NORMY: Non-Uniform History Modeling for Open Retrieval Conversational Question Answering

Muhammad Shihab Rashid
Computer Science and Engineering
University of California Riverside
Riverside, CA, USA
mrash013@ucr.edu

Jannat Ara Meem
Computer Science and Engineering
University of California Riverside
Riverside, CA, USA
jmeem001@ucr.edu

Vagelis Hristidis
Computer Science and Engineering
University of California Riverside
Riverside, CA, USA
vagelis@cs.ucr.edu

***Abstract*— Open Retrieval Conversational Question Answering (OrConvQA) answers a question given a conversation as context and a document collection. A typical OrConvQA pipeline consists of three modules: a *Retriever* to retrieve relevant documents from the collection, a *Reranker* to rerank them given the question and the context, and a *Reader* to extract an answer span. The conversational turns can provide valuable context to answer the final query. State-of-the-art OrConvQA systems use the same history modeling for all three modules of the pipeline. We hypothesize this as suboptimal. Specifically, we argue that a broader context is needed in the first modules of the pipeline to not miss relevant documents, while a narrower context is needed in the last modules to identify the exact answer span. We propose NORMY, the first unsupervised non-uniform history modeling pipeline which generates the best conversational history for each module. We further propose a novel Retriever for NORMY, which employs keyphrase extraction on the conversation history, and leverages passages retrieved in previous turns as additional context. We also created a new dataset for OrConvQA, by expanding the doc2dial dataset. We implemented various state-of-the-art history modeling techniques and comprehensively evaluated them separately for each module of the pipeline on three datasets: OR-QUAC, our doc2dial extension, and ConvMix. Our extensive experiments show that NORMY outperforms the state-of-the-art in the individual modules and in the end-to-end system.**

***Index Terms*—question answering, history modeling, conversational, retriever, reranker, reader**

I. Introduction

Conversational Question Answering (CoQA) has recently attracted a lot of attention due to the widespread adoption of voice assistant platforms such as Siri, Alexa, and Google Assistant, and the advances in deep learning [1]–[3]. Given a text passage and a conversation, the goal of CoQA is to extract the answer to the last question of the conversation from the passage. CoQA is an extension to Question Answering (QA) where the input is just one question instead of a conversation [4]–[6]. However, in practice users do not provide an input passage when performing QA or CoQA. This led to the newer problems of Open Retrieval QA (ORQA) [7]–[9] and Open Retrieval Conversational Question Answering (OrConvQA) [10], where the input is a whole document collection.

State-of-the-art works on OrConvQA (also for ORQA) employs a pipeline of three modules [10]–[12]. The first one is a *Retriever*, which retrieves a set of relevant passages from the collection. Both term-based (TFIDF/BM25) and embedding-based approaches may be used by the Retriever. A *Reranker* module then re-ranks the already retrieved documents to better match the question, and finally, a *Reader* module extracts an answer span from the re-ranked documents. Recent advances in transformers have produced pre-trained models like BERT which are highly effective in reader tasks [13].

A key challenge in CoQA and OrConvQA is that the final user question may have co-references or ellipses, that is, some terms may refer to terms in the past conversation, while other useful contexts may be missing from the question. Further, previous (historic) turns of the conversation may add valuable context to the question being asked. Clearly, some of the past turns may be more useful than others as context for the last question. Blindly adding all turns may lead to a noisy history model. Including all turns may also be infeasible for some models like BERT, which can only support 512 tokens as the query and passage.

Previous CoQA works propose different approaches to model the conversational history: some append all history turns to the final query making it a one big query and use it to retrieve the answer [1], [2], or use a backtracking algorithm which selects/disregards a particular history turn using deep reinforcement learning [14], or rewrite the final query using the context of the whole conversation [15]–[17]. Previous OrConvQA works either use the previous 6 turns [10], [11] or all turns with predicted answers [12] as context.

Despite the different history modeling approaches of these previous works, they all use the same history model for all three modules of the pipeline. We hypothesize that this is suboptimal. Specifically, our hypothesis is that as we move towards the right of the *Retriever*→*Reranker*→*Reader* pipeline and the number of the input passages (or documents) decreases, the history context should become shorter and more focused. That is, the Retriever should have access to broader context to not miss any relevant documents, whereas the Reader should have little context to help it identify the exact text span that answers the user question.

