UGent-T2K at the 2nd DialDoc Shared Task: A Retrieval-Focused Dialog System Grounded in Multiple Documents

Yiwei Jiang[*], **Amir Hadifar**[*], **Johannes Deleu, Thomas Demeester, Chris Develder**
Ghent University – imec, IDLab
Ghent, Belgium
first_name.last_name@ugent.be

Abstract

This work presents the contribution from the Text-to-Knowledge team of Ghent University (UGent-T2K)[1] to the MultiDoc2Dial shared task on modeling dialogs grounded in multiple documents. We propose a pipeline system, comprising (1) document retrieval, (2) passage retrieval, and (3) response generation. We engineered these individual components mainly by, for (1)-(2), combining multiple ranking models and adding a final LambdaMART reranker, and, for (3), by adopting a Fusion-in-Decoder (FiD) model. We thus significantly boost the baseline system's performance (over +10 points for both F1 and SacreBLEU). Further, error analysis reveals two major failure cases, to be addressed in future work: (i) in case of topic shift within the dialog, retrieval often fails to select the correct grounding document(s), and (ii) generation sometimes fails to use the correctly retrieved grounding passage. Our code is released at this link.

1 Introduction

Most prior research on document-grounded dialog systems assumes a single document for each dialog (Choi et al., 2018; Reddy et al., 2019; Feng et al., 2020). There are relatively few works on Multi-Document Grounded (MDG) dialog modeling, which requires a dialog system to (i) retrieve grounded passages (or documents) given the user question, and then (ii) generate responses based on the retrieval results and dialog context. Real-world applications (e.g., administration question answering, travel booking assistance and procedural task guidance) for MDG are challenging because of more complex user behavior in such dialogs on diverse information sources. In particular, for (i) retrieval of grounding text passage(s) the challenges pertain to keeping track of dialog state, topic shift (e.g., switching from driving license requirements to car insurance), vocabulary mismatch, vague question formulation, etc. Furthermore, (ii) response generation needs to appropriately phrase the answer to fit in a human(-like) dialog rather than simply copying a source document snippet.

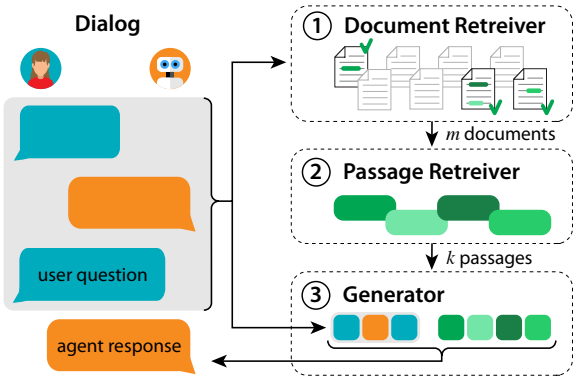

Figure 1: Our proposed pipeline dialog system.

We leverage the recently released dialog dataset, MultiDoc2Dial (Feng et al., 2021), to tackle aforementioned challenges. We build a pipeline system (Fig. 1) comprising (1) a document retriever, (2) a passage retriever, and (3) an answer generator fusing multiple grounding input passages. Given the dialog context (i.e., the dialog history and user question), a document retriever searches given supporting documents to select the top-m related ones. Subsequently, these full documents are segmented into shorter passages ranked by a passage retriever. For these retrieval components (1)-(2), we use an ensemble approach — combining BM25, cosine similarity, etc.; for passage retrieval, we included Dense Passage Retrieval (DPR; Karpukhin et al., 2020) — followed by a reranking step using LambaMART (Burges, 2010). The top-k passages are fused with the dialog context by a response generator to produce knowledge-grounded responses, based on Fusion in Decoder (FiD; Izacard and

[*] These authors contributed equally to this work.
[1] https://ugentt2k.gitlab.io

Proceedings of the Second DialDoc Workshop on Document-grounded Dialogue and Conversational Question Answering, pages 115 - 122

Grave, 2021).

We contribute with: (i) a multi-stage pipeline system comprising first the grounding text retrieval stages, split further into document and subsequent passage retrieval components (both using a multi- feature ensemble system), and second an answer generation model fusing information from multiple passages; (ii) experiments demonstrating that our pipeline system outperforms the baseline method by a large margin (over +10 points for both F1 and SacreBLEU); (iii) insightful error analysis, suggesting that the main shortcomings of the current system include failures of (a) the *retrieval* stages in case of topic shifts by the user, and (b) the answer *generation* stage to identify the correct grounding passage among its inputs. Our codes are released at https://github.com/YiweiJiang2015/ugent-t2k-dialdoc

2 Task Definition

The MultiDoc2Dial shared task comprises two subtasks: in the *seen-domain* (referenced by subscript S), the system can rely on training data comprising both exemplary dialogs as well as the corresponding document set from the domains it will be tested on, whereas in the *unseen-domain* (referenced by U) no related dialogs nor documents have been seen by the system before.

In general, for both subtasks, a system first retrieves relevant documents from a document pool (D_S or D_U) given the dialog context, i.e., a user's question Q_i (i is the turn number) and the full conversation history $Q_{<i}$. Current state-of-the-art solutions split long documents into passages (P_S or P_U) to facilitate more fine-grained location of the grounding information. Second, the grounding information G (span(s) or passage(s)) for Q_i has to be identified within the passages of retrieved documents. The MultiDoc2Dial dataset was curated such that G for each question can be exactly found within one document, while the full dialog's answers jointly may span multiple documents, thus requiring a model to decide when to switch to a different document. (Note that, depending on how exactly a document is split into shorter passages, G may extend over more than one passage.) Third, a generation model takes as input G and $Q_{\leq i}$ to generate responses whose meaningfulness and coherence are rated using automatic metrics (i.e., F1_U, SacreBLEU, Rouge-L and Meteor).

3 Model

The next subsections detail the aforementioned components (1)-(3) of our pipeline system.

3.1 Document Retrieval

The input to our document retrieval model is a user's dialog question Q and the output is a set of m documents $\{d_1, d_2, \ldots, d_m\}$ selected from the document pool D. For each question, we rank all the documents by computing the similarity scores. We utilize various scoring modules as input to the LambdaMART reranker (Burges, 2010). Our scoring modules include: (i) different BM25 (Trotman et al., 2014) configurations, (ii) cosine similarity between dense representations on both word-level and passage-level, and (iii) off-the-shelf term-matching techniques provided by Terrier (Macdonald et al., 2012).

3.2 Passage Retrieval

Given the top-m documents returned by the document retriever, a passage retriever ranks passages belonging to these documents. More specifically, we follow the baseline's segmentation of a document into passages, ensuring a fair performance comparison between our passage retriever and the baseline. The same scoring modules for the document retrieval are applied on the passage level, with additional similarity features computed by DPR (Karpukhin et al., 2020)

3.3 Response Generation

We choose FiD (Izacard and Grave, 2021) as our generation model, which can be trained independently from the retrieval module. FiD was originally proposed for the open-book question answering problem (Kwiatkowski et al., 2019; Joshi et al., 2017) and showed great power in incorporating knowledge from multiple passages. It is built on top of a transformer-based seq2seq model. We employ BART (Lewis et al., 2020a) as the pretrained weights of FiD instead of T5 as in (Izacard and Grave, 2021), since fine-tuning BART is computationally more affordable in our case. The FiD's encoder takes as input a question and a list of top-k ranked passages formatted as $((Q, P_1), (Q, P_2)...(Q, P_k))$. Each pair (Q, P) is encoded individually. Concatenation of the k encodings is used as the memory accessed by the decoder for the cross-attention operation. The train-

ing objective is the cross-entropy loss between generated sequences and gold responses.

4 Experiments and Analysis

4.1 Dataset

We evaluate our pipeline system on the MultiDoc2Dial dataset, containing 4,796 conversations grounded in 488 documents. In the dialog data, each conversation covers at least one topic from four domains (see Appendix B.2). It is challenging to retrieve the grounding information when users shift their topic (i.e., implicitly referring to another document) during a dialog. In total, there are 61,078 turns, including 29,746 user questions that are split into 21,451, 4,201 and 4,094 for train, dev and test sets respectively.

