Learning to Acquire Knowledge from a Search Engine for Dialogue Response Generation

Anonymous ACL submission

Abstract

Knowledge-aided dialogue response generation aims at augmenting chatbots with relevant external knowledge in the hope of generating more informative responses. The majority of previous work assumes that the relevant knowledge is given as input or retrieved from a static pool of knowledge. However, this assumption violates the real-world situation, where knowledge is continually updated and a chatbot has to *dynamically* retrieve useful knowledge. In this paper, we propose a dialogue model that can access the vast and dynamic information from any search engine for response generation. To this end, we design a query producer that generates queries from a dialogue context to interact with a search engine. The query producer is trained without any human annotation of gold queries, making it easily transferable to other domains and search engines. More specifically, we design a reinforcement learning algorithm to train the query producer, where rewards are obtained by comparing retrieved articles and gold responses. Experiments show that our query producer can achieve R@1 and R@5 rates of 62.4% and 74.8% for retrieving gold knowledge, and the overall model generates better responses over a strong BART (Lewis et al., 2020) model and other typical baselines.

1 Introduction

The task of knowledge-aided dialogue response generation aims to find useful knowledge for an on-going conversation to help a chatbot generate more relevant and engaging responses. This is an important direction for dialogue response generation due to three advantages: (1) it allows a dialogue model to access a large pool of knowledge beyond local conversational contexts; (2) it enables a dialogue model to capture the dynamic nature of the world (Komeili et al., 2021), where knowledge sources are frequently updated; (3) it may enhance the interpretability of dialogue models by examining retrieved knowledge and allows fine-grained interventions by replacing certain pieces of knowledge (Adiwardana et al., 2020; Zhang et al., 2020; Roller et al., 2021).

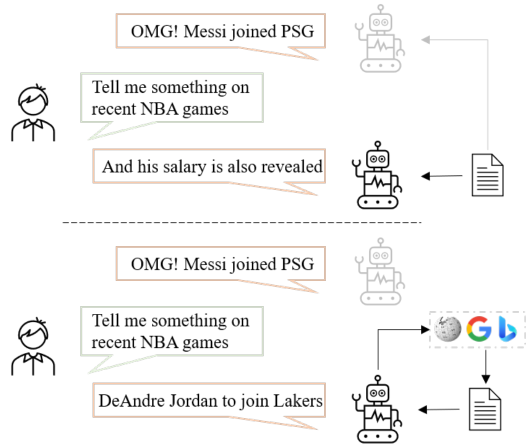

Figure 1: Previous knowledge-aided dialogue response generation (up), where related articles are given as input, versus our model (down), which can dynamically fetch knowledge from a search engine.

Initial efforts (Ghazvininejad et al., 2018; Liu et al., 2018; Wu et al., 2019; Zhou et al., 2020; Tian et al., 2020; Chen et al., 2020; Kim et al., 2020) on knowledge-aided response generation assume that relevant knowledge (e.g., news or movie reviews) is given as input and design dialogue systems that can effectively utilize the provided knowledge. However, as shown in Fig. 1, this static setting violates the dynamic nature of real-world scenarios. This gives rise to approaches that can retrieve and select information from a knowledge source for response generation (Zhao et al., 2020; Dinan et al., 2019; Lee et al., 2019). These projects assume searching from a static pool of articles (e.g., a Wikipedia dump). The queries and articles are represented as sparse vectors of n-grams (Dinan et al., 2019) or even dense contextualized vectors (Lee et al., 2019) for retrieval. However, these approaches with a static pool of knowledge still fall short of taking the dynamic nature of knowledge into account.

In this paper, we propose a dialogue model that can access the vast and dynamic knowledge from any search engine for response generation. We choose to work with search engines based on two reasons. First, search engines like Google store continually updating knowledge, which well captures the dynamic nature of our world. Second, we get rid of the difficulties of building our own search engines with n-grams and dense contextualized vectors, since the ranking algorithms of well-established search engines are highly optimized. Fig. 2 shows the framework of our model, consisting of a query producer and a response generator. The query producer generates queries from a dialogue context. Then, we send the queries to a search engine to obtain relevant articles. The response generator takes both the retrieved articles and the dialogue context to generate a response.

As a key component in our model, the query producer determines the quality of fetched knowledge, which further affects response generation. To obtain automatic training signals for our query producer, we design a function based on existing cheap noisy supervision for scoring queries. It compares the retrieved articles of a query with the corresponding gold response to estimate the quality of the query. The scoring function does not require extra annotations, such as gold queries, making our model easily transferable to other domains and search engines.

We use Wizard of Wikipedia (WoW, Dinan et al. 2019), a well-established benchmark on knowledge-aided response generation, for evaluating our model, taking the publicly free search engine from Wikipedia to retrieve knowledge instead of using the static knowledge provided by WoW. Experiments show that our query producer can achieve a R@1 (R@5) rate of 62.4% (74.8%) for retrieving the correct knowledge on the *unseen* test set of WoW. Besides, our model generates better replies than a strong BART (Lewis et al., 2020) model and knowledge-aided baselines with heuristic algorithms for query acquisition. These results indicate the feasibility of using a search engine as the knowledge source for response generation.[1]

2 Model

Formally, given a dialogue context of prior $t-1$ turns $\mathcal{D}_{<t} = \{u_1, u_2, ..., u_{t-1}\}$, our model first predicts a query $\tilde{q}$ (optionally from a set of query candidates $\mathcal{Q} = \{q^1, q^2, ..., q^{|\mathcal{Q}|}\}$ selected by a heuristic algorithm), before sending it to a search engine for retrieving a list of articles $\mathcal{K}^{\tilde{q}} = \{k_1^{\tilde{q}}, k_2^{\tilde{q}}, ..., k_{|\mathcal{K}^{\tilde{q}}|}^{\tilde{q}}\}$. With the retrieved knowledge $\mathcal{K}^{\tilde{q}}$ and dialogue context $\mathcal{D}_{<t}$, a response u_t is generated.

Fig. 2 visualizes the workflow of our model. In the rest of this section, we introduce the two key components, the query producer (§2.1) and the response generator (§2.2).

2.1 Query Production

We explore two popular directions based on either extraction (§2.1.1) or generation (§2.1.2) to build our query producer. We further prune the query search space to minimize the number of possible queries and speed up training (§2.1.3). We use cheap noisy supervisions to train the query producers with MLE-based pre-training and reinforcement learning fine-tuning (§2.1.4).

2.1.1 Extraction-based Query Producer

Extraction-based query producer aims to extract text spans from the dialogue context $\mathcal{D}_{<t}$ as queries. We use a pre-trained language model (`PLM`) as its backbone and add a linear layer with the softmax activation (`MLP-Softmax`) as the output layer to predict the probability distribution $\mathbf{P}$ over all query candidates $\mathcal{Q} = [q^1, \ldots, q^{|\mathcal{Q}|}]$:

$$\begin{aligned} \mathbf{P} &= \text{MLP-Softmax}([\mathbf{H}^{q^1}, ..., \mathbf{H}^{q^{|\mathcal{Q}|}}]), \\ \mathbf{H}^{q^i} &= \text{MeanPooling}(\mathbf{H}_{beg_i:end_i}), \\ \mathbf{H} &= \text{PLM}(\mathcal{D}_{<t}), \end{aligned} \tag{1}$$

where $\mathbf{H}$ represents the contextualized embeddings produced by `PLM`, and beg_i and end_i are the begin and end indices for the i-th candidate span in $\mathcal{D}_{<t}$. Each candidate query q^i is a continuous span in a turn of $\mathcal{D}_{<t}$. We use MeanPooling over the contextualized embeddings of its tokens from beg_i to end_i to get its representation $\mathbf{H}^{q^i}$.

2.1.2 Generation-based Query Producer

Different from the extraction-based model, this generation-based model adopts a seq2seq architecture to construct search queries from scratch. It can produce queries that are not contained in $\mathcal{D}_{<t}$ at the cost of a larger search space. We adopt a pre-trained encoder-decoder model (denoted as `PGM`) to generate queries in an auto-regressive manner, and beam search is adopted during decoding to produce multiple queries at the same time (Meng et al.,

[1] Code will be released upon acceptance.

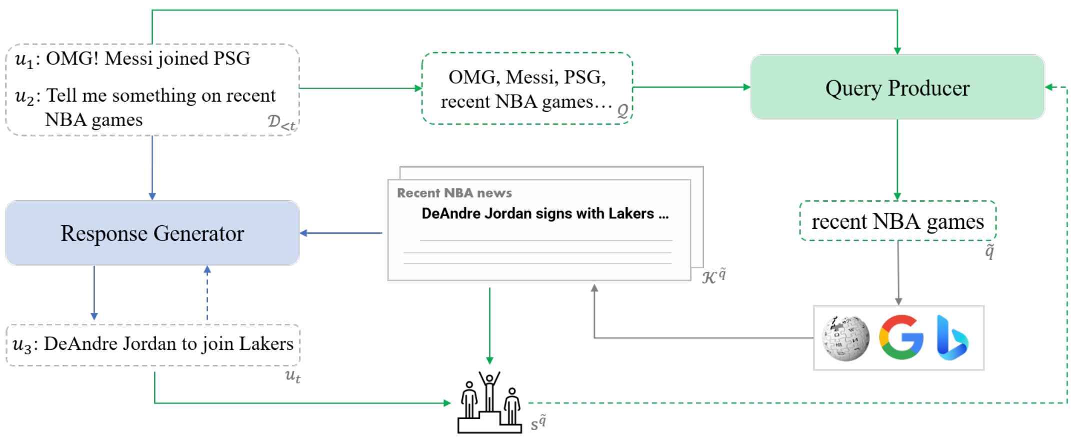

Figure 2: The training process using the example in Fig. 1, where solid lines (→) and dashed lines (⇢) indicate forward and backward pass. **First** (→), input utterances $\mathcal{D}_{<t}$ and (optional) query candidates $\mathcal{Q}$ are fed into the query producer to get search query $\tilde{q}$, and then (→) relevant articles $\mathcal{K}^{\tilde{q}}$ are retrieved from a search engine with $\tilde{q}$. **Next** (→), the response generator constructs u_t given both $\mathcal{D}_{<t}$ and $\mathcal{K}^{\tilde{q}}$. **Finally**, both the query producer and the response generator are trained by the corresponding signals.

2017). The score s_i for a query q^i is the sum of the log probabilities for its tokens over the whole vocabulary:

$$s_i = \frac{\sum_{j=1}^{|q^i|} \log \text{MLP-Softmax}(\mathbf{H}_j^{q^i})}{\sqrt{|q^i|}}, \quad (2)$$
$$\mathbf{H}_j^{q^i} = \text{PGM}(\mathcal{D}_{<t}, q^i_{<j}),$$

where $\mathbf{H}_j^{q^i}$ is the decoder state of the j-th step for query q_i, and $\sqrt{|q_i|}$ is the length-based normalization item to ease the preference of short candidates (Wu et al., 2016).

2.1.3 Pruning Query Search Space

Querying a search engine can be time consuming for training a query producer, as the training process can take hundreds of thousands of steps, and each query can take more than 0.1 seconds. A natural solution for this issue is to create an offline cache of articles for all possible queries before the actual training. However, both extraction-based and generation-based models take a large search space of candidate queries. Given a dialogue of m turns with n words for each turn, there are $\mathcal{O}(m \cdot n^2)$ possible queries for the extraction-based model, while the number is exponential to average query length for the generation-based model.

We study different methods to prune the search space for query production, so that an offline cache can be efficiently established, while the coverage of the pruned space is still large enough. In particular, we explore the two main directions in the task of keyword acquisition (Siddiqi and Sharan, 2015).

- *Dictionary-based*: Typical methods in this direction (Ferragina and Scaiella, 2010) consider the overlap between each dialogue context and a predefined taxonomy as the search space, where the taxonomy is constructed from a large knowledge source (e.g. Wikipedia).

- *Metric-based*: Approaches in this direction (Rose et al., 2010; Campos et al., 2020) extract keywords from a dialogue context based on metric scores (e.g., TF-IDF) without using any vocabulary, and then they merge adjacent keywords into larger spans by heuristic rules.

2.1.4 Training with Cheap Noisy Supervision

We leverage a *cheap noisy supervision* signal to train our query producers, which makes it easier to transfer to other domains and search engines compared with using human annotations (Komeili et al., 2021). The whole training process contains *pre-training with cross-entropy loss* and *reinforcement learning fine-tuning*. The reinforcement learning fine-tuning directly uses the supervision signals as reward, while the pre-training uses the signals as gold labels.

Cheap noisy supervision for query scoring We design a function f that leverages the corresponding gold response u as cheap noisy supervision to assign a score s_q for each query q to indicate its quality. In particular, the function f compares the corresponding top articles $\mathcal{K}^q = \{k_1^q, k_2^q, \ldots\}$ retrieved by q with the gold response u for calculating score s^q:

$$s^q = f(\mathcal{K}^q, u). \quad (3)$$

We consider this as a type of *cheap* supervision because the function f *does not* require extra annotations (e.g., the annotations of gold queries). We study different approaches and choose the popular BM25 metric (Robertson and Walker, 1994) to implement f. More specifically, it first calculates the score for each article by $s_i^q = \text{BM25}(k_i^q, u)$, before determining the overall score s^q as the maximum among them: $s^q = \max(\{s_1^q, s_2^q, \ldots\})$.

We introduce two pre-processing methods for improving upon the vanilla BM25. The first method adopts coreference resolution, which finds the actual entity referred by a pronoun. We then expand response u by concatenating it with the entity mentions referred by its pronouns. This is important as coreference frequently exists in human conversations. The second method drops function words from both articles $\mathcal{K}$ and response u before passing them to the noisy supervision function f. This makes f focus more on content words.

Pre-training with noisy labels At this stage, we take the query with the highest score s^q by function f (Eq. 3) from query candidates $\mathcal{Q}$ as pseudo ground-truth to train both extraction-based and generation-based producers with the standard cross-entropy loss:

$$\mathcal{L}_{ext.}^{pt} = -\log P(\bar{q}|\mathcal{D}_{<t}, \theta_{ext.}), \quad (4)$$

$$\mathcal{L}_{gen.}^{pt} = -\sum_{i=1}^{|\bar{q}|} \log P(\bar{q}_i|\mathcal{D}_{<t}, \bar{q}_{<i}, \theta_{gen.}), \quad (5)$$

where $\bar{q}$ denotes the pseudo ground-truth, $\mathcal{L}_{ext.}^{pt}$ and $\mathcal{L}_{gen.}^{pt}$ are loss terms for extraction-based and generation-based models respectively, and $\theta_{ext.}$ and $\theta_{gen.}$ are the parameters for the models.

Reinforcement learning fine-tuning At fine-tuning stage, we adopt the REINFORCE algorithm (Williams, 1992) with the cheap noisy supervision f as the reward. We subtract a baseline value, which is set to the reward of the candidate query with the highest model score (calculated by Eq. 1 or 2) from f to reduce variance. As BM25 scores are not bounded, we further normalize them to reduce training variance. For each dialog turn with multiple query candidates, we rescale the reward r_i for the i-th candidate as $\frac{r_i - min}{max - min} - 0.5$ with the minimum (min) and maximum (max) values within the candidates. The losses for both producers at fine-tuning stage are defined as:

$$\mathcal{L}^{ft} = -\Delta(r_s, r_b) \log p_s, \quad (6)$$

where p_s is the probability of a candidate query sampled from the model output distribution, r_s and r_b are the rescaled rewards for the sampled and the baseline candidates, respectively.

2.2 Response Generation

After retrieving relevant articles, the next step of our model is to generate a proper response using the articles and the dialogue context. We implement response generators, Rank-Gen and Merge-Gen, based on two representative research directions. Both models use different strategies to leverage the retrieved articles, and thus we can better study the robustness of our query producer.

2.2.1 Rank-Gen

Rank-Gen takes an explicit ranker to choose one piece from a set of articles (Lian et al., 2019; Zhao et al., 2020). There are several benefits of this direction, such as improving the explainability and the ability of handling large knowledge set. The ranker first selects a piece of knowledge $\tilde{k}$ from candidates $\mathcal{K}$, then the seq2seq-based generator predicts the response given the dialogue context $\mathcal{D}_{<t}$ and selected knowledge $\tilde{k}$:

$$\begin{aligned} \tilde{k} &= \text{argmax}_{k \in \mathcal{K}} \text{Ranker}(\mathcal{D}_{<t}, k), \\ u_t &= \text{Generator}(\mathcal{D}_{<t}, \tilde{k}). \end{aligned} \quad (7)$$

We adopt reinforcement learning to jointly train the ranker and generator, where the ranker is guided by the signal from the generator via policy gradient, and the generator is trained by cross-entropy loss taking sampled knowledge $\tilde{k}_s$ from the ranker:

$$\mathcal{L}_{RG} = \mathcal{L}_{rank} + \mathcal{L}_{gen}, \quad (8)$$

$$\mathcal{L}_{rank} = -(\mathcal{L}_{gen}^{\tilde{k}_b} - \mathcal{L}_{gen}^{\tilde{k}_s}) \log P(\tilde{k}_s|\mathcal{D}_{<t}, \mathcal{K}), \quad (9)$$

$$\mathcal{L}_{gen} = -\sum_{i=1}^{|u_t|} \log(u_{t,i}|u_{t,<i}, \mathcal{D}_{<t}, \tilde{k}_s), \quad (10)$$

where $\tilde{k}_b$ is the baseline knowledge to reduce variance, and $\mathcal{L}_{gen}^{x}(x \in \{\tilde{k}_b, \tilde{k}_s\})$ is the generation loss taking the corresponding knowledge as extra input.

Before joint training, we also introduce a warm up stage following Zhao et al. (2020), where the ranker is trained with cross-entropy loss on the pseudo ground-truth knowledge $\bar{k}$ that has the highest BM25 score among knowledge candidates, and the generator is also trained with cross-entropy loss taking $\bar{k}$ as the additional input:

$$\bar{k} = \text{argmax}_{k \in \mathcal{K}} \text{BM25}(\mathcal{D}_{<t}, \mathcal{K}), \tag{11}$$

$$\mathcal{L}_{rank}^{pt} = -\log P(\bar{k}|\mathcal{D}_{<t}, \mathcal{K}), \tag{12}$$

$$\mathcal{L}_{gen}^{pt} = -\sum_{i=1}^{|u_t|} \log(u_{t,i}|u_{t,<i}, \mathcal{D}_{<t}, \bar{k}). \tag{13}$$

2.2.2 Merge-Gen

Merge-Gen follows another popular direction (Izacard and Grave, 2021) by compressing and consuming all input knowledge. Particularly, each knowledge piece k_i in knowledge pool $\mathcal{K}$ is first paired with the dialogue context $\mathcal{D}_{<t}$. Then, these pairs $\{\mathcal{D}_{<t}, k_i\}_{k_i \in \mathcal{K}}$ are compressed into dense vectors independently before being concatenated as inputs to the decoder for response generation:

$$\begin{aligned} u_t &= \text{Decoder}([\mathbf{H}_1; \mathbf{H}_2; ...; \mathbf{H}_{|\mathcal{K}|}]), \\ \mathbf{H}_i &= \text{Encoder}(\mathcal{D}_{<t}, k_i). \end{aligned} \tag{14}$$

Comparing with Rank-Gen, Merge-Gen does not suffer from the risk of selecting wrong knowledge by a ranker. However, it lacks explainability and may potentially lose information when compressing input knowledge into dense vectors. The training signal is based on the standard cross-entropy loss over gold response u_t:

$$\mathcal{L}_{MG} = -\sum_{i=1}^{|u_t|} \log(u_{t,i}|u_{t,<i}, \mathcal{D}_{<t}, \mathcal{K}). \tag{15}$$

3 Experiment

We study the effectiveness of our model, especially the usefulness of knowledge retrieval using search queries for response generation.

3.1 Dataset

We choose the Wizard-of-Wikipedia (WoW, Dinan et al. 2019) dataset for evaluation. The dataset is split into 18,430/967/968 for train/dev/test, respectively. For each dialogue, it includes relevant knowledge (e.g., the titles of ground-truth articles) annotated by human. Therefore, we can use WoW to measure the performance of query production by comparing retrieved knowledge and ground-truth knowledge. We use its *unseen* test set for evaluation. We remove the first turn of each dialogue, because the first turn reveals the title of the Wikipedia article for discussion, which will expose the main topic of the dialogue.

3.2 Setting

We choose the hyperparameters by following previous work or development experiments.

Query production We take an ELECTRA-base (Clark et al., 2020) model[2] and a BART-base (Lewis et al., 2020) model[3] as the backbones for our extraction and generation-based query producers, respectively. We use AdamW (Loshchilov and Hutter, 2019) as the optimizer with learning rate 1e-5. The batch size is set to 64. The extraction-based producer is pre-trained for 1 epoch, while the generation-based producer is pre-trained for 5 epochs. To prune the search space of query production, we adopt two keyword acquisition tools, TagMe (dictionary-based) and YAKE! (metric-based). We use recall, denoted as R@x ($x \in \{1, 3, 5\}$), which compares the top x retrieved candidates with ground-truth knowledge to evaluate the performance of query producers.

Response generation Both Rank-Gen and Merge-Gen use a BART-base model for response generation. All models are trained using AdamW with learning rate 1e-5 and batch size 64. The warm-up stage for ranker in Rank-Gen takes 2 epoch. We perform early stopping based on the perplexity (PPL) on the development set. Following previous work, We adopt PPL and Unigram F1 to evaluate response generation.

Search engine As most commercial search engines are not publicly free, we adopt Wikipedia search.[4] We retain the **top 5** retrieved Wikipedia articles of each query for evaluation. The summary of each article (the first paragraph for a Wikipedia article) is extracted as external knowledge.

3.3 Development Experiments

We explore the design choices for query space pruning (§2.1.3) and the scoring function f (Eq. 3), as they determine the quality of query production, which in turn affects response generation.

Different choices of space pruning and query scoring algorithms Table 1 shows the development results of several popular query scoring algorithms with *TagMe* and *YAKE!* for search space pruning. We consider the following scoring algorithms:

[2] https://huggingface.co/google/electra-base-discriminator
[3] https://huggingface.co/facebook/bart-base
[4] https://en.wikipedia.org/wiki/Special:Search

Pruning	Query Scoring	R@1	R@3	R@5
TagMe	Random	12.55	31.27	44.19
	TF-IDF	39.30	61.28	67.26
	BM25(q,u)	36.09	58.73	65.89
	BM25	53.36	65.25	69.46
	BM25$_{++}$	**60.59**	**69.81**	**72.49**
YAKE!	Random	14.21	33.96	46.00
	TF-IDF	36.92	58.63	64.78
	BM25(q,u)	28.01	52.94	62.59
	BM25	50.70	65.32	69.91
	BM25$_{++}$	57.97	69.15	72.03

Table 1: Development results of various search-space pruning methods and query scoring algorithms.

- *Random*: It randomly picks a query from the candidate pool.
- *TF-IDF*: It averages the TF-IDF scores of all words within each candidate query as its ranking score. This algorithm *only considers the query information*.
- *BM25(q,u)*: It measures the similarity between q and u using BM25 without considering the actual retrieved knowledge by q.
- *BM25*: It is our proposed scoring function f (Eq. 3) with standard BM25.
- *BM25$_{++}$*: It is also based on f using BM25 but equipped with pre-processing methods: coreference resolution and function words dropping.

Regarding search-space pruning, the average candidate number and the ceiling performance (R@M in Fig. 3) using TagMe are 17.45 and 75.47%, respectively, while the corresponding numbers are 21.64 and 75.04% for YAKE!. **First**, the upper bound does not reach 100% because: (1) the pruning method fails to keep some good search queries; (2) some dialogue turns (4.7%) do not require any external knowledge; (3) speakers change the topics in some turns, which requires queries that are not contained in the dialogue context. Overall, we get a decent number of around 75%. **Second**, most ranking algorithms using TagMe outperform their corresponding ones using YAKE!. Besides, TagMe reaches higher upper bound (75.47% vs 75.04%) with less candidates (17.45 vs 21.64) than YAKE!. Based on the results, we choose TagMe for query space pruning in further experiments.

Regarding query scoring, BM25$_{++}$ outperforms all other algorithms, demonstrating the effectiveness of coreference resolution and function words dropping. BM25 is the second best method, which shows that the retrieved articles provide more information beyond the query and the response. We choose BM25$_{++}$ for future experiments.

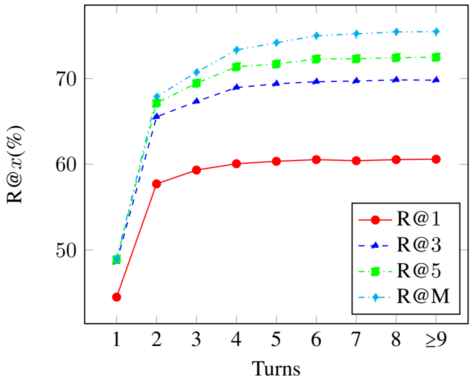

Figure 3: Development results of BM25$_{++}$ and the ceiling performances (R@M) given keyword candidates from the last k turns.

The number of dialogue turns for obtaining candidate queries With the pruning method and query scoring algorithm determined, the next step is to choose the number (k) of turns for obtaining candidate queries. Intuitively, considering more turns will increase the ceiling performance on knowledge retrieval with extra noise on the query scoring algorithm. As shown in Fig. 3, the performance of BM25$_{++}$ consistently improves with the increase of k. This demonstrates that the benefit of considering longer dialogue context for candidate queries exceeds the cost (extra noise). Therefore, we choose to consider all turns for the remaining experiments.

3.4 Main Results

Table 2 shows the main testing results including the performance on search query production and response generation. We compared our models with typical baselines with different query acquisition techniques: (1) no external knowledge is used (line 1); (2) using all search queries extracted from the last k turns[5] (line 2-4); (3) using search queries produced by different techniques (line 5-7).

We can draw the following conclusions: **First**, models leveraging external knowledge perform better than the baseline (line 1) without using external knowledge, verifying that using retrieved knowledge is generally helpful for response generation. Merge-Gen based models surpass all Rank-Gen based ones, as it avoids the error propagation from the ranker. This demonstrates the effectiveness of

[5]They are based on the heuristic that people tend to keep talking the topics just mentioned in the last few turns.

Line Num.	Query Production Method	Avg. Num. of Querying	Query Ranking			Rank-Gen		Merge-Gen	
			R@1	R@3	R@5	PPL↓	Uni. F1	PPL↓	Uni. F1
1	None	None	–	–	–	25.26	16.53	25.13	16.64
2	All from last 2 turns	8.29	–	–	–	22.77	17.43	20.04	17.55
3	All from last 4 turns	13.38	–	–	–	22.86	17.38	19.89	17.72
4	All from all history turns	17.45	–	–	–	23.03	17.32	**19.79**	17.71
5	TF-IDF	**1**	43.41	61.63	66.65	22.86	17.28	21.53	17.64
6	Extraction-based	**1**	**62.41**	**72.91**	**74.87**	**21.60**	**17.81**	20.20	**18.15**
7	Generation-based	**1**	56.77	66.08	68.22	21.65	17.51	20.69	17.95

Table 2: Main results of query production and response generation on WoW unseen testset, where "PPL↓" and "Uni. F1" indicates perplexity and unigram F1, respectively.

System	R@1	R@3	R@5
Extraction-based	62.41	72.91	74.87
w/o pre-train	61.97	71.84	73.77
w/o fine-tune	61.36	73.08	74.94
w/o prune search space	60.65	67.68	69.97
Generation-based	56.77	66.08	68.22
w/o pre-train	38.14	54.91	59.83
w/o fine-tune	51.91	65.82	69.75
w/ prune search space	60.67	71.55	73.52

Table 3: Ablation study on both extraction-based and generation-based query producers.

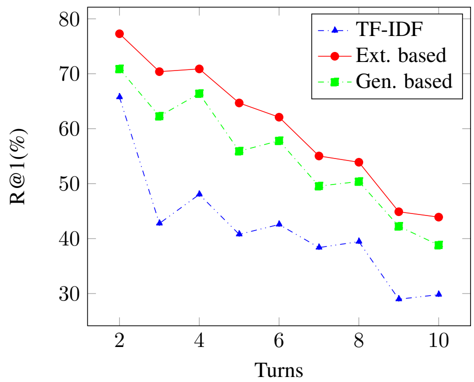

Figure 4: Performance of different query producers at different dialogue turns.

incorporating multiple pieces of knowledge. **Second**, for the baselines using multiple queries (line 2-4), Rank-Gen and Merge-Gen show opposite trends when the number of turns for obtaining queries increases with Merge-Gen being consistently better. This confirms the advantage of Merge-Gen over Rank-Gen by preventing the error propagation from a ranker. However, the time of knowledge gathering (querying a search engine and retrieving pages) also grows linearly with the query number. **Finally**, our models using either of the proposed query producers perform better than all baselines for most situations, indicating that our query producer trained with cheap noisy supervision signals can retrieve useful contents for response generation. The baselines (line 2-4) using multiple queries show slightly better perplexity values than our models when combined with Merge-Gen. But, their knowledge fetching process is at least 8-time slower than ours. Besides, our models still manage to get better Uni. F1 scores with fewer times of search-engine querying.

3.5 Analysis

Ablation study Table 3 shows the ablation study on our query producers. We can draw the following conclusions. **First**, both pre-training with cross-entropy loss and reinforcement learning fine-tuning are helpful for query producers. For extraction-based approach, pre-training (w/o fine-tune) mainly helps the performance on R@3 and R@5, while fine-tuning (w/o pre-train) mostly helps the performance on R@1. In general, fine-tuning provides more robust performances than pre-training, as it can better handle the noisy supervision. For generation-based method, both training stages are very crucial, probably due to its large search space. In this case, pre-training-alone (w/o fine-tune) outperforms the fine-tuning-alone counterpart (w/o pre-train). This is because RL-based fine-tuning from scratch is slow to converge (Paulus et al., 2018; Wang et al., 2018). **Second**, adding search space pruning brings in significant performance gains on both extraction-based and generation-based methods, proving the importance of limiting the search space to high-quality candidate queries.

Performances of query producers at different turns We further compare the R@1 of 3 query producers at various turns. Among them, the *TF-IDF* baseline only takes the information from a query and ignores the retrieved articles, while *Ext. based* and *Gen. based* are our proposed producers based on extraction and generation, respectively. Generally, the last several turns yield more query candidates than the first ones, causing larger search

Model	Query Soundness	Article Knowledge Coverage	Response Naturalness	Response Knowledgeable
BART	–	–	2.39	1.89
w/ query extract	**2.79**	**2.76**	**2.65**	2.39
w/ query generate	2.65	2.59	2.58	**2.44**

Table 4: Human evaluation results.

spaces. As shown in Fig. 4, the performance of all producers drops rapidly when a dialogue continues. Our extraction-based producer consistently outperforms all others, with its performance being roughly 45% at the last turn, which is 15% higher than TF-IDF.

3.6 Human Evaluation

We conduct human evaluation on 100 test samples, and we choose Merge-Gen as the response generator, because it shows better performance than Rank-Gen on automatic metrics. The models are rated regarding both query production and response generation. For query production, we measure **Soundness**, which means whether the query is sound by itself[6], and **Knowledge Coverage**, which means how relevant is the retrieved knowledge. For response generation, we follow previous work to measure **Naturalness**, indicating how fluent and relevant a response is, and **Knowledgeable**, representing how much knowledge is used in a response. We ask 3 annotators capable of fluent English communication to score each aspect with 3-point schema[7], and we average their scores as the final score of the aspect. The inner-annotator agreement (Fleiss' κ) is 0.5461, which is in the moderate level.

As shown in Table 4, our models improve (+0.50 for "w/ query extract" and +0.55 for "w/ query generate" over 3) the BART baseline on the Knowledgeable aspect. We see moderate gains (+0.26 for "w/ query extract" and +0.19 for "w/ query generate" over 3) regarding Naturalness, because BART can already generate fluent replies with large-scale pre-training on text generation. Note that general replies like "*Sorry, I don't know*" are considered natural in certain context like "*Do you know Mike Tyson?*". Generally, we observe positive correlation between query production and response generation, and thus we can expect another improvement on response generation if query production can be further enhanced. We list typical examples from our human study in Appendix.

[6]Sometimes, a sound query may not retrieve good knowledge due to search-engine mistake.

[7]We attach detailed guidelines in Appendix.

4 Related Work

Internet-aided dialogue response generation One related preprint draft in parallel (Komeili et al., 2021) studies using Bing[8] as the knowledge source for dialogue response generation. We both share a similar motivation of using a search engine as the knowledge source. However, Komeili et al. (2021) manually annotate 48K queries to train their query generator. Thus the supervision signals are expensive to obtain and may not be transferable to other domains and search engines. On the other hand, our model is search-engine agnostic and the training signals are cheaper to obtain.

Keyword production As a longstanding task, keyword production was initially proposed to automatically create keywords for articles. Classic techniques (e.g., TF-IDF and TextRank) have been widely used over decades. In the past few years, deep learning has made notable progress on this task. Initially, neural keyword producers (Zhang et al., 2016; Luan et al., 2017) are extraction-based that extract keywords from inputs. Recently, generation-based methods (Meng et al., 2017; Chen et al., 2018, 2019; Meng et al., 2021) using a seq2seq model are gaining popularity. We produce keywords as queries to a search engine and study both extraction-based and generation-based methods on our task in conversational domain.

5 Conclusion

We have introduced a model that leverages a general search engine for knowledge-aided response generation. To effectively interact with the search engine, it adopts a query producer to generate search queries. We design cheap noisy supervision signals to train our query producer, so that no extra human annotation is needed, making our model easily transferable to other search engines and domains. Experimental results under both automatic metrics and human judges show the superiority of our model over a pre-trained BART model and other baselines.

[8]https://www.bing.com/

References

Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. 2020. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*.

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, 509:257–289.

Jun Chen, Xiaoming Zhang, Yu Wu, Zhao Yan, and Zhoujun Li. 2018. Keyphrase generation with correlation constraints. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4057–4066.

Wang Chen, Yifan Gao, Jiani Zhang, Irwin King, and Michael R. Lyu. 2019. Title-guided encoding for keyphrase generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6268–6275.

Xiuyi Chen, Fandong Meng, Peng Li, Feilong Chen, Shuang Xu, Bo Xu, and Jie Zhou. 2020. Bridging the gap between prior and posterior knowledge selection for knowledge-grounded dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3426–3437.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *International Conference on Learning Representations*.

Paolo Ferragina and Ugo Scaiella. 2010. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 1625–1628.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Gautier Izacard and Édouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880.

Byeongchang Kim, Jaewoo Ahn, and Gunhee Kim. 2020. Sequential latent knowledge selection for knowledge-grounded dialogue. In *International Conference on Learning Representations*.

Mojtaba Komeili, Kurt Shuster, and Jason Weston. 2021. Internet-augmented dialogue generation. *arXiv preprint arXiv:2107.07566*.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Rongzhong Lian, Min Xie, Fan Wang, Jinhua Peng, and Hua Wu. 2019. Learning to select knowledge for response generation in dialog systems. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5081–5087. International Joint Conferences on Artificial Intelligence Organization.

Shuman Liu, Hongshen Chen, Zhaochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1489–1498.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Yi Luan, Mari Ostendorf, and Hannaneh Hajishirzi. 2017. Scientific information extraction with semi-supervised neural tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2641–2651.

Rui Meng, Xingdi Yuan, Tong Wang, Sanqiang Zhao, Adam Trischler, and Daqing He. 2021. An empirical study on neural keyphrase generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4985–5007, Online. Association for Computational Linguistics.

Rui Meng, Sanqiang Zhao, Shuguang Han, Daqing He, Peter Brusilovsky, and Yu Chi. 2017. Deep keyphrase generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 582–592, Vancouver, Canada. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*.

Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR'94*, pages 232–241. Springer.

Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Eric Michael Smith, Y-Lan Boureau, et al. 2021. Recipes for building an open-domain chatbot. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 300–325.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1:1–20.

Sifatullah Siddiqi and Aditi Sharan. 2015. Keyword and keyphrase extraction techniques: a literature review. *International Journal of Computer Applications*, 109(2).

Zhiliang Tian, Wei Bi, Dongkyu Lee, Lanqing Xue, Yiping Song, Xiaojiang Liu, and Nevin L Zhang. 2020. Response-anticipated memory for on-demand knowledge integration in response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 650–659.

Li Wang, Junlin Yao, Yunzhe Tao, Li Zhong, Wei Liu, and Qiang Du. 2018. A reinforced topic-aware convolutional sequence-to-sequence model for abstractive text summarization. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4453–4460.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256.

Wenquan Wu, Zhen Guo, Xiangyang Zhou, Hua Wu, Xiyuan Zhang, Rongzhong Lian, and Haifeng Wang. 2019. Proactive human-machine conversation with explicit conversation goal. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3794–3804.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

Qi Zhang, Yang Wang, Yeyun Gong, and Xuan-Jing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 836–845.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and William B Dolan. 2020. Dialogpt: Large-scale generative pre-training for conversational response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 270–278.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390.

Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. 2020. Kdconv: A chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7098–7108.

#1	
Context	**A:** One of my favorite bands despite usual protests is Nickelback. How about you or who is one of your favorites? **B:** I like Nickelback too. They are my favorite Canadian rock band and I've liked them since 1995 they started. **A:** Speaking of Canadian rock bands, my absolute all time favorite band is Rush. I have every album by them!
Ref response	Yes Rush! They formed in 1968 and took their first United States tour in 1974.
Vanilla	
BART	Rush is one of my favorite bands too. They are from Hanna, Alberta.
TF-IDF	
Seach query	Nickelback
Retrieved passages	Nickelback; Rockstar (Nickelback song); Nickelback discography; Chad Kroeger; Dark Horse (Nickelback album)
Knowledge fragment	...Nickelback is one of the most commercially successful Canadian rock bands, having sold more than 50 million albums worldwide...
Rank-Gen	Rush is one of my favorite bands too. They have sold more than 50 million albums worldwide.
Merge-Gen	Rush is one of my all time favorite bands. They have sold more than 50 million albums worldwide.
Extraction-based	
Seach query	Rush
Retrieved passages	Rush (Rush album); Into the Rush; Rush (band)
Knowledge fragment	...Rush was a Canadian rock band formed in Toronto in 1968, consisting of Geddy Lee (bass, vocals, keyboards, composer), Alex Lifeson (guitars, composer), and Neil Peart (drums, percussion, lyricist)...
Rank-Gen	Rush is one of my favorite bands too. They were formed in 1968 in Toronto.
Merge-Gen	Rush was formed in 1968 by Geddy Lee, Alex Lifeson, and Neil Peart.
#2	
Context	**A:** Do you like Elvis Presley? he was a famous american singer **B:** He is pretty good, I am from Nashville, So i've even been to graceland.
Ref response	Presley was born in Tupelo, Mississippi, and then to memphis
Vanilla	
BART	He was born and raised in Nashville, TN
TF-IDF	
Seach query	graceland
Retrieved passages	Graceland; Graceland (album); Graceland (TV series); Graceland University
Knowledge fragment	...Graceland is located at 3764 Elvis Presley Boulevard in the vast White haven community, about nine miles (14 kilometers) from Downtown and less than four miles (6.4 km) north of the Mississippi border.It was opened to the public as a museum on June 7, 1982. The site was listed in...
Rank-Gen	Yes, it was opened to the public as a museum in 1982
Merge-Gen	I've been to Graceland a few times. It was opened to the public as a museum in 1982
Extraction-based	
Seach query	Elvis Presley
Retrieved passages	Elvis Presley; Elvis Presley (album); Priscilla Presley; Elvis Presley albums discography; Lisa Marie Presley
Knowledge fragment	...Presley was born in Tupelo, Mississippi, and relocated to Memphis, Tennessee with his family when he was 13...
Rank-Gen	He was born in Tupelo, Mississippi and moved to Memphis when he was 13
Merge-Gen	He was born in nashville and moved to Memphis when he was 13

Table 5: Two examples on query production and response generation from WoW test set.

A Annotation Guidelines

All aspects are based on a 3-point scheme: 3 means flawless; 2 means containing minor flaw; 1 means having major flaw but with values; 0 means being completely wrong.

Query Soundness It considers if the selected topic is active (the one being discussed).

- The score can be 3 if the active topic is selected, otherwise the score can be 2, 1 or 0 depends on how close the selected topic is to the active one.
- If the active topic (e.g., "plants vs zombie") is emerged from a parent topic (e.g., "zombie"), the score can be 2 if the parent topic is chosen.

Article Knowledge Coverage It measures how relevant (and useful) are the retrieved articles regardless of the query (sometimes a bad query can yield good articles).

- If the article talks about something (e.g., guitars) close to the dialogue topic (e.g., a guitarist), then the score can be 2.
- If the article is slightly relevant to the dialogue topic (e.g., a musician or an album), the score can be 1.
- The score can be 0 if no article is retrieved (sometimes this is due to bad queries).

Naturalness How sound a reply is to the dialogue context. A sound reply should be consistent both in purpose and in topic to the context. But it does not reflect the knowledge aspect.

- If there is a question like "Do you like ...?", a sound reply should contain something like "Yes...", "No, I don't..." or "I do..."

Knowledgeable A knowledgeable reply should contain new stuff, so examples like "Oh, that's cool!" is not knowledgeable. In this situation, scores can range from 0 to 1, where 1 can be chosen if the reply actually does not require knowledge.

Besides, knowledgeable replies should not violate factoid statements in both dialogue context and in retrieved knowledge. For instance, if the context mentions "the band sold 500 million albums worldwide", it is not knowledgeable if the reply says "the band sold 400 million albums worldwide".

- For replies that violate existing factoid statements, the score can be 1.
- For replies that cannot be determined true or false given dialogue context and retrieved knowledge, the score can be 2.
- For replies that can be found true given dialogue context and retrieved knowledge, the score can be 3.