For example, in Figure 1 we see that modeling the history with Full Conversational Context (FC) returns the most relevant passage (top one) which has all the necessary information needed to answer the query. In contrast, narrower context –

 DOI: 10.1109/ICSC59802.2024.00022

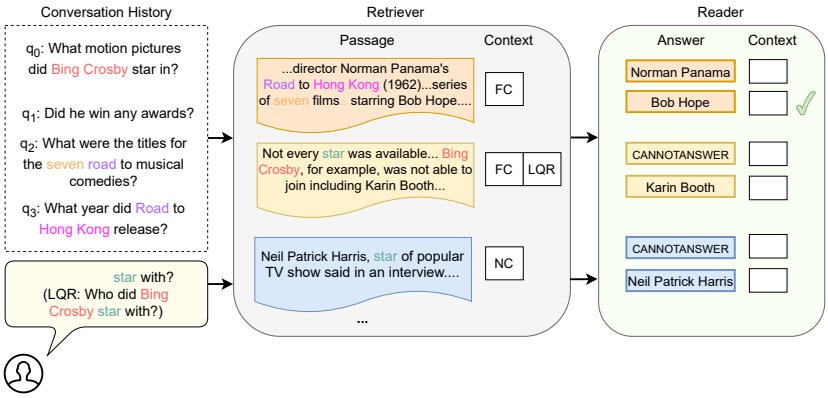


Fig. 1: Example of the impact of non-uniform conversational history modeling. Full Conversational Context (FC) retrieves the most relevant passages in the *Retriever* module, while a narrower context, Last Question Rewrite (LQR) predicts the correct answer span in the *Reader* module.

Last Question Rewrite (LQR) or No Context (NC) – returns suboptimal passages that do not contain the answer. Once documents are retrieved, additional context (FC) may act as noise for the *Reader* module, whereas more focused context (LQR) is able to extract the right span.

In addition to using the same context, the state-of-the-art pipelines [10]–[12] and history modeling approaches [14]–[16] depend fully on training data to fine-tune the *Retriever*, *Reranker*, and *Reader* modules. Finding quality training data for various domains of OrConvQA datasets is challenging. Ideally, a pipeline should be domain-agnostic and use appropriate history modeling to capture the context. In this paper, we contribute in three ways towards solving the OrConvQA problems: (a) we propose NORMY[1], the first unsupervised solution pipeline, (b) we build and publish a new dataset, and (c) we implement and experimentally compare various state-of-the-art history modeling algorithms for each of the three modules of the *Retriever*→*Reranker*→*Reader* pipeline.

Our proposed system, NORMY, uses a *non-uniform* history context for the three pipeline modules. We also propose a novel history modeling algorithm for the *Retriever* module that produces improved results over state-of-the-art baselines. Unlike previous approaches where the passages retrieved in previous turns are discarded, our Retriever algorithm considers past passages as candidates and proposes a ranking function that combines turn-based decay with context-based reranking of each passage. This ensures we do not miss an important passage due to noise being added in later turns. Table I summarizes the related work landscape, where we see that none of the previous work addresses all aspects of the problem. Only NORMY is question answering, open retrieval, conversational, performs history modeling, and it is non-uniform.

We evaluate our individual modules and the overall pipeline using three varied datasets. First, we use the *ORQUAC* dataset [10], which is an extension of the CoQA dataset [1]. A drawback of this dataset is that it does not portray natural dialogue conversation, as the chat is limited to asking questions and getting answers. Thus, we selected the *doc2dial* [22] dataset for additional evaluation. However, this dataset is not created for the open retrieval conversational QA task as there are only a small number of documents as a corpus and the focus was to generate natural language answers and not text spans. For that, we created an updated *doc2dial* dataset, which we call *doc2dial-Or*. Third, we conduct experiments on ConvMix [23], where the corpus includes single-sentence passages and the history turns contain fewer co-references than previous datasets mentioned. NORMY outperforms the state-of-the-art in all three datasets. In summary, we make the following contributions in this paper:

- We identify the problem of uniform history modeling in conversational QA and propose the first end-to-end pipeline for OrConvQA that uses non-uniform history modeling.
- We propose NORMY, a new unsupervised non-uniform universal history modeling pipeline. NORMY employs a novel history modeling approach for the Retriever module, which builds on keyphrase extraction principles, and leverages returned passages from previous history turns.
- We perform an extensive comparison and analysis of

TABLE I: Comparison of selected tasks and datasets on the dimensions of Question Answering(QA), Open Retrieval(OR), Conversational (Conv), History Modeling (HM) and Non-uniform History Modeling (NHM)

Task/Dataset	QA	OR	Conv	HM	NHM
NQ [4], SQuAD [6] TriviaQA [18],	✓	✗	✗	✗	✗
MSMarco [7],DrQA [19] CoQA [1],	✓	✓	✗	✗	✗
QuAC [2],ShARC [20]	✓	✗	✓	✗	✗
HAE [21], RL [14], RW [15]	✓	✗	✓	✓	✗
OrConvQA [10], d2d [22]	✓	✓	✓	✗	✗
NORMY**[ours]**	✓	✓	✓	✓	✓

[1]**N**on-Unif**ORM** Histor**Y** Modeling

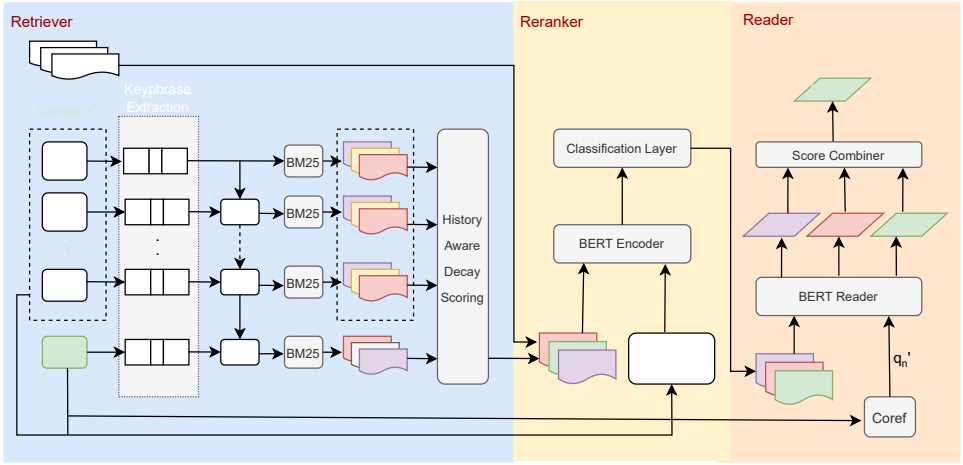


Fig. 2: The architecture of NORMY. The input is the current question q_n, all history questions q_i^{n-1}, and the document collection D. The *Retriever* module models the history using keyphrase extraction per history turn and retrieves passages $P_0 \cdots P_k$ using BM25. Our novel History Aware Decay Scoring module refines all returned passages and outputs top-k. The *Reranker* reranks the passages using most recent w turns and *Reader* uses coreference resolution to rewrite the last query q_n and outputs the best answer span combining all three modules' scores.

various history modeling techniques for each module of the pipeline, on three diverse and structurally different datasets, and show that using the same modeling is suboptimal.
- We expand the *doc2dial* dataset for the OrConvQA task and make our full source code and dataset available to the community [2].

The rest of the paper is organized as follows. An overview of the problem and solution are presented in Section II. We present the details of NORMY including our novel Retriever algorithm in Section III. We present the datasets and results of our experimental evaluation in Section IV. We discuss the related work in Section V and finally conclude in Section VI.

II. Problem Definition and Overview of NORMY

Problem Definition. The input to the OrConvQA problem is a question q_n, the conversational history $C = q_0, \cdots, q_{n-1}$, and a document collection D. As in previous work, the history does not contain the answers to the questions, we also assume no access to the ground truth answers [10]. The output is an answer span a_n, extracted from one of the documents in D, which best answers q_n. The solution pipeline is shown in Figure 2. There are two key decisions we have to make for each module. First, pick what algorithm to employ (e.g. BM25 [24] or BERT [13] and so on) and with what parameters. Second, define what conversational context C to input to the algorithm. In this work, we employ the state-of-the-art algorithm for each module and focus on the choice of conversational context for each module.

Overview of NORMY. NORMY, as shown in Figure 2, generates a different model of the conversational history for each module of the pipeline. Given the Retriever's history model discussed below and the collection, the *Retriever* selects the top k passages using BM25. Then, using a history of the last w turns, the *Reranker* reranks the k passages using transformer-based similarity measures. Finally, the transformer-based *Reader* module models the history by rewriting the final query into a self-contained query, using coreference resolution, to find the best answer span. The answer span with the highest combined score from all three modules is the final answer. Note that our whole system is designed in an unsupervised fashion. There is no training data needed.

III. Modules of NORMY

A. Retriever

The *Retriever* module retrieves the k most relevant passages from a document collection D, given a query q_n and context C. We considered two types of search algorithms: classic Information Retrieval BM25-style ranking methods, and dense retriever methods. Although dense retriever approaches like ORQA [25] and DPR [26] which use encodings of documents using ALBERT [27], have shown to provide better results, they require training data for fine-tuning. Their vanilla pre-trained models without training do not perform as well as BM25 (shown in Section IV). Further, BM25 is more scalable for large collections. Hence, given that the main focus of this paper is history modeling, we picked BM25 for our Retriever. Specifically, we index the documents using Lucene[3]. Then we retrieve top k documents using BM25, which is a term frequency based document ranking algorithm.

History Modeling. NORMY's Retriever has two key novelties. First, we use a *keyphrase extraction*-based candidate selection algorithm to identify the key context from the whole

[2]https://github.com/shihabrashid-ucr/normy

[3]https://lucene.apache.org/pylucene/

Algorithm 1 NORMYRetriever

Input: Context C, q_n
Output: SEL: Top k returned passages
1: $SEL \leftarrow []$, $P \leftarrow []$
2: **for** *each turn* $i \in 1 \cdots n$ **do**
3: $R(q_i) \leftarrow YAKE(q_i)$
4: $R(C) \leftarrow R(q_0) \cup \cdots \cup R(q_i)$
5: $P_i \leftarrow Retrieve_k(R(C))$ //top-k by BM25
6: $P \leftarrow P \cup P_i$
7: **for** *each passage* $p \in P_i$ **do**
8: $Compute\ score\ S_{rt}(p)\ using\ Eq.\ 2$
9: $SEL \leftarrow$ SelectTopk(P) //based on $S_{rt}(p)$
10: **return** SEL

conversational history. Second, we consider all passages returned by previous turns alongside passages returned by final turn as candidate passages, and rank them using *history aware decay scoring* method to return top k.

Our retrieval algorithm is shown in Algorithm 1. We extend the keyphrase extraction algorithm YAKE [28] to select y best keywords per conversation turn using the YAKE formula shown in Equation (1). YAKE considers features like the casing of the word, word positions, word frequencies, word relatedness to context, etc. to assign a score $S(b)$ to each word b.

$$S(b) = \frac{W_{Rel} \cdot W_{Pos}}{W_{Case} + (W_{freq}/W_{Rel}) + (W_{DifS}/W_{Rel})} \quad (1)$$

where W_{Rel} is the relatedness to context score, W_{Pos} is the word position score, W_{Case} is the word casing score, W_{freq} is the word frequency divided by the sum of mean term frequency and standard deviation σ, and W_{DifS} is calculated based on how many times a particular word appears in other sentences. The detailed equations of all the terms can be found in [28]. We compute the union $R(C)$ of the reformulated questions $R(q_0) \cdots R(q_n)$ to retrieve the top k passages (line 4-5). Each passage returned has a BM25 score assigned to it.

History-Aware Decay Scoring. After retrieving k passages for the current turn n, we refine their scores by considering their similarity to the retrieved passages. Further, we assign less weight to passages returned from previous turns. Specifically, we use a decay weight λ to update the scores of older passages. The passages returned from previous turns may be relevant for subsequent modules but they do not share equal weight to passages returned from current turn n. Next, to assess the relevance of each passage P_{nj}, $\{j = 1 \cdots k\}$ from turn n, we compute the average pairwise similarity with passages returned in the previous turn $P_{(n-1)i}$, $\{i = 1 \cdots k\}$. This ensures that the passages returned has relevance to the whole conversation. Passages returned from irrelevant conversation turns will be scored less. To compute the similarity, we use SBERT [29] to produce embeddings of passages and perform cosine similarity. We update the score of P_{nj} using this similarity. Finally, we rank all the passages using updated scores S_{rt} and select top k. The retriever score of a passage p is shown in Equation (2).

$$S_{rt}(q_n, C, p) = max(BM(R(C \cup q_n), p) - \lambda, 0) \cdot \sum_{i=1}^{k} sim(p, P_{(n-1)i})/k \quad (2)$$

where ***BM(.)*** is the BM25 score of a passage and ***sim(.)*** returns semantic similarity between two passages.

B. Reranker

The *Reranker* module reranks the retrieved top k passages using transformer-based encoders and a neural network to compute passage relevance score. The transformer based Reranker augments BM25 ensuring an extra layer of passage relevance. As $k <<$*total size of collection*, using a transformer encoder is inexpensive. A Reranker has been shown to improve the overall performance of the end-to-end system with little additional cost [10], [30]. However, we show that using the same history modeling as the previous module or using the context from all history turns do not give the best results as now we have grounded documents as evidence. Our experimental results show that using a context with a history window size w works best. The input to the module is the final query q_n, the context C, and k passages retrieved by the Retriever. A reranking score S_{rr} is assigned to every passage.

Encoder. Our Reranker module uses BERT to encode the input representation. We use the last w history turns before q_n and concatenate them together to model the history. We then concatenate the retrieved passage p_j, $j = \{1 \cdots k\}$ to the appended history turns to create the final input sequence (q_n, C, p_j) = [CLS] q_{n-w} [SEP] $\cdots$ [SEP]q_{n-1} [SEP]q_n [SEP]p_j. We use the contextualized vector representation of the input sequence $\nu_{[CLS]}$, and use it as input to a fully connected feed-forward layer that classifies the given passage as either *relevant* or *non-relevant* and outputs a classification score S_{rr}:

$$\nu_{[CLS]} = W_{[CLS]} BERT(q_n, C, p_j)_{[CLS]} \quad (3)$$

$$S_{rr} = P(Rel = 1 | q_n, C, p_j) \stackrel{\triangle}{=} softmax(\nu_{[CLS]}) \quad (4)$$

where $\nu_{[CLS]} \in \mathbb{R}^T$, T is the model embedding dimension, which is 768, and $W_{[CLS]}$ is a projection of the $[CLS]$ representation to obtain the sequence representation $\nu_{[CLS]}$. We compute the score for each passage in top k independently and rerank them based on S_{rr}.

C. Reader

The *Reader* module inputs the final query q_n, the context C and the reranked passages $\{p_1, p_2 .... p_k\}$ and outputs a span from one of the passages as the answer.

History Modeling. As the documents have already been narrowed down using the conversational context in previous modules, we show that using a history modeling with full contextual information produces worse results than a history model that uses less context. This happens due to: 1) The passages already hold the necessary contextual information

TABLE II: Dataset Statistics

	ORQUAC	doc2dial-OR	ConvMix
# Dialogues	771	661	1679
# Questions	5571	4253	2284
# Avg tokens/qstn	6.7	10	6.39
# Avg tokens/ans	12.2	21.6	2.17
# Avg questions/conv	7.2	6.4	5.00
# Passages	11M	11.6M	5.94M

from the history, 2) Previous history questions in the context misdirects the BERT Reader model into predicting incorrect answer spans. The naive idea would be to use just the final query as input. However, the final query is prone to co-references and ellipses as users will not use self-contained utterances in a natural conversation. Thus, we use a co-reference resolution model to generate a resolved final query $q_n{}'$ using the previous context. We adapt the huggingface neural co-reference model [4] which uses two neural networks to assign a score to each pair of mentions (or co-references) in the input and their antecedents [31]. The history turns q_0 to q_{n-1} are concatenated and used to rewrite q_n into $q_n{}'$.

Encoder. Our Reader module uses similar BERT architecture as the previous module to encode the input. The input sequence "$[CLS]q_n{}'[SEP]p_j$" is used to generate a representation of all tokens in the input. Two sets of parameters, a start vector W_s and an end vector W_e are used to compute the score for the m-th token.

$$\nu_{[m]} = BERT((q_n{}', p_j))_{[m]} \tag{5}$$

$$S_s(q_n{}', p_j, [m]) = W_s\nu_{[m]} \quad S_e(q_n{}', p_j, [m]) = W_e\nu_{[m]} \tag{6}$$

where S_s is the start score of a token and S_e is the end score. The span Reader score S_{rd} is computed as the maximum score of each token being either the start or end token. The start token must appear before the end token in the input.

$$S_{rd}(q_n{}', p_j, s) = \max_{[m_s],[m_e]\in(q_n{}', p_j)} S_s(q_n{}', p_j, [m_s]) + S_e(q_n{}', p_j, [m_e]) \tag{7}$$

where s is the answer span with the start token $[m_s]$ and end token $[m_e]$. The answer spans are re-ranked using the combined score of all three modules and the top answer is given as a prediction.

$$S(q_n, C, p_j, s) = S_{rt}(q_n, C, p_j) + S_{rr}(q_n, C, p_j) + S_{rd}(q_n{}', p_j, s) \tag{8}$$

IV. Experimental Evaluation

A. Datasets

We use three datasets with different conversation structures. The first dataset: ORQUAC [10] is an aggregation of three existing datasets: QuAC, CANARD [32], and Wikipedia corpus that serves as a knowledge source for open retrieval. The Wikipedia corpus is a collection of 11 million passages that are created from splitting Wikipedia articles into 384 tokens.

The second dataset is *doc2dial-OR*, which was created by us, as an extension of *doc2dial* [22] dataset, which consists of natural information-seeking goal-oriented dialogues that are grounded in documents. doc2dial has more complex questions than ORQUAC, associated with multiple sections of a document. However, this dataset is only grounded to 480 long documents collected from different government websites, which is not ideal for an open retrieval task. *doc2dial-OR* extends doc2dial by having a much larger set of passages consisting of (a) 11 million Wikipedia passages[5], and (b) the 480 documents of doc2dial split into 384-token chunks. The second difference between doc2dial-OR from doc2dial is that we convert free-text ground truth answers to exact text spans from the gold passage in the dataset, to make it suitable for a span prediction task, as is the case for ORQUAC.

The third dataset is ConvMix [23], which contains documents from heterogeneous sources: Wikipedia info boxes, tables, and text snippets (passages). They use the Wikipedia dump from 2022-01-31. To adapt this dataset for our task, we selected the 5.94 million textual snippets as our collection and the question turns in a conversation where the answer can be extracted from these text snippets. The dataset statistics are shown in Table II.

B. Experimental Setup

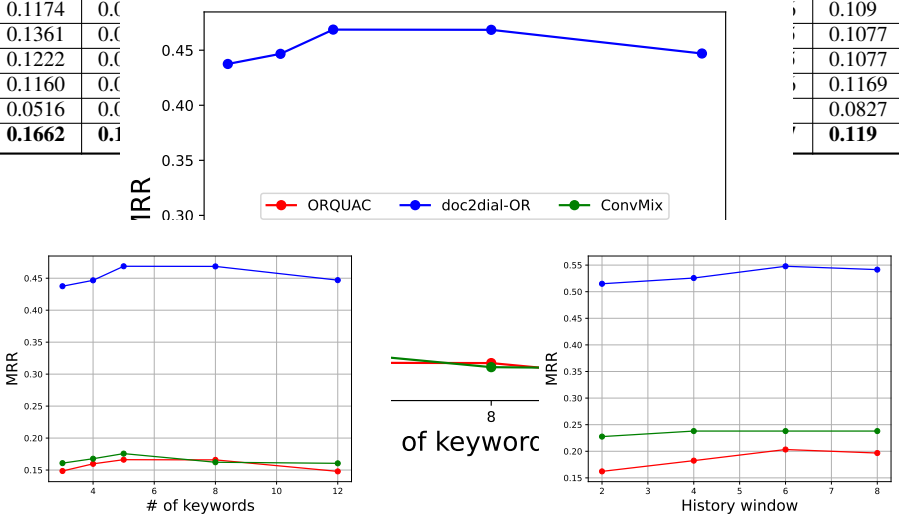
Competing History Models. To the best of our knowledge, there are no fully unsupervised non-uniform history modeling approaches. There are supervised systems (OrConvQA [10], WS-OrConvQA [11]) that uses history window of 6 for all the modules. ConvADR-QA [12] uses all history turns along with their predicted answers as context. However, they require annotated data to train the modules. We adapt their approaches to an unsupervised setting. There are also conversational closed retrieval systems (RL [14], RW [15]). We adapt such history modeling methods to an open-retrieval setting. Further, we also propose some standard history modeling techniques. Thus, we have identified the following baselines: **1) No History [2]:** Where we do not perform any history modeling. The last question turn is used as input to each individual module. **2) First-Last:** We propose an intuitive history modeling baseline, where we define the history as the combination of the immediately previous user utterance and the first utterance of the conversation. **3) Full History:** With all previous turns concatenated. For the modules where we use transformer models with token limitation, we prune the earlier tokens if the total token size exceeds 384. The input sequence is $C = [CLS]q_0[SEP]q_1\ [SEP]\cdots[SEP]q_n$ **4) YAKE [28]:** Keyphrase extraction-based history modeling has not been used previously in any research work. We extract y keyphrases per history turn and concatenate them with the last turn to create the input sequence. **5) Backtracking [RL]:** We adapt the *immediate reward* based history selection proposed by Qiu et al. [14] for closed retrieval systems. We select a history turn if the similarity with previously selected history

[4]https://huggingface.co/coref/

[5]https://dumps.wikimedia.org/enwiki/20191020

TABLE III: Retriever Results

Setting	ORQUAC		doc2dial-OR			ConvMix		
	MRR	R@	MRR	R@1	R@5	R@1	R@5	R@10
No History	0.0312	0.0	0.0	0.0	0.0	0.0696	0.1449	0.186
First Last	0.1174	0.0	0.0	0.0	0.0	0.109	0.2263	0.281
Full History	0.1361	0.0	0.0	0.0	0.0	0.1077	0.2364	0.2911
Fixed window	0.1222	0.0	0.0	0.0	0.0	0.1077	0.2364	0.2911
Backtracking	0.1160	0.0	0.0	0.0	0.0	0.1169	0.2429	0.3104
Rewriting	0.0516	0.0	0.0	0.0	0.0	0.0827	0.1654	0.2123
NORMYRetr[ours]	**0.1662**	**0.1**	**0.0**	**0.0**	**0.0**	**0.119**	**0.2513**	**0.3139**



(a) Retriever performance with y

(b) Reranker performance with w

Fig. 3: (a) and (b) subgraphs show the impact of number of keywords **y** and history window size **w** for Retriever and Reranker.

turns is greater than 0.5. For each history turn($i = 1...n$) we calculate the immediate reward with SBERT sentence encoder. **6) Question Rewriting [RW]:** We adapt the algorithm of Vakulenko et al. [15] for closed retrieval systems, which uses a question rewriting model to resolve ambiguous questions (co-references) into self-contained questions. Their model requires training data thus we use *neuralcoref* to resolve the co-references of the final query using previous history turns. There are other query rewriting models like QReCC [33] which also require annotated data. **7) Fixed Window [OrConvQA, WS-OrConvQA] [10], [11]:** WS-OrConvQA is an improvement over OrConvQA but uses training data to learn weak supervision signals. Both models use a history window size w. Window size 6 is shown to produce the best results for both. To select the baseline, we also compared different window sizes (2,4,6,8) in a small validation set and $w = 6$ has given the best results. **8) Fixed Window with Ans. [ConvADR-QA] [12]:** This model predicts the answers for each historical question with a teacher model using annotated query rewrites and appends the predicted answer to the context along with historical questions. For a fully unsupervised pipeline, we adapt this approach to predict an answer for each question using OrConvQA and append the answer to the context. The input sequence is $C = [CLS]q_0a_0[SEP]q_1a_1\ [SEP]\cdots[SEP]q_na_n$, where a_n is the predicted answer for turn n. **Implementation Details.** NORMY is fully unsupervised without requiring any training data. The pre-trained models are implemented with the open-source library Huggingface. We index our document collection using PyLucene with *StandardAnalyzer* as tokenizer Indexing is done with term frequencies, document frequencies, and positions. Keyphrase extraction is done with open source library *pke* [34]. For computing similarity, we use SBERT's pre-trained *roberta-large* model. For *Reranker* module, we use pre-trained BERT model finetuned on MSMARCO dataset [35]. For our *Reader* module we use a pre-trained *bert-large* model finetuned on SQUAD dataset. We make our full code available to the research community.

C. Retriever Results

Evaluation Methodology. We use the commonly used *Mean Reciprocal Rank (MRR)* and *Recall (R@k)* methods to measure our retrieval performance. *MRR* calculates how far down the ranking the first relevant document is on average, where higher is better. *R@k* measures the fraction of times the correct document is found in the top k predictions, where higher is better.

Results. We first experiment with different values of the number of keywords y in the keyphrase extraction shown in Figure 3a and find that $y = 5$ performs best. We use $y = 5$, $k = 10$ and $\lambda = 0.1$ in Table III. We found that using a larger k increases the execution time of the system with a very small accuracy benefit. Figure 3 shows our experimental results for different parameters used in NORMY.

We compare different history modeling techniques using BM25 in Table III. For all three datasets, we see that our NORMY Retriever performs significantly better than other baselines. This is due to a couple of factors: 1) We remove irrelevant information from each history turn rather than eliminating the history turn altogether, 2) We consider previously retrieved passages from previous history turns as candidate passages. We can also see that history models that use fewer contexts like Question Rewriting, First-Last, and No History

TABLE IV: Reranker Results

Setting	ORQUAC				doc2dial-OR				ConvMix			
	MRR	R@1	R@5	R@10	MRR	R@1	R@5	R@10	MRR	R@1	R@5	R@10
No History	0.1408	0.1033	0.1927	0.2891	0.4572	0.3592	0.5901	0.6780	0.2264	0.1834	0.285	0.3139
First Last	0.1779	0.1491	0.2179	0.2891	0.5247	0.4514	0.6301	0.6780	0.2396	0.2052	0.2955	0.3139
Full History	0.1996	0.1702	0.2402	0.2891	0.5401	0.4644	0.6415	0.6780	0.2405	0.2020	0.2944	0.3139
NORMY(Fixed w)	**0.2033**	**0.1767**	**0.2411**	**0.2891**	**0.5478**	**0.4747**	**0.6515**	**0.6780**	0.2405	0.2020	0.2944	0.3139
Backtracking	0.1954	0.1661	0.2361	0.2891	0.5325	0.4635	0.6368	0.6780	0.2403	0.2017	0.2940	0.3139
Rewriting	0.1696	0.1344	0.2181	0.2891	0.4583	0.3604	0.5911	0.6780	0.2295	0.1873	0.2867	0.3139
YAKE	0.2008	0.1731	0.2387	0.2891	0.5396	0.4624	0.6377	0.6780	**0.2418**	**0.2031**	**0.2946**	**0.3139**

TABLE V: Reader F1

Setting	ORQUAC	doc2dial-OR	ConvMix
No History	0.1557	0.1898	0.6785
First Last	0.0996	0.1291	0.4842
Full History	0.0845	0.1196	0.3711
Fixed Window	0.0848	0.1200	0.4859
Backtracking	0.1118	0.1541	0.5591
NORMY(Rewriting)	**0.1774**	**0.2220**	**0.7393**
YAKE	0.1277	0.1551	0.67

TABLE VI: Entire Pipeline F1. ‡ means statistically significant improvement over baseline with $p < 0.5$.

Setting	ORQUAC	doc2dