong query generation was reduced to the well-known NP-HARD set-cover problem, which motivated their greedy algorithm solution, as is common for such problems. Kumar et al. concluded that a query length of less than four words is on average sufficient to induce a strong query, but no restrictions were placed on term scarcity.

2.2 Query generation

The task of query generation is commonly explored from two angles: (1) given a query and the goal is to generate the alternatives (Sheldon et al. 2011; Liu et al. 2019; Sordoni et al. 2015; Bailey et al. 2016; Ahmad et al. 2019; Boldi et al. 2008; Huang et al. 2003; Wu et al. 2018; Jiang and Wang 2018); or (2) given one or more documents, generate queries that are more effective (Lavrenko and Croft 2001; Lee and Croft 2012; Nogueira et al. 2019; Nogueira and Lin 2019).

There are several approaches commonly used for query suggestion. For example, query log mining (Boldi et al. 2008; Sheldon et al. 2011; Liu et al. 2019) or crowdsourcing (Bailey et al. 2016). Automatic query reformulation has also received a great deal of attention in previous work (Huang et al. 2003; Wu et al. 2018; Sordoni et al. 2015; Ahmad et al. 2019; Jiang and Wang 2018), and often rely on user interaction data and deep learning. For example, the approaches of Sordoni et al. (2015) and Ahmad et al. (2019) explored query reformation as a generative task for session-based retrieval, and exploited user interaction information to iteratively improve quality. The user interaction data (click information) required by both approaches, and necessary accurately model contextual information. We approach our problem in a different way: (i) we do not include user interaction information, (ii) our goal is not to predict the "next query", but to generate queries for any document in a collection. Our setting, which is to generate queries from a given document, can also be linked to the large body of work on pseudo-relevance feedback (PRF) (Lavrenko and Croft 2001). The work of Lee and Croft (2012) is closely related to our problem, where the problem explored was query generation on text passages selected by a user. They rely on phrase-based extractive summarization. Although this work is clearly related, there are a few notable differences: (1) we do not use an extractive technique; (2) we also aim to generate a natural language query, instead of only extracting a phrase directly from the original document; and (3) our problem setting also explores the importance of informativeness. Indeed, extractive summarization techniques could easily be used within our framework, but not explored here.

Query generation can also be cast as a machine translation (MT) problem (Berger and Lafferty 1999) where a document is "translated" to "query" in order to resolve vocabulary mismatching. The "Doc2query" technique (Nogueira et al. 2019) uses a machine

translation model to generate candidate queries based on a given document. However, instead of regarding the generated queries as new reformulations of the original user query, the queries are used to enrich the target document. This is a common technique used to learn question answering models as the overlap in terms found in the question and the answer may be minimal. Nogueira et al. showed that their approach dramatically reduced vocabulary mismatches in the MS-MARCO passage retrieval task and showed that the model could be improved further in follow-on work by applying an even more sophisticated language model to the MT task (Nogueira and Lin 2019; Raffel et al. 2020). This work is relevant to our problem setting in two ways: (1) Our goal is also query generation from a targeted document; and (2) the generated queries should have properties similar to natural language text. Nogueira et al. do not explicitly optimize for readability or informativeness, but the quality of the queries generated are consistently good when manually inspected by humans, and hence are used as a baseline in our work.

2.3 Abstractive summarization

Text summarization is the process of automatically condensing natural language text into another succinct but semantically correct text representation without losing any of the key information from the original. Current NLP approaches to this problem are generally categorized as being abstractive or extractive. In this paper, we focus primarily on abstractive summarization as this provides greater flexibility in the generated text and the learned language model. Our primary goal is to automatically discover alternative phrases for query rewriting as well as finding relevant documents that rely on alternative word choices. Recent advances in abstractive summarization models, and more generally NLG models learned with attention-based transformers are now comparable to humans (see for example Nallapati et al. 2016; Cohan et al. 2018; Dong et al. 2019).

A related issue which has historically plagued NLG tasks is repeated phrase generation (Paulus et al. 2018; See et al. 2017). This can now be mitigated by using pointer-generation (See et al. 2017) or intra-temporal attention mechanisms (Paulus et al. 2018). Our work relies heavily on these and related abstractive summarization techniques, but with an additional optimization goal – the generated summary of the document should also be an effective query that can be used to easily refind the document in the collection.

2.4 Interpretability of query reformulation

Interpretability has recently become a focus in the machine learning field, and has also been explored in IR as search engines often apply ranking models in multiple stages of the retrieval process. Several researchers have shown that improving the transparency and interpretability of a search engine can help the user word a query for their targeted information need for more effective retrieval (Koenemann and Belkin 1996; Muramatsu and Pratt 2001; Thomas et al. 2019). Among this prior work, the "transparent query" proposed by Muramatsu and Pratt (2001) is the most similar to our own. The authors show that the users prefer guidance during the query reformulation process, and a conspicuous system which provides such techniques improves retrieval effectiveness and leads to higher user satisfaction. Instead of providing explanations for each choice made by the search engine, our goal is to generate queries that are easily interpretable by a user and that also improve the quality of the retrieval results.

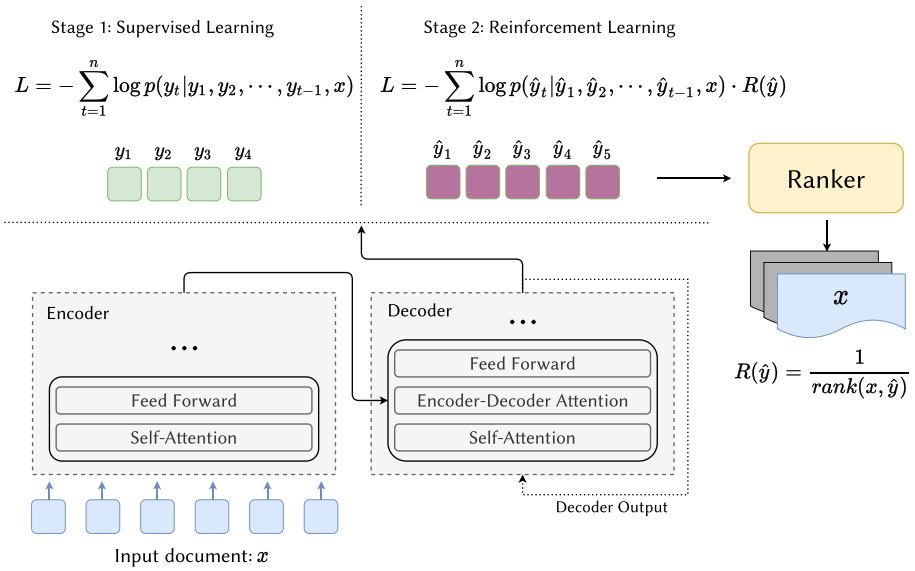


Fig. 1 The model architecture overview of our solution. First, documents are encoded (bottom of figure), and then are processed by a decoder generator. The figure illustrates a typical transformer in Fairseq (Ott et al. 2019), but any sequence-to-sequence model can be used. Stage 1 and stage 2 are trained on different collections, one for summarization and one for retrieval. Stage 1 training is shown in the top left pane where we take advantage of a summarization collection and maximize the quality of the text generator. Stage 2 training is shown in the top right, where the model is retrained using a document retrieval collection in order to optimize the retrieval effectiveness

3 Methodology

3.1 System overview

Figure 1 provides a high-level view of our system architecture. The only hard constraint on the summarization model deployed is that it must be retrainable directly with Reinforcement learning. We use a recent configuration of the Fairseq abstractive summarizer as described by Ott et al. (2019), but any state-of-the-art model could be used, as discussed later. The pace of new abstractive summarization models being proposed makes it impossible to use the "best known" model in our experiments as it changes daily on the major leader-boards.

We have adopted a two-stage training strategy in our framework. First we fine-tune a model for the summarization task on a target collection. Once the model reliably generates high quality summaries for documents in the collection, we further train the model for our second optimization goal: query effectiveness. The model is optimized to generate queries which rank the original documents at the topmost positions using commonly used terms whenever possible.

We achieve the effectiveness goal through reinforcement learning as it enables us to optimize non-differentiable targets directly. We cast the query generation problem as a contextual bandit problem and uses policy gradient algorithm to optimize our model. We did not use the more direct approach of training the two objectives (summarizer quality and

ranking effectiveness) jointly as two unrelated objectives may not converge. More specifically, we are faced with two competing objectives which commonly rely on two different types of loss function. The readability objective generates text capturing properties from the queries and collection and is as close to the human-written ground truth as possible, while the effectiveness objective reweights term choice to improve effectiveness.

Generally speaking, text generators often rely on encoder-decoder based architectures while ranking models often rely on encoder based architectures. So combining the two objectives to find an optimal solution is a challenging problem, and is not achievable by simply combining the two objective loss functions and optimizing the combination directly.

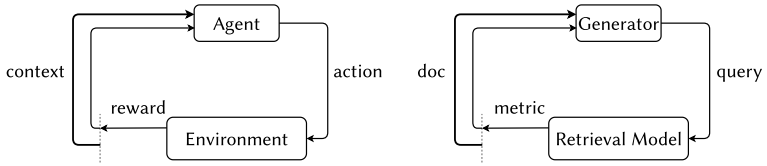
The order of the two stages — supervised learning followed by reinforcement learning — is also important. First stage training of the summarizer ensures that the query generator generates high quality text snippets. Using reinforcement learning tasks for high-dimensional action spaces are susceptible to "the curse of dimensionality" problem (Bertsekas and Tsitsiklis 1995) which can prevent convergence during training in certain circumstances (Dulac-Arnold et al. 2016; Zahavy et al. 2018). So, unless a properly trained summarization model is used, the query generator will select terms from a random distribution over the entire term vocabulary at each time step. Concretely, reinforcement learning is guided by the reward, which would be *zero* for randomly selected terms, and produce no gradient. This is because random terms do not capture real-world term dependencies and are not relevant to the source document or the query, and so the model gets no reward. In our experiments, we observed that the RL model did not converge until we introduced first stage training for the target collection. So, the summarizer training stage is crucial as it provides the necessary contextual knowledge between terms in a target document, and reduces the search space that the query generator must explore.

3.2 Summarizer

As discussed above, we use a configuration of the Fairseq abstractive summarizer as described by Ott et al. (2019) as our starting point for the summarization objective. As a point of reference, the model trained achieves ROUGE-{1,2,L} scores of 41.34, 18.35, and 37.16, which are comparable with the most effective abstractive summarization results available for the CNN/Dailymail collection based on the current leaderboard for the collection. Fairseq uses a sequence-to-sequence transformer model consisting of two parts, an encoder stack which encodes the input sequence, and a decoder stack which generates sequences given the encoded information. The architecture applies attention (Vaswani et al. 2017), with 6 stacked layers, and each layer contains an 8-head self-attention layer and a 2048 hidden-unit feed-forward network. We refer the readers to the Fairseq GitHub repository[1] for further implementation details.

Note that we also adopt the widely used "teacher forcing" (Williams. and Zipser 1989) method to train the model for summarization, which minimizes the maximum-likelihood loss at each decoding step. In this approach, the input token of the decoder is used as a ground truth token instead of the final token predictions at training time. Formally, given a training pair (x, y), where x is a document, and $y = \{y_1, y_2, \cdots, y_n\}$ is a ground-truth summary, the cross entropy loss is defined as:

[1] https://github.com/pytorch/fairseq



(a) Contextual Bandit **(b)** Query Generation.

Fig. 2 Query generation as a contextual bandit

$$L = -\sum_{t=1}^{n} \log p(y_t|y_1, y_2, \cdots, y_{t-1}, x) \tag{1}$$

At each time step, the output y_t is conditioned on the ground truth tokens which is different from Eq. 3 where the actual generated sequence is used.

3.3 Query generator

Once the summarization model is trained, it is retrained for the second task using the MS-MARCO collection. A key limitation of the query generation task is that there is a limited set of queries that can be regarded as "labels". However, in our task, we do know the original document used as input and it can be used as a relevance label just as is common practice on the known-item finding task.

3.3.1 Contextual bandit

We model the query generation process as a contextual bandit problem. Figure 2 illustrates how a contextual bandit is adapted to the query generation task. In Fig. 2a, an agent tries to find the best action given different contexts. It issues actions to the environment and receives a reward. The agent uses the reward to learn which actions are better and is optimized to favor high-reward actions. Correspondingly, in Fig. 2b, our generator generates a query from a document, and receives an effectiveness score from the retrieval model. Given both, we are in a position where we can take advantage of optimization techniques used in contextual bandit problems, with the policy gradient algorithm being the key addition.

3.3.2 Policy learning

We use a summarizer fine-tuned for the target collection as the starting point of a policy network. A policy is a decision-making rule used by the agent to determine its next action. For instance, the reward for every possible action can be estimated, and then the one with the highest reward is chosen. This is known as a *value-based method*. However this approach is not commonly used for high-dimensional action space problems. In the query generation problem, one query can be viewed as an action, and the possible number of queries are enormous unless a set of constraints are imposed, such as query length. However, a policy-based method can be used to produce an action which it believes is promising without needing to consider the entire action space. The policy is then optimized based on the reward provided by the environment. If an action results in a high reward then it is reinforced using a policy gradient algorithm, and vice versa. For additional details on the

theoretical formulation of policy gradient methods and their motivation, see Chapter 13 of Sutton and Barto (2018).

Given a training document x, our training target maximizes the reward received from the generated query $\hat{y}$, which is defined as

$$R(\hat{y}) = \begin{cases} \frac{1}{\text{rank}\,(x,\hat{y})} & \text{if x is retrieved using } \hat{y} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where rank is the rank position of document x in a rank list scored with a retrieval model $\mathscr{M}$. The reward can also be defined to incorporate other features such as query length. In our model, we use the REINFORCE (Williams 1992) training algorithm, and the loss is defined as:

$$L = -\sum_{t=1}^{n} \log p(\hat{y}_t|\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_{t-1}, x) \cdot R(\hat{y}) \tag{3}$$

Note $\hat{y}_t$ is conditioned on the actual output sequence $\hat{y}_1, \cdots, \hat{y}_{t-1}$ and ground truth labels $y_1, \cdots, y_{t-1}$ are not needed. This loss is intuitively interpreted as follows: the greater the reward, the more likely the model will generate similar outputs. For example, if the probability of choosing a token is 0.5 and it delivers a very high reward, the probability of choosing this token again in the future will be increased as training continues.

4 Experiments

4.1 Collection details

We use the CNN/Dailymail collection to train and validate the summarizer, and MS-MARCO to train and test the query generator.

4.1.1 CNN/Dailymail

CNN/Dailymail is a large news corpus commonly used for the summarization task. Every document in the dataset contains an article and a few human-written highlights, which can be used as the ground truth. We truncated documents and highlights for training efficiency as in prior work (Nallapati et al. 2016; Paulus et al. 2018; See et al. 2017). Specifically, source documents are truncated to 800 terms and the highlights to 100 terms. The collection is split into training, validation, and testing splits based on publicly available data.[2] The final dataset contains 287, 227 training pairs, 13, 368 validation pairs, and 11, 490 testing pairs.

4.1.2 MS-MARCO

The MS-MARCO collection contains both document ranking and passage re-ranking tasks, with 8.8 million passages extracted from 3.6 million web documents. The document

[2] https://github.com/abisee/cnn-dailymail

Table 1 A summary of all methods used in this work

Method	Description
POP	Generate queries by popularity sampling.
DIS	Generate queries by discriminativity sampling.
GREEDY	Generated queries by greedy algorithm.
PSG	Use passage beginning as query, truncated to match queries generated by our method.
T5	A high-quality abstractive summarizer proposed by Raffel et al. (2020) and fine-tuned using query-document pairs as described by Nogueira and Lin (2019).
UniLM	A high-quality abstractive summarizer recently proposed by Dong et al. (2019). As with T5 (Nogueira and Lin 2019), we fine-tuned the model using query-document pairs for query generation.
RL	The reinforcement learning model proposed in this paper.

ranking and passage re-ranking tasks share the same queries. In our experiments, we removed around 3.5% of the documents which contained serious parsing errors, or that contained fewer than 25 terms, as these contained uninformative content such as "this site requires cookies to be enabled to function". The QREL files released with document ranking task and passage re-ranking task were joined to map a total of 309, 387 document-passage pairs, and used for training and testing in our experiments.

Our validation set contains all of the unique documents extracted from the document validation QREL file.[3] Training and testing sets were constructed as follows. All unique documents in the document training QREL file [4] minus documents in the validation set were randomly sampled to produce test and training. The final split contains 299, 535 training examples, 4926 validation examples, and 4926 testing examples.

4.1.3 Shared vocabulary

In order to transfer the model from one collection to another, a joint vocabulary was created. We used the spaCy[5] toolkit to tokenize and process the text and applied subword-nmt[6] as is common practice. A vocabulary size of 40K was used.

4.1.4 Query generator parameters

Queries were generated using beam search with a beam size of 5. Query lengths are determined in two ways, fixed or sampled. For fixed, the target was 10 tokens per query. Note that our summarizer was also initially trained with a 10 term target, as user queries rarely exceed this length (Crane et al. 2017). We also sample lengths from a *poisson* distribution between length 3 to 10 for comparison purposes.

[3] https://msmarco.blob.core.windows.net/msmarcoranking/msmarco-docdev-qrels.tsv.gz
[4] https://msmarco.blob.core.windows.net/msmarcoranking/msmarco-doctrain-qrels.tsv.gz
[5] https://spacy.io
[6] https://github.com/rsennrich/subword-nmt

4.2 Baselines

All of the approaches used for our empirical evaluation are described in Table 1. The suffix _P is appended when query length is sampled from a *poisson* distribution, and _F is used when the query length is *fixed*, which will be explained shortly. Two additional points of reference are provided that represent the two best models currently available for the abstractive summarization task for the CNN/DailyMail collection – UniLM and T5– both of which consistently generate high-quality natural language text. In our experiments, we also fine-tune UniLM using query-document pairs and T5 in the same way as initially described by Nogueira and Lin (2019). Both can be regarded as reference points for query quality, and are much larger (and recent) pre-trained transformers than the Fairseq summarizer we had available when initially undertaking this work. However, as demonstrated in Table 2, high quality summaries for a document are not necessarily effective queries. The generated summaries from these two algorithms were truncated as this makes them more comparable to our methods while still retaining the quality generated text.

4.2.1 Popularity sampling (POP)

The popularity sampling baseline was initially proposed by Azzopardi et al. (2007), and favors popular or frequent terms in a document. Formally, terms are sampled from a mixture distribution of the document and the collection.

$$p(t|\theta_d) = (1 - \lambda)p(t|d) + \lambda p(t) \tag{4}$$

where

$$p(t|d) = \frac{n(t,d)}{\sum_{t'} n(t',d)}, \quad p(t) = \frac{n(t)}{\sum_{t'} n(t'i)} \tag{5}$$

where $n(t, d)$ is term occurrences of t in d, and $n(t)$ is term occurrences in the collection.

4.2.2 Discriminativity sampling (DIS)

Discriminativity sampling is our second baseline, and also from Azzopardi et al. (2007). This method favors terms that are rare. It is proportional to the inverse term collection frequency. Compared to POP, the difference is how $p(t|d)$ is defined.

$$p(t|d) = \frac{b(t,d)}{p(t) \cdot \sum_{t' \in d} \frac{b(t',d)}{p(t')}} \tag{6}$$

where $b(t|d) = 1$ if t is present in d. For all terms in a document, the only variant is $p(t)$, so the probability of a term being sampled is proportional to the inverse term collection frequency.

4.2.3 Greedy (GREEDY)

This is the original strong query algorithm proposed by Kumar et al. (2011), which generates the shortest query capable of ranking a document at the topmost position. The algorithm incrementally selects the rarest term from the document, until the intersection of the documents containing these terms contains only the target document. Refer to the original paper for further details (Kumar et al. 2011).

Note that this algorithm has an important limitation on large collections. The algorithm behaves erratically when the target document is a subset of other documents, or if there are duplicates in the test collection. So, in some cases, the algorithm will not terminate until every term in the document has been added to the query. In the MS-MARCO collection, we encountered several instances of this erratic behavior as there are several (near-) duplicate documents. In such cases, we truncate the queries to 5 tokens. During our experiments, we found this bound to be more than sufficient to find the original document very effectively in the vast majority of cases.

4.2.4 Passage (PSG)

We also use the corresponding passage of a document released in the collection as a reference query. While long, these are human-readable and often highly effective as they are extracted directly from the original document. Human crafted summaries would perhaps be a more interesting baseline and exhibit higher variance, but are not available for MS-MARCO at this time.

Since the passages are extracted from the target documents, it is not surprising they are highly effective. We also extract the first a few tokens from the passages to match the query lengths of other methods for the fairest comparison. This is similar to the commonly used NLP approach where the first few sentences are used as a summary baseline, and is often difficult to beat in practice (See et al. 2017), particularly for news documents. When the query length is sampled from a *poisson* distribution, we refer to it as PSG_P; when the length is same as the query generated by our network, we refer to it as PSG_F.

4.3 Automatic evaluation

To evaluate the effectiveness of the generated queries, we use reciprocal rank and average rank position. As millions of iterations are required during the reinforcement learning stage, efficient ranking is critical. So, we used the Pisa search engine[7] and BM25 to iteratively rank documents and evaluate the reciprocal rank of the documents during the learning phase.

How best to evaluate fluency or readability of the generated text is less straightforward as there is no standard benchmark unless human judgments are performed. The general best-practice is still to have both human and automatic evaluation results when assessing natural language text (Hashimoto et al. 2019). So, we have considered many different measures in order to capture different dimensions of quality, including crowdsourcing of human assessments to ensure the validity of our conclusions. We will refer to automated

[7] https://github.com/pisa-engine/pisa

methods that measure some notion of natural language text quality using the generic term READ_A. Human evaluation is discussed further in the next subsection.

The first READ_A measure we consider is the Flesch reading-ease test. It is calculated based on the # of syllables, # of words, and # of sentences. Unfortunately, this metric does not account for word order. We also considered ROUGE where the original document is the reference. While researchers are often skeptical about ROUGE, we find that it does indeed correlate with our human assessment as it can capture basic higher order dependency information, but not informativeness. The only other recent solutions found in the literature leverage Perplexity (Kann et al. 2018). The measures proposed by Kann et al. (2018) are somewhat intuitive in that the idea is to get an average perplexity of the terms in the summary. However, no publicly available implementation of SLOR or WP-SLOR is currently available, and so we have computed perplexity using the pre-trained GPT model[8] which is known to generate high-quality text, and normalized by the length of the query. We refer to this automatic method as PPL/QL henceforth.

4.4 Human evaluation

As is common with all abstractive summarization techniques, the only way to confidently assess the quality of the generated text is to perform human assessments to judge the quality. So, we validate our findings using a human assessment exercise.

We gathered judgments for both Readability and Informativeness, which is standard practice when assessing the quality of natural language text generators (Filippova and Altun 2013). Readability is synonymous with the grammatical correctness of the text (Clarke and Lapata 2006; Toutanova et al. 2016) and informativeness refers to the "meaning" (Toutanova et al. 2016) or "importance" (Clarke and Lapata 2006). Readability measures if the query is grammatically correct and fluent; informativeness measures if the query is informative and representative with respect to the document. In contrast to READ_A, we will use READINF_H to refer to the human judgments.

4.4.1 Method

We performed human evaluation using a crowdsourcing exercise on Amazon Mechanical Turk.[9] We cast the evaluation process as a headline quality evaluation which is a well-defined task in the NLP community. We gathered judgments for 50 documents sampled from our holdout set. The text of document was shown to an assessor, and informed them that they would be judging a series of headlines that are automatically generated for each. Then we asked the assessor to evaluate the quality based of two different criteria: readability and informativeness. Assessors were asked to evaluate each headline using a 5-point scale. The criteria used for assessments was:

– **Readability** - is the headline grammatically correct and fluent, where a 0 = "major errors, words have no apparent ordering" through 4 = "Outstanding, grammatically correct English sentence".

[8] https://github.com/huggingface/transformers

[9] https://www.mturk.com/

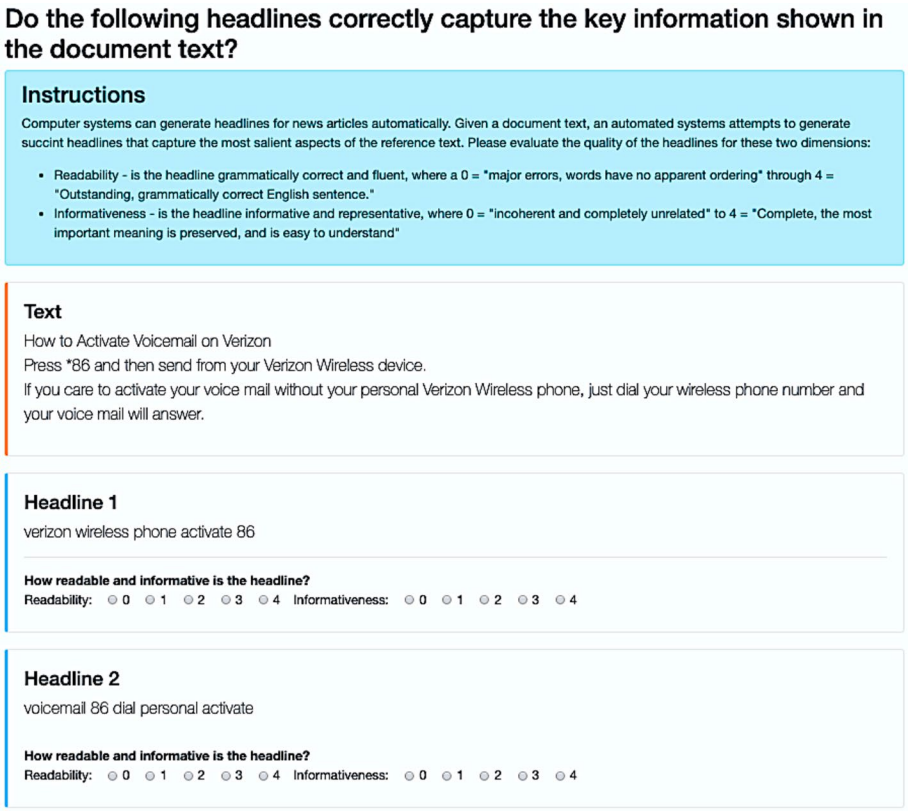


Fig. 3 A screenshot of crowdsourcing user interface

– **Informativeness** - is the headline informative and representative, where 0 = "incoherent and completely unrelated" to 4 = "Complete, the most important meaning is preserved, and is easy to understand".

For each document, we create two tasks, one for the *poisson* set and one for the *fixed* set so that they were compared directly and independently with the most appropriate counterpart. So one task presented GREEDY, PSG_P, POP_P, DIS_P and RL_P queries, and the other task presented GREEDY, PSG_F, POP_F, DIS_F and RL_F queries. The queries were shown in random order to keep the assessors from inferring any pattern. In addition, a third task was run for the same documents to gather assessments for the current state-of-the-art methods : T5 and UniLM. Both of these models are capable of generating high quality natural language text and provide an important point-of-reference for future work.

A screenshot of the user interface is show in Fig. 3. Note that five headlines were shown at one time, but are trimmed to two in the example image.

4.4.2 Crowdsourcing aggregation

We collected five assessments for every generated query in the 50 document set, and several pilots were run to determine the most suitable configuration for the crowdsourcing study. Assessors were paid \$20c USD per hit, with worker eligibility requirements set to Master, greater than 10, 000 approved HITs, and the Turker had to have a HIT approval rate for all HITs greater than 97%. A total of 32 assessors participated in our study. 9 assessors completed only 1 assignment. 7 assessors completed 2-9 assignments. 16 assessors completed at least 10 assignments. The mean and median of tasks completed by one assessor are 20 and 10 respectively. On average, each assessor spent 150 seconds to complete 1 assignment. In order to validate the quality of our judgments, we computed the inter-assessor agreement using Krippendorff's α (Krippendorff 2011), which was 0.552 for readability judgments and 0.472 for informativeness judgments. This is in line with similar crowdsourcing experiments which incorporate graded relevance, and in order to further improve the reliably, we applied the the EM (expectation maximization) algorithm (Dawid and Skene 1979) on the 5 judgments per query we collected to generate the final scores for both criteria across all methods. This approach has been shown to reliably normalize judgments and make them more resilient against assessor disagreement problems in crowdsourcing experiments, particularly when graded assessments are used (Quoc Viet Hung et al. 2013), and has been used in the same manner in several other recent studies (Ipeirotis et al. 2010; Joglekar et al. 2015, 2013). We used the publicly available EM implementation of Sinha et al. (2018) to create the final labels.

We also used a simple quality quality control test during the exercise to filter out assessors who were not clearly not assessing correctly. For this, we compared the informativeness score given for the original passage snippet against their score for the greedy algorithm, which is often one or two terms and should be the least informative. If they scored greedy higher than the actual passage, their judgments were dropped and they they were blocked from participating in the exercise.

4.4.3 Document selection

The MS-MARCO collection contains a substantial number of parsing errors and uninformative pages. For example, on a web page, "Q&A_Children can suffer" was incorrectly parsed as "Q&AChildren can suffer" (the space between "A" and "Children" is missing) and ends up with a rare but invalid word "achildren" which can be easily exploited by the GREEDY and DIS algorithm. Another example is a web page consisting of many hex color codes was excluded for evaluation as the queries produced by all of the methods were effective but also were not informative as they did not align with the original reference queries. While walking the list of available documents, we attempted to select 50 documents of reasonable length and that were parsed cleanly by the creators of the original MS-MARCO collection.

Table 2 Retrieval effectiveness and readability/fluency comparison for all 12 methods

Model	Len	RR	Rank	Scarcity	PPL/QL	ROUGE Precision				Read
					Avg	1	2	L	W	Flesch
PSG	55.09	0.790	67.3	2.9%	7.6846	91.31	81.67	87.88	77.72	21.93
GREEDY	2.44	0.923	2.3	53.1%	51260.7324	100.00	0.19	85.85	80.01	4.96
UniLM	5.5940	0.2612	146.4813	3.5526%	113412.1469	82.8436	39.9982	81.4783	65.6547	71.9400
T5	5.9330	0.2319	200.0148	3.0857%	11501.3075	80.7334	35.6415	79.0168	62.7713	75.6401
PSG_P	5.80	0.347	239.2	0.5%	8282.6021	93.68	83.25	93.36	81.36	67.16
POP_P	5.80	0.246	414.6	2.0%	28466.0371	86.23	4.43	74.03	56.53	71.92
DIS_P	5.80	0.847	11.8	52.6%	17949.3093	73.11	0.73	54.84	42.00	37.22
RL_P	5.80	0.530	114.5	1.1%	22297.5028	94.04	18.77	80.35	62.54	69.00
PSG_F	9.60	0.519	102.7	0.7%	307.5536	93.76	84.18	92.82	80.22	66.43
POP_F	9.60	0.492	159.5	3.5%	2586.8096	86.02	4.38	67.59	46.78	68.64
DIS_F	9.60	0.886	5.7	54.1%	1878.8989	66.65	0.78	46.02	31.76	37.82
RL_F	9.60	0.780	11.0	1.0%	1128.1373	94.88	20.11	75.34	53.68	68.98

5 Results

5.1 Retrieval effectiveness

The retrieval effectiveness results are shown in Table 2. First, we consider the impact of query length. PSG and GREEDY queries represent both ends of the spectrum of query length. The former is formulated using natural language and is verbose. In our experiments the average length of PSG is 55.09 terms. Conversely, GREEDY extracts the most discriminative terms yielding queries that contain 2.44 terms on average, which is even lower than the average of 3.26 originally reported by Kumar et al. (2011).

We report both reciprocal rank (RR) and average rank for effectiveness comparison as RR is the evaluation method used to determine the leader-board for MS-MARCO. Both PSG and GREEDY queries are highly effective. The reciprocal rank of PSG is 0.790 and an average rank is 67.3. Note that we use a simple bag-of-words ranker (BM25) and performance would very likely be improved using any of the more complex deep learning models that dominate the leader-board. Nevertheless, the queries induced are highly effective with even simple ranking models. So, while often effective, long passages extracted directly from a document are not necessarily good queries for refinding the document. Given that there are similar or even duplicate documents in the collection, the variance is as expected. The GREEDY algorithm is designed specifically to retrieve the original document, and this is confirmed by the impressive RR and average rank effectiveness the model achieves. However, duplicate documents again have an impact on the quality.

Following GREEDY in Table 2 are the two best abstractive summarizers T5 and UniLM. They achieve surprisingly low retrieval effectiveness, further justifying the use

of a reinforcement learning retraining stage as proposed in this work. We suspect that their low effectiveness may be related to their "abstractive" nature which can improve the linguistic content, but not refinding.

We now turn to the Poisson sampling experiments. For each document, we sample a length from the Poisson distribution, and all systems generate queries of that length. In this group, PSG_P and POP_P queries have the worst reciprocal rank and average rank. This is understandable as PSG_P can contain several stop words. The same limitation affects POP_P as sampling is based purely on frequency, and stop words have a high probability. DIS_P performs the best among this group, with a reciprocal rank of 0.847 and average rank of 11.8. Essentially, DIS is a non-deterministic variation of the GREEDY algorithm, resulting is a similar behavior of choosing rare words as a query. Both methods exploit rare terms in a document, so very few terms can often be found to reliably refind the original document. Our proposed RL model achieves a reciprocal rank 0.530 and average rank 114.5, and as we will show shortly also performs well for our other target goal – informativeness / readability. When considering only effectiveness, it falls between POP_P and DIS_P, and is consistently better than PSG_P.

The final grouping compares fixed length generation. While 10 terms is rather long for a "normal human query", it is a realistic target for a compressed summary that retains some natural language properties, thus making them more understandable (hopefully). Compared to the Poisson results, we see that retrieval performance improves as the query length grows. This is again not unexpected as verbose queries often favor precision over recall (Gupta and Bendersky 2015). Note that the PSG_F and POP_F queries show considerable improvements in this configuration when compared with DIS_F queries. This is because PSG_F and POP_F queries are more likely to incorporate at least one or more discriminative terms as the length increases. Another major improvement can be observed for RL_F. This could be an artifact of the initial summary training stage which also targeted ten terms, and will be investigated further in future work.

In general, our retrieval results align with our intuition. PSG queries are extracted from the original document and do not suffer from vocabulary mismatches, so they are often effective queries for refinding. POP is inferior to DIS. With very few rare terms, GREEDY can consistently produce short and effective queries. Our RL method also reliably improves the effectiveness of the queries, easily outperforming PSG, POP, UniLM and T5, and is nearly as effective as DIS when query length is increased.

5.2 Automatic evaluation results

As shown in the previous section, retrieving the original document is not a difficult problem from an algorithmic perspective if only rare terms are used. But how readable and informative are such queries? Do they capture the most salient aspects of the underlying document? Can users easily understand their intention? The answer is almost certainly no in many cases. So there is a natural tension between the effectiveness of a query and the interpretability of it.

Recall in Sect. 4.3 that reliably measuring fluency in an automated manner is still an open research problem. In Table 2 we compare a variety of different evaluation metrics in order get a better indication of the quality of the queries being generated. All of these metrics have been used for automated evaluation of human generated text in the past. The first metric we consider and arguably still the most commonly used one for summarization tasks is ROUGE. We report only precision-based ROUGE where the document is the

reference. PSG 's ROUGE scores are high, but not 100 as expected because there are a few incorrect document to passage mappings in the MS-MARCO collection. The creators have acknowledged that the passage corpora and document corpora are collected at different times, causing such inconsistencies. Overall, queries derived from passages all have high ROUGE scores. For GREEDY, ROUGE-1 clearly shows that all the terms are from the document. ROUGE-2, ROUGE-L and ROUGE-W may not be meaningful for the GREEDY algorithm as 2, 414 out of 4, 926 queries contain only one term, resulting in a score of 0 for ROUGE-2 and score of 100 for ROUGE-L and ROUGE-W.

The ROUGE scores of UniLM and T5 also have several notable properties. Both systems prefer more bigrams (high ROUGE-2) than the other methods, and maintain term ordering (high ROUGE-L) even when they are not adjacent. This is an important contributing factor to their highly readable text.

The ROUGE scores for the Poisson sampled methods reveal that DIS_P are the least compliant for ordered sequence measures, and POP_P also commonly permutes term orderings. The RL method achieves higher scores especially in ROUGE-2, ROUGE-L, and ROUGE-W, which is promising as it suggests the model has retained its NLG capabilities learned during summarization. Similar trends can be observed in the fixed length grouping.

The last automatic metric we consider is the Flesch score. The readability score for PSG is low since punctuation was removed for query purposes, thus removing the sentence normalization factor. GREEDY queries also perform quite poorly. Despite being short, the number of syllables in the terms selected are large, indicating they are indeed rarely seen terms, such as "nebuchadnezzar" or "hypochlorous". For all other methods, Flesch scores are similar except for the DIS which also favors rare terms.

While ROUGE and Flesch provide some insight into the characteristics of these queries, they capture little or nothing about grammatically, informativeness, or fluency. So, our final automated evaluation technique leverages perplexity, which we derive using a high quality pretrained language model OpenAI GPT. The score produced is inversely proportional to the chance of a query being a proper sentence in the collection using a high quality NLG model. PSG has the lowest score, or highest chance of being natural language. GREEDY has the highest perplexity score except for UniLM which we will analyze later, again illustrating the two ends of the spectrum. For Poisson sampled methods, PSG_P has the lowest score, POP_P is least likely based on the reference distribution, and surprisingly DIS_P is the best, with our RL method falling somewhere in the middle. For fixed length methods, DIS_F is still better than POP_F, which is also surprising. For longer queries, RL outperforms both POP_F and DIS_F.

5.2.1 Effectiveness-readability tradeoffs

Until now, we have only considered effectiveness and quality as independent goals, but recall that our real goal is to find the best trade-off between both of these objectives. This is best visualized by the Pareto frontier between readability and effectiveness. Figure 4 shows the trade-off between retrieval effectiveness (reciprocal rank) and query fluency measured as the perplexity normalized by query length. The best performing methods appear in the top left quadrant of the tradeoff graph. We can see that our proposed RL approach does indeed find the best balance between the two competing goals.

Since the retrieval problem is trivial for DIS_P and GREEDY if the rarest terms can be selected from a document (even parsing errors), we also explore their performance when the vocabulary is limited. We explored limiting the vocabulary for the two methods to the

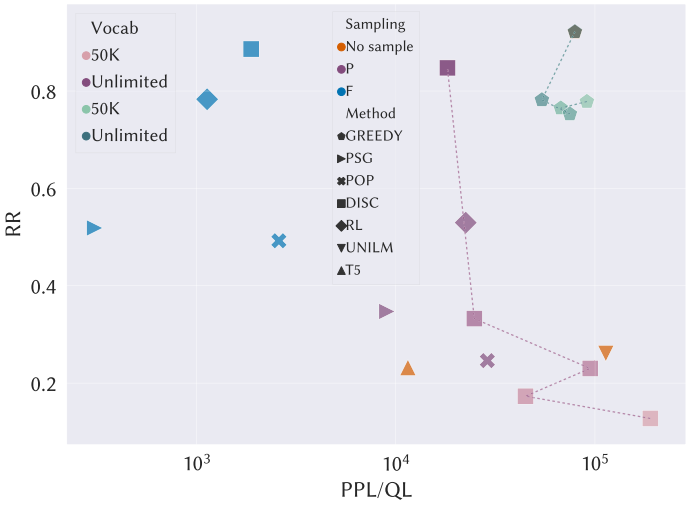


Fig. 4 Effectiveness-Readability Tradeoffs for all models. The dashed lines show the average effectiveness for DIS_P and GREEDY when vocabulary size is limited

most frequent 50K, 100K, 200K, and 500K terms, and measured the performance impact. Figure 4 shows the changes in performance for both of these algorithms when allowed to only use the most common 50, 100, 200, or 500 thousand terms in the collection. Overall DIS_P is highly sensitive to vocabulary size. We also observed that the performance of DIS_P can be unstable when using a limited vocabulary. The reason is that rare terms are no longer available, and so term impact varies less, and random sampling increases the variance as the term frequency distributions become less skewed. The GREEDY method is remarkably consistent across different vocabulary sizes, as it can always take full advantage of the rarest terms remaining in any vocabulary.

Results for T5 and UniLM can be seen in the bottom right corner of Fig. 4. We observed high perplexity for these models. This is somewhat unexpected, but we believe it is related to how we have conditioned perplexity on the GPT language model, and not the models used to generate the queries. Care should be taken when using perplexity to automatically evaluate quality, as we have observed that the results are strongly dependent on the text the language model is conditioned on.

5.3 Human evaluation results

Given the newly created judgments gathered with Turk we can properly assess the quality of our results in terms of human readability. The key results are shown in Table 3. Similar to the complete holdout test set results, we also computed a battery of the automated metrics. The automated metrics do indeed demonstrate similar trends in Sect. 5.2 when compared directly against the human assessments. The final human assessed scores are shown on the right of the table under "Human/Readability" and "Human/Informativeness". The PSG_P and PSG_F tend to result in high quality queries as they were extracted directly from the documents, but truncation clearly has an effect on both. The longer the query,

Table 3 Natural language evaluation for the 50 documents assessed by humans, where † signifies significance at $p \leq 0.05$ in a paired t-test relative to our methods RL_P or RL_F

	PPL/QL	ROUGE Precision				Readability	Human Evaluation	
Model	Avg	1	2	L	W	Flesch	Read	Info
GREEDY	46718.0026	100.0000	1.8889	72.2492	59.8296	6.6512	1.20†	0.44†
UniLM	27101.7811	82.8373	41.6571	82.2273	66.0661	66.9024	3.06†	2.76†
T5	3023.1643	82.1993	38.4683	80.6680	63.9631	72.4522	3.36†	2.48†
PSG_P	6588.3780	93.1889	83.3444	92.7618	81.9077	70.5106	2.82†	1.92†
POP_P	240609.6429	85.8278	5.7548	72.1331	55.6146	71.6948	0.62†	0.12†
DIS_P	126284.6189	79.0492	1.5000	57.0534	43.8882	24.4860	0.62†	0.24†
RL_P	14152.5832	98.1389	16.6794	81.5964	63.9150	66.9576	1.74	0.80
PSG_F	152.1972	92.1333	81.0278	90.6054	78.9792	68.5222	3.40†	3.02†
POP_F	2467.5493	84.8365	5.8135	65.2798	45.5146	69.3684	0.34†	0.06†
DIS_F	8118.2435	71.1516	1.3611	47.5165	32.7808	34.3424	0.06†	0.12†
RL_F	750.7770	96.9056	19.9524	73.4767	52.8070	70.0454	1.74	1.20

the better they perform. Interestingly, UniLM and T5 received the highest readability and informativeness scores after PSG_F, indicating they are very good at representing the document and they can generate coherent and succinct queries that are of high quality. However, as discussed previously, they are not necessarily effective queries. When compared to other baselines, our approach RL_P and RL_F are significantly different based on our preliminary human assessments.

5.3.1 Tradeoffs revisited

We are now in a position to fully assess our primary objective, which was to find the best tradeoffs between effectiveness and readability/informativeness.

The GREEDY algorithm remains the strongest in terms of raw effectiveness (RR) and in either *poisson* (suffixed by _P) or *fixed* set (suffixed by _F), our proposed methods RL rank second after the DIS approach for effectiveness. This trend is consistent with experiments on the complete holdout set. Importantly, we can also see from the results of UniLM and T5 that highly readable and informative summaries are not necessarily effective queries. The natural language quality of these two systems was the highest in Table 3, but the effectiveness shown here is lower than methods that are optimized directly or indirectly for effectiveness. The comparison of RL against UniLM and T5 further illustrate the importance of the proposed second stage reinforcement learning training for balancing effectiveness and natural language quality.

We plot each system's effectiveness and the aggregation of readability and informativeness score in Fig. 5. Figure 5.1 illustrates the tradeoff between effectiveness and human evaluated readability and informativeness. PSG_F, PSG_P, T5, and UniLM are the clear winners on readability and informativeness but all are clearly less effective. Similarly, the GREEDY and DIS approaches achieve high effectiveness but were determined to be less readable by human assessors. Our system sits in a more balanced position. Figure 5.2

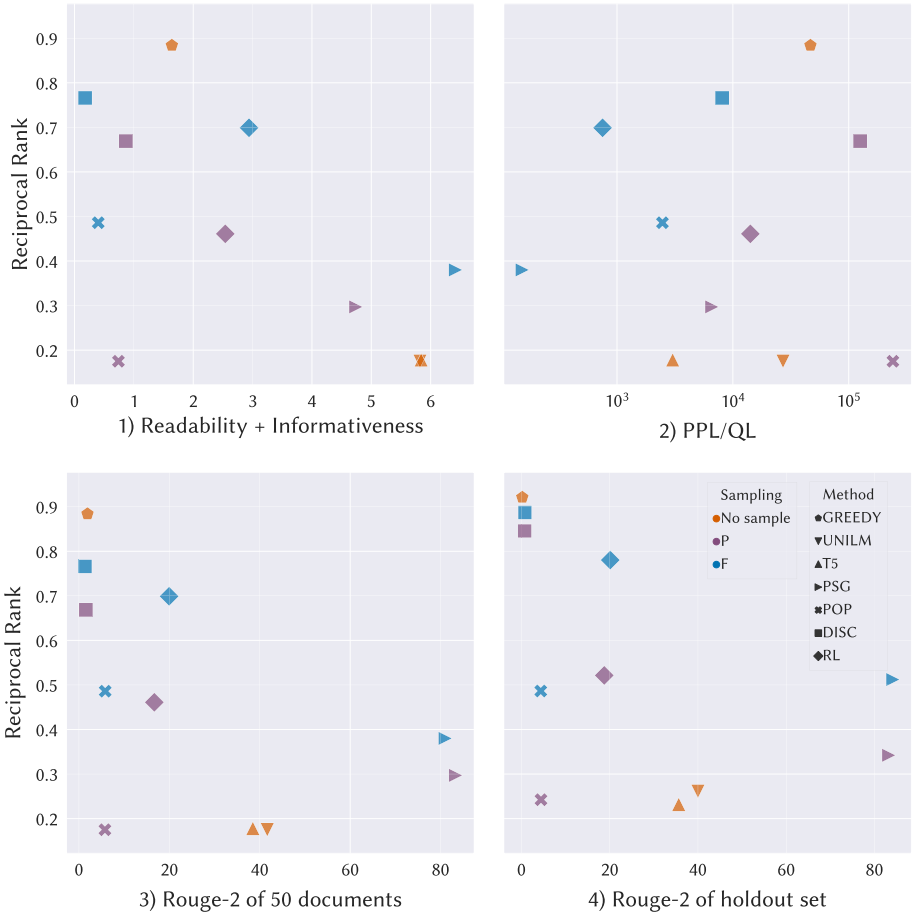


Fig. 5 Effectiveness-readability/informativeness tradeoff of all systems

illustrates effectiveness versus PPL/QL. Figure 5.3 extends human evaluation to Rouge-2, as we found Rouge-2 is correlated with human evaluation with a Pearson correlation coefficient of 0.64. The correlation can also be visually observed in the figure. Figure 5.4 confirms that the trends for ROUGE-2 on the human assessed documents are consistent with those computed for the complete holdout set, suggesting that ROUGE-2 is indeed a reasonable surrogate for a human assessment when attempting to benchmark quality automatically, at least for preliminary testing of a new model.

5.4 Qualitative analysis

We now discuss the results produced for some documents in order to provide a little more intuition into the queries being generated by each approach, that were subsequently evaluated by the crowdworkers.

Table 4 Query examples for documents in the MS-MARCO test collection

System	Query
Query examples for document D133390	
RL_F	Susan the tennessee civil constitution movement gave women right
T5	Why was the 19th amendment important
UniLM	How did the 19 th amendment change voting rights for women
POP_F	The last became was illegal under state th get
DIS_F	Leser schenck the commentscomment symbolic lawmaker electorate cady this
GREEDY	Leser commentscomment
Query examples for document D143592	
RL_F	Sony full frame produce at 400 plus mm lens but
T5	Difference between full frame and aps-c
UniLM	Difference between full frame and aps - c
POP_F	Full and many are the com it sony lens there
DIS_F	tcav new normal that or dimmest become founder by were
GREEDY	tcav luminance
Query examples for document D2989514	
RL_F	Microsoft visual studio code is close to community operation
T5	What is a linq
UniLM	.net framework extensions
POP_F	Preview technology the center all data close laptop visual
DIS_F	Nowmicrosoft and water fashion tae assume language tree fodmap
GREEDY	Nowmicrosoft extension

The queries generated by the six systems are shown in Table 4. Document D133390 is a description of the 19th amendment which grants women the right to vote. Both T5 and UniLM clearly generate high quality text. POP_F and DIS_F, as their names suggest, select popular and discriminative words respectively. Neither model attempts to induce a natural language ordering of the terms selected, making them much more difficult to comprehend. Similarly, GREEDY, a deterministic version of DIS_F, generates the two rarest terms from the document and provides no meaningful information from the original document. RL_F is not fully correct grammatically since there is a bias towards discriminative terms, and so the term-wise probability is skewed when beam search is applied. Nevertheless, semantically correct phrases such as "gave women right" show that the natural language summarizer is having a positive effect. The method also selected keywords such as "susan", a key figure in the movement, and "tennessee", where the final needed approval was passed increasing the informativeness scores while also being discriminative. In other examples, we can see a similar trend. When compared to T5 and UniLM, our method sometimes sacrifices grammatical correctness and selects "higher impact" terms for the ranking model being used – increasing the likelihood of refinding the document in the collection.

In summary, the best-known techniques such as T5 and UniLM are consistently good at generating human readable text that accurately represent the source document, while our

proposed algorithmic framework retains important properties from summarization but also has much better retrieval effectiveness. Some technical limitations of our proposed method can also be observed in the queries being generated—some readability is being sacrificed in the second stage in order to improve ranking effectiveness, causing the generated queries to be less syntactically complete. In future work we intend to continue exploring how both of these important objectives might be directly optimized simultaneously, further improving our ability to find the more desirable trade-offs between these two competing objectives.

6 Conclusion

In this work, we formalize and study the SNLQ problem. By combining state-of-the-art abstractive summarization with reinforcement learning, we are able to learn a model capable of generating queries that balances the competing objectives of ranker effectiveness and human readability. In future work, we intend to continue our exploration of techniques which better enable direct optimization of two or more competing goals. We are also planning a deeper investigation into better methods to more reliably measure human-readability of the text being produced in NLG related tasks. Automatic and comprehensive evaluation of high-order natural language models such as T5 and UniLM continues to be an elusive and open research problem.

Funding This work was supported by the Australian Research Council's Discovery Projects Scheme (DP170102231).

Availability of data and material All data is publicly available.

Declarations

Conflict of interest Editor conflicts are Shane Culpepper.

Code availability Source code for all of the models described is available upon request.

References

Ahmad, WU., Chang, KW., & Wang, H. (2019). Context attentive document ranking and query suggestion. In: Proc. SIGIR, pp 385–394

Azzopardi, L., & de Rijke, M. (2006). Automatic Construction of Known-item Finding Test Beds. In: Proc. SIGIR, pp 603–604

Azzopardi, L., de Rijke, M., & Balog, K. (2007). Building simulated queries for known-item topics: An Analysis using Six European Languages. In: Proc. SIGIR, pp 455–462

Bailey, P., Moffat, A., Scholer, F., & Thomas, P. (2016). UQV100: A test collection with query variability. In: Proc. SIGIR, pp 725–728

Bailey, P., Moffat, A., Scholer, F., & Thomas, P. (2017). Retrieval consistency in the presence of query variations. In: Proc. SIGIR, pp 395–404

Belkin, N.J., Cool, C., Croft, WB., & Callan, J.P. (1993). The effect of multiple query variations on information retrieval system performance. In: Proc. SIGIR, pp 339–346

Bendersky, M., Metzler, D., & Croft, WB. (2012). Effective query formulation with multiple information sources. In: Proc. WSDM, pp 443–452
Berger, A., & Lafferty, J. (1999). Information retrieval as statistical translation. In: Proc. SIGIR, pp 222–229
Bertsekas, D., & Tsitsiklis, J. (1995). Neuro-dynamic programming: An overview. In: Proceedings of 1995 34th IEEE Conference on Decision and Control, vol 1, pp 560–564
Boldi, P., Bonchi, F., Castillo, C., Donato, D., Gionis, A., & Vigna, S. (2008). The query-flow graph: Model and applications. In: Proc. CIKM, pp 609–618
Cai, F., & de Rijke, M. (2016). A survey of query auto completion in information retrieval. *Foundations and Trends in Information Retrieval, 10*(4), 273–363.
Callan, J., & Connell, M. (2001). Query-based sampling of text databases. *ACM Transactions on Information Systems, 19*(2), 97–130.
Clark, K., Khandelwal, U., Levy, O., & Manning, CD. (2019). What does BERT look at? An analysis of BERT's attention. In: Proc. BlackboxNLP@ACL, pp 276–286
Clarke, J., & Lapata, M. (2006). Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In: Proc. ACL, pp 377–384
Cohan, A., Dernoncourt, F., Kim, DS., Bui, T., Kim, S., Chang, W., & Goharian, N. (2018). A discourse-aware attention model for abstractive summarization of long documents. In: Proc. NAACL-HLT
Crane, M., Culpepper, J.S., Lin, J., Mackenzie, J., & Trotman, A. (2017). A comparison of document-at-a-time and score-at-a-time query evaluation. In: Proc. WSDM, pp 201–210
Dai, Z., & Callan, J. (2019). Deeper text understanding for IR with contextual neural language modeling. In: Proc. SIGIR, pp 985–988
Dang, V., & Croft, WB. (2010). Query reformulation using anchor text. In: Proc. WSDM, pp 41–50
Dawid, A. P., & Skene, A. M. (1979). Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society Series C (Applied Statistics), 28*(1), 20–28.
Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M., & Hon, HW. (2019). Unified language model pre-training for natural language understanding and generation. In: Proc. NeurIPS, pp 13042–13054
Dulac-Arnold, G., Evans, R., van Hasselt, H., Sunehag, P., Lillicrap, T., Hunt, J., Mann, T., Weber, T., Degris, T., & Coppin, B. (2016). Deep Reinforcement Learning in Large Discrete Action Spaces. arXiv:151207679 [cs, stat] 1512.07679
Filippova, K., & Altun, Y. (2013). Overcoming the lack of parallel data in sentence compression. In: Proc. EMNLP, pp 1481–1491
Gupta, M., & Bendersky, M. (2015). Information retrieval with verbose queries. *Foundations and Trends in Information Retrieval, 9*(3–4), 209–354.
Hashimoto, T.B., Zhang, H., & Liang, P. (2019). Unifying human and statistical evaluation for natural language generation. In: Proc. NAACL-HLT, pp 1689–1701
Hauff, C., Hagen, M., Beyer, A., & Stein, B. (2012). Towards realistic known-item topics for the clueweb. In: Proc. IIiX, pp 274–277
He, Y., Tang, J., Ouyang, H., Kang, C., Yin, D., & Chang, Y. (2016). Learning to rewrite queries. In: Proc. CIKM, pp 1443–1452
Huang, C. K., Chien, L. F., & Oyang, Y. J. (2003). Relevant term suggestion in interactive web search based on contextual information in query session logs. *Journal of the American Society for Information Science and Technology, 54*(7), 638–649.
Ipeirotis, PG., Provost, F., & Wang, J. (2010). Quality management on Amazon Mechanical Turk. In: Proc. SIGKDD Workshop HCOMP, p 64
Jiang, J.Y., & Wang, W. (2018). Rin: reformulation inference network for context-aware query suggestion. In: Proc. CIKM, pp 197–206
Joglekar, M., Garcia-Molina, H., & Parameswaran, A. (2013). Evaluating the crowd with confidence. In: Proc. KDD, pp 686–694
Joglekar, M., Garcia-Molina, H., & Parameswaran, A. (2015). Comprehensive and reliable crowd assessment algorithms. In: Proc. ICDE, pp 195–206
Kann, K., Rothe, S., & Filippova, K. (2018). Sentence-level fluency evaluation: References help, but can be spared! In: Proc. ACL, pp 313–323
Koenemann, J., & Belkin, NJ. (1996). A case for interaction: A study of interactive information retrieval behavior and effectiveness. In: Proc. SIGCHI, pp 205–212
Krippendorff, K. (2011). Computing Krippendorff's Alpha-Reliability. Departmental Papers (ASC) Retrieved from https://repository.upenn.edu/asc_papers/43
Kumar, R., Lattanzi, S., & Raghavan, P. (2011). An Algorithmic Treatment of Strong Queries. In: Proc. WSDM, pp 775–784
Lavrenko, V., & Croft, WB. (2001). Relevance based language models. In: Proc. SIGIR, pp 120–127

Lee, CJ., & Croft, WB. (2012). Generating Queries from User-Selected Text. In: Proc. IIiX, pp 100–109
Liu B, Craswell N, Lu X, Kurland O, Culpepper JS (2019) A Comparative Analysis of Human and Automatic Query Variants. In: Proc. ICTIR, p 4
Muramatsu, J., & Pratt, W. (2001). Transparent queries: Investigation users' mental models of search engines. In: Proc. SIGIR, pp 217–224
Nallapati, R., Zhou, B., dos Santos, C., Gulcehre, C., & Xiang, B. (2016). Abstractive text summarization using sequence-to-sequence rnns and beyond. In: Proc. SIGNLL, pp 280–290
Nogueira, R., & Lin, J. (2019). From doc2query to doctttttquery. https://cs.uwaterloo.ca/~jimmylin/publications/Nogueira_Lin_2019_docTTTTTquery-v2.pdf, accessed 2020-01-21
Nogueira, R., Yang, W., Lin, J., & Cho, K. (2019). Document expansion by query prediction. arXiv:190408375
Ogilvie, P., & Callan, J. (2003). Combining Document Representations for Known-Item Search. In: Proc. SIGIR, pp 143–150
Ott, M., Edunov, S., Baevski, A., Fan, A., Gross, S., Ng, N., Grangier, D., & Auli, M. (2019). fairseq: A fast, extensible toolkit for sequence modeling. In: Proc. ACL, pp 48–53
Paulus, R., Xiong, C., & Socher, R. (2018). A Deep Reinforced Model for Abstractive Summarization. In: Proc. ICLR
Hung, Quoc Viet, & N., Tam, N. T., Tran, L. N., & Aberer, K. . (2013). An Evaluation of Aggregation Techniques in Crowdsourcing. *Proc. WISE, 8181,* 1–15.
Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research, 21*(140), 1–67.
See, A., Liu, PJ., & Manning, CD. (2017). Get to the point: Summarization with pointer-generator networks. In: Proc. ACL, pp 1073–1083
Sheldon, D., Shokouhi, M., Szummer, M., & Craswell, N. (2011). Lambdamerge: merging the results of query reformulations. In: Proc. WSDM, pp 795–804
Sinha, VB., Rao, S., & Balasubramanian, VN. (2018). Fast Dawid-Skene: A Fast Vote Aggregation Scheme for Sentiment Classification. In: Proc. KDD WISDOM, p 8
Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Simonsen, JG., & Nie, JY. (2015). A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In: Proc. CIKM, pp 553–562
Sutton, R. S. (2018). *& Barto, AG*. An introduction: Reinforcement learning.
Teevan, J. (2007). The Re: search engine: simultaneous support for finding and re-finding. In: Proc. UIST, pp 23–32
Thomas, P., Billerbeck, B., Craswell, N., & White, R. W. (2019). Investigating searchers' mental models to inform search explanations. *ACM Transactions on Information Systems, 38*(1), 1–25.
Toutanova, K., Brockett, C., Tran, KM., & Amershi, S. (2016). A Dataset and Evaluation Metrics for Abstractive Compression of Sentences and Short Paragraphs. In: Proc. EMNLP, pp 340–350
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, AN., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In: Proc. NeurIPS, pp 6000–6010
Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning, 8*(3), 229–256.
Williams, R. J., & Zipser, D. (1989). A learning algorithm for continually running fully recurrent neural networks. *Neural Computation, 1*(2), 270–280.
Wu, B., Xiong, C., Sun, M., & Liu, Z. (2018). Query suggestion with feedback memory network. In: Proc. WWW, pp 1563–1571
Xue, X., & Croft, W. B. (2013). Modeling reformulation using query distributions. *ACM Trans Inf Syst, 31*(2), 6:1-6:34.
Zahavy, T., Haroush, M., Merlis, N., Mankowitz, DJ., & Mannor, S. (2018). Learn What Not to Learn: Action Elimination with Deep Reinforcement Learning. Advances in Neural Information Processing Systems pp 3562–3573