4.2 Baseline System

The baseline system uses the Retrieval Augmented Generation (RAG; Lewis et al., 2020b) model composed of two neural modules: DPR for passage retrieval and BART for response generation. First, a pre-trained DPR is finetuned on the passage retrieval task built from MultiDoc2Dial dataset. Second, RAG is finetuned to generate responses for MultiDoc2Dial dialogs by inserting the finetuned DPR weights and freezing DPR's context encoder.

4.3 Retrieval and Generation Results

We present experiment results for the document retriever, passage retriever and generator separately. Ablation studies of the document retriever focus on analysing the contributions of different features. We validate the effectiveness of first using the document retriever, to boost the passage retriever's performance. Results of response generation experiments show that there is an optimal number of passages input to the FiD model. We also discuss FiD's inefficiency in recognizing grounding knowledge among multiple passage inputs.

4.3.1 Retriever

Document retrieval — Table 1 presents our results for the document retrieval. The first row shows a simple BM25 with the same configuration as official baseline but on document level. $BM25_{tuned}$ indicates BM25 with additional preprocessing and postprocessing over its input features and output rankings (see Section B.1 for details). $BM25_{title}$ is another BM25, solely trained on document titles and subtitles. The reason for this choice is to distinguish the importance of the title words from other words, as the title provides a strong signal for document retrieval. In addition, to capture semantic relatedness and to address the word-mismatch problem between questions and documents, we compute word-level and passage-level embeddings to retrieve relevant documents. For word-level (denoted by 'Word emb.'), we simply average word vectors to obtain question and document representations, then using TF-IDF weighting and principal component removal (Arora et al., 2017) followed by cosine similarity. For passage-level (denoted by 'Passage emb.'), we use a pre-trained model[2] to embed a document's passages and use the highest passage score to rank the document. Macdonald et al. (2012) offer various term-matching approaches for text retrieval. The best performing model in our experiment is DPH (Amati, 2006).

Model	R@1	R@5	R@10	R@25
BM25 (baseline)	46.6	67.7	74.3	82.3
$BM25_{tuned}$	57.8	84.2	89.6	95.8
$BM25_{title}$	46.5	73.2	82.2	91.6
Word emb.	36.4	60.4	70.1	82.9
Passage emb.	34.9	61.9	71.3	84.5
DPH	49.3	77.2	85.9	94.1
$BM25_{tuned}$				
+ $BM25_{title}$	62.0	87.8	93.0	97.3
+ Terrier	62.5	88.9	93.5	97.5
+ embeddings	66.3	91.1	95.2	97.9
LambdaMART	65.9	92.3	96.1	98.7

Table 1: Recall scores for document retrieval on dev set.

In the second block of Table 1, we combine various ranking methods in an ensemble using rank fusion, simply summing the various scores.

We first aggregate scores from $BM25_{tuned}$ and $BM25_{title}$. The next row presents adding the combination of 13 term-matching techniques borrowed from the Terrier IR framework.[3] Finally, we add the embedding scores into the ensemble model, which significantly boosts the performance (increasing R@1 from 62.5 to 66.3), indicating the complementarity of the various ranking criteria. Finally, instead of naively summing all scores, we employ the LambdaMART algorithm, which yields the highest recall scores (except for R@1).

Passage retrieval — Table 2 compares our passage

[2] msmarco-bert-base-dot-v5: available at https://bit.ly/3ID92fF

[3] http://terrier.org/ — Note that due to our limited time budget for the challenge, we did not properly analyze the contribution of the various Terrier features; therefore some of them may be unnecessary.

Model	m	R@1	R@5	R@10	R@15
BM25 (baseline)	488	19.6	41.9	50.8	-
DPR (baseline)	488	49.0	72.3	80.0	-
$\text{DPR}_{\text{top1 doc}}$	1	55.6	71.6	73.0	73.1
$\text{DPR}_{\text{top5 docs}}$	5	49.2	72.0	80.6	84.1
$\text{DPR}_{\text{top10 docs}}$	10	47.8	69.8	77.9	82.4
$\text{DPR}_{\text{top30 docs}}$	30	46.6	67.8	75.3	80.1
LambdaMART	30	57.0	82.1	88.3	91.4

Table 2: Recall scores for passage retrieval on dev set. Baseline scores come from (Feng et al., 2021). m denotes the number of top documents that are used for passage retrieval.

retrieval results to that of the baseline (Feng et al., 2021). To validate whether the document retrieval stage helps to limit the search space of passage retrieval, we perform a simple test that uses DPR to only rank passages from the top-m documents. Restricting the DPR to retrieve passages only from the top-1 document increases R@1 from 49.0 to 55.6 while it hurts R@10 (dropping from 80.0 to 73.0). By increasing m, R@10 improves at the cost of lowering R@1 as we expose the DPR to more passages that are similar to the dialog question. The maximum performance (R@15 = 91.4) is attained by LambdaMART on passages from the top-30 documents.[4]

Error analysis — We noted that the document retriever fails on 42 cases out of 4201 (i.e., R@30 = 99.0). We identified 4 major error types: (i) *topic shift* (22 cases), where grounding information hops from one document to another; (ii) *vague question* formulation (12 cases), where user questions are unclear and require the agent to ask follow-up questions for clarification; (iii) *annotation errors* (4 cases) due to some meaningless utterances; (iv) *hard examples* (4 cases) where our retriever totally failed.

4.3.2 Response Generator

Generation models are trained and evaluated on our LambdaMART retriever's output, ranking passages from the top-30 documents. The number of preceding dialog turns from the history (that are fed as input for the generator, see Fig. 1) is fixed at 5, which is the length leading to the best performance on the dev set in our preliminary experiments. Each turn is prefixed by a role indicator, i.e., ⟨AGENT⟩ and ⟨USER⟩. A separator ⟨CONTEXT⟩ is inserted between the question and passage text. See Appendix C for hyperparameter details. The evaluation metrics are calculated by the official shared task script. Our experiments study the impact of the number of passages in the generator's input and establish upper bounds of its performance. In addition, we introduce "knowledge misrecognition rate" to quantify limitations of our generation model (see further).

Upper-bound Tests — We perform three types of upper-bound tests as shown in Table 3: (i) only the grounding *passage* is provided to the FiD model (for both of train and dev sets) to generate a response (row 3); (ii) only the grounding *span* (phrases or sentences within the grounding passage) is input to the FiD model for generation (row 4); (iii) use the grounding span as the response to be evaluated against the gold one (row 5). Scores in Table 3 demonstrates a notable gap (78.33 w.r.t. total score) between the baseline (row 4) and an upper-bound model (row 1). It is noteworthy that directly using the grounding span as the response yields better performance (224.66) than inputting it to FiD (214.1), implying that a span-extraction model might get higher scores than the current generation one. However, while extracting correct spans provides users the needed information, it cannot satisfy the pragmatic requirements of a conversation (e.g., greetings at the beginning, yes/no prefix before giving the details). Thus, we choose a generation model as it has greater power in generating more coherent phrases at potential cost of losing partial information.

Impact of the Number of Passages $\mathbf{N_p}$ — Intuitively, the more passages are fed as input to FiD, the higher is the chance for FiD to capture the grounding information. Yet, it then also becomes harder to recognize the correct passage. We thus hypothesize that there should be an optimal number of passages N_p for which FiD to attains the best performance, without being distracted by too much information. Figure 2 shows that all performance metrics slightly drop when N_p exceeds 15 (even though they mostly recover once $N_p \geq 30$). The performance of the best model ($N_p = 15$) on the dev set is listed in row 5 of Table 3, with F1_U = 42.98 and SacreBLEU = 27.05.

Knowledge misrecognition — Comparing our system results to the upper bounds (see row 1 in Table 3), we note there is still considerable room for improvement. To identify where our generation model fails, we scrutinize generated responses sampled from inference results on dev set of our best

[4] We select 30 documents, because at the document level, we find R@30 = 99.

	Retriever	Model	F1_U	SacreBLEU	Meteor	ROUGE_L	Total
1	Perfect-	FiD + Grounding passage	51.99	37.97	47.60	50.20	187.76
2	Retriever	FiD + Grounding span	58.03	43.56	55.31	57.20	214.10
3		Grounding span as response	61.00	47.37	60.09	56.18	224.66
4	DPR	Baseline (RAG)	32.62	18.97	27.22	30.61	109.43
5	LambdaMART	FiD + LambdaMART	**42.89**	**27.05**	**42.69**	**40.51**	**153.15**

Table 3: Generation performance of the baseline and our FiD-BART-base model (*seen-domain* task; on dev set). Row 1-3 list the upper-bound performance. A perfect-retriever assumes that the grounding passage is always ranked as the top 1. Row 4-5 use realistic retrievers. The baseline scores are our reproduction results.

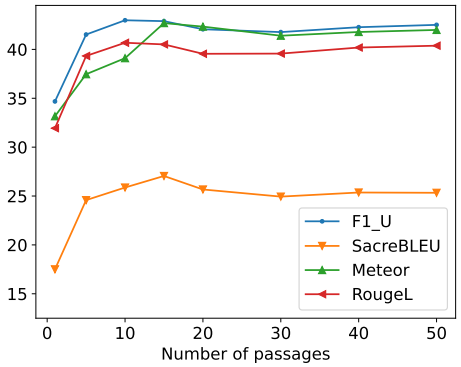

Figure 2: Impact of the number of passages ($N_p \geq 1$) on generation metrics. (*seen-domain* task; FiD-BART-base model; on dev set)

system ($N_p = 15$). An interesting observation is that the FiD model may behave poorly even when the grounding passage is retrieved among the top-15 results presented to the generator: FiD cannot always recognize the grounding passage among its multiple inputs. We propose to quantify this with "knowledge misrecognition rate" μ, calculated as the fraction of low-quality responses among all cases where the correct passage is included in the retrieved ones as fed as input to the generator. For example, using SacreBLEU, a low value thereof (e.g., <10) suggests that the model did not actually use elements from the ground truth passage in generating the response. Thus, using SacreBLEU < 10 as an indication of a "low-quality" response, we find that the misrecognition rate of our best system is $\mu = 50.3\%$ on the dev set. This means that over half of the correct retrieval results are lost in the generation phase. The high rate also implies that the FiD model alone lacks the necessary inductive bias to identify the grounded information among multiple passages. We consider this as a key element in designing future versions of the response generation component.

Leaderboard Submission — Our submission results on the test sets (including test-dev and test-test) are listed in Table 4. For the *unseen-domain* task, inference was performed by the model trained on *seen-domain* data as a test of our system's zero-shot ability. Besides the FiD-BART-base model, we also train a FiD-BART-large model, which achieves our best scores. For the *seen-domain* task, our best model outperforms the baseline by 11.05 and 10.07 for F1_U and SacreBLEU. For the *unseen-domain* task, these two metrics are improved by 14.10 and 14.88. As a result, our UGent-T2K team was ranked second and third for the *seen-domain* and *unseen-domain* tasks respectively.

5 Conclusion

We propose a pipeline system for dialogs grounded in multiple documents. Our system consists of a document retriever, a passage retriever and a multi-passage-fusing generator. The retriever is designed to limit the passage search space by first ranking documents, which proves to enhance the passage retrieval performance considerably for the MultiDoc2Dial shared task. Compared to the baseline RAG model, our multi-passage-fusing generator achieves better knowledge-grounded answer generation. Based on error analysis of our current system, future work will focus on the topic shift issue for conversational retrieval and investigate the knowledge misrecognition problem for dialog generation.

Acknowledgements

This research received funding from the Flemish Government under the *"Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen"* programme. The first author was supported by *China Scholarship Council* (201806020194). We thank the anonymous reviewers whose comments helped to improve our work.

Task	Model	F1_U	SacreBLEU	Meteor	ROUGE_L	Total
seen-domain	Baseline	35.85	22.26	34.28	33.82	126.21
	FiD-BART-base	42.51	28.52	42.8	40.3	153.13
	FiD-BART-large	**46.90**	**32.23**	**47.96**	**44.89**	**171.98**
unseen-domain	Baseline	19.26	6.32	16.77	17.16	59.52
	FiD-BART-base	29.35	19.87	29.57	27.84	106.64
	FiD-BART-large	**33.36**	**21.20**	**33.57**	**31.47**	**119.60**

Table 4: Submission results on the leaderboard (on test-test set).

References

Giambattista Amati. 2006. Frequentist and bayesian approach to information retrieval. In *Proceedings of ECIR*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of ICLR*.

Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Microsoft Research Technical Report*.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. QuAC: Question answering in context. In *Proceedings of EMNLP*.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine Learning*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL*.

Song Feng, Siva Sankalp Patel, Hui Wan, and Sachindra Joshi. 2021. MultiDoc2Dial: Modeling dialogues grounded in multiple documents. In *Proceedings of EMNLP*.

Song Feng, Hui Wan, Chulaka Gunasekara, Siva Patel, Sachindra Joshi, and Luis Lastras. 2020. doc2dial: A goal-oriented document-grounded dialogue dataset. In *Proceedings of EMNLP*.

Xiao Han, Yuqi Liu, and Jimmy Lin. 2021. The simplest thing that can possibly work: (pseudo-)relevance feedback via text classification. In *Proceedings of SIGIR-ICTIR*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of EACL*.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of ACL*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of EMNLP*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020b. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of NeurIPS*.

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

Craig Macdonald, Richard McCreadie, Rodrygo LT Santos, and Iadh Ounis. 2012. From puppy to maturity: Experiences in developing Terrier. In *Proceedings of SIGIR-OSIR*.

Siva Reddy, Danqi Chen, and Christopher D. Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*.

Andrew Trotman, Antti Puurula, and Blake Burgess. 2014. Improvements to bm25 and language models examined. In *Proceedings of ADCS*.

Appendices

A Passage segmentation

The current version of the MultiDoc2Dial dataset provides 488 documents in which we found 56 duplicate documents[5]. The baseline relies on a

[5]The full list of duplicates can be found in https://bit.ly/376TxPX

structure-wise segmentation method. More specifically, in a document html file, a header tagged by <h1> or <h2> and its children nodes are treated as a passage prefixed by its hierarchical titles. We note that some passages produced in this way are too short (424 passages are shorter than 20 tokens, e.g., headers with empty content below) or too long (24 passages longer than 1,000 tokens) as shown in Fig. 3(a), not to mention those repetitive passages due to document duplicates. Given that common transformer-based generation models takes input up to 512 tokens, such length distribution either wastes a generation model's capacity when short passages are padded or loses a significant portion of information when long passages are truncated. To eliminate these extreme cases, three measures are taken based on our cleaned document set: (i) We remove the 56 duplicate documents. (ii) For each of the remaining documents, we first split it using the structure-wise method, calling the results "sections" to differentiate from the baseline's "passages". If a section has fewer than 150 tokens, it is directly added to the final passage list. If not, it will be further split into passages using a flexible sliding window which allows for a passage with tokens fewer than the window size in order to not break sentences.[6] (iii) Next, a passage with fewer than 60 tokens is merged with its following passage — except if it appears at the end of a section, in which case it will be appended to its preceding one. Figure 3(b) depicts the passage length distribution using our segmentation method. The long tail problem of the baseline is largely resolved. As Table 5 shows, our new segmentation method reduces the total number of passages from 4,110 to 3,734 while it increases the average passage length from 130.4 to 154.1.

Passage-Segmentation	#passages	avg length (tokenizer)	avg length (white space)
Baseline	4,110	130.4	105.4
Ours	3,734	154.1	132.5

Table 5: Total number of passages and average passage length produced by the baseline method and ours. "tokenizer" and "white space" denote using the BART tokenizer and splitting words by white space respectively.

[6]Window size $\leq$ 150, stride = 50. Since we rely on Spacy to extract sentences, some of them may be broken depending on Spacy model's decision.

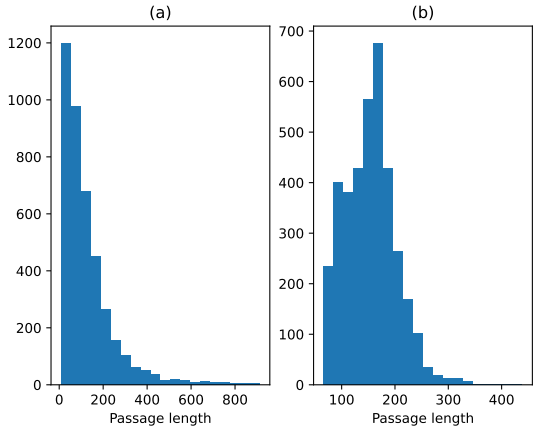

Figure 3: Passage length histograms of baseline and our passage segmentation. The length is the number of tokens processed by the BART tokenizer. (a) Baseline passages. The x-axis is truncated to 1,000 to make smaller value bins more clear. (b) Our passages after removing duplicate documents and merging short passages. No passage is omitted.

B Experiments

This section reports (i) the ablation study of BM25 for document retrieval revealing how different features affect the retrieval performance; (ii) domain classification that enhances document retrieval; (iii) passage retrieval experiments based on our new segmentation method.

B.1 Ablation study of BM25 for document retrieval

Table 6 presents our results for $\text{BM25}_{\text{tuned}}$ on document retrieval. The first row shows the simple BM25 model without any preprocessing on inputs (question and documents). The next four rows respectively represent: lower casing inputs, removing stop-words, removing punctuation, and stemming, which greatly improve the performance (over +10 points for R@25). We obtained slight improvement with a domain classifier that predicts the conversation domain (see Appendix B.2). We also observed that using n-grams ($n = 1,2,3$) features instead of unigrams brings a further improvement with additional 3.2 points of R@25.

B.2 Domain Classifier

In the training data of MultiDoc2Dial , the grounding documents were crawled from 4 U.S. government websites,[7] covering 4 domains: Social Security Administration, U.S. Department of Veterans

[7]ssa.gov, va.gov, dmv.ny.gov, studentaid.gov

Model	R@1	R@5	R@10	R@25
BM25	45.6	66.3	73.3	81.4
+ lowering	46.6	67.7	74.3	82.3
+ stop-removal	50.2	74.3	82.2	90.2
+ punk-removal	52.4	77.5	84.4	92.5
+ stemming	50.3	75.9	83.7	91.6
+ domain-scores	50.7	76.6	84.5	92.6
+ n-grams	57.8	84.2	89.6	95.8

Table 6: $\text{BM25}_{\text{tuned}}$ recall scores for document retrieval on dev set.

Affairs, Department of Motor Vehicles (New York State) and Federal Student Aid, which are respectively noted as `ssa`, `va`, `dmv` and `student`. We applied the idea proposed by Han et al. (2021) to further improve BM25 performance by training a domain classifier, i.e., finetuning the RoBERTa-large model (Liu et al., 2019) to predict a domain label for a given dialog. The domain scores are multiplied to BM25 after which a weighted combination between the initial BM25 and the new scores is used to create the final ranked list. In our experiments, we simply assume equal weights (0.5) for the two scores. Table 7 presents different classifiers' accuracy for *seen-domain* prediction.

Model	Accuracy
SVM (Cortes and Vapnik, 1995)	96.7
Bert-large (Devlin et al., 2019)	97.0
Roberta-large (Liu et al., 2019)	98.2

Table 7: Domain classifier accuracy on dev set.

B.3 Retrieval based on new segmentation

Table 8 presents the passage retrieval results based on our passage segmentation. We experiment with three models: DPR ranking all the passages, DPR ranking only the passages within top-m documents and the LambdaMART model based on top-30 documents. Restricting DPR's search space within the top-5 documents increases R@15 from 80.1 to 87.1, which further grows to 90.4 with the LambdaMART model.

Model	m	R@1	R@5	R@10	R@15
DPR	3,734	46.3	68.2	76.0	80.1
$\text{DPR}_{\text{top1 doc}}$	1	52.9	70.8	73.2	73.6
$\text{DPR}_{\text{top5 docs}}$	5	47.2	74.0	83.0	86.9
$\text{DPR}_{\text{top10 docs}}$	10	45.9	71.8	81.0	85.3
$\text{DPR}_{\text{top30 docs}}$	30	45.2	70.1	78.8	83.1
LambdaMART	30	48.0	80.0	87.4	90.4

Table 8: Recall scores for passage retrieval on dev set. The passage set is produced by the method described in Appendix A.

C Hyperparameters

FiD was finetuned from pretrained BART weights with the following hyperparameter settings:

```
batch_size=4
total_epochs=15
max_source_length=400
max_target_length=64
label_smoothing=0.1
optimizer=AdamW
weight_decay=0.1
adam_epsilon=1e-08
max_grad_norm=1.0
lr_scheduler=linear
learning_rate=5e-05
warmup_steps=500
gradient_accumulation_steps=2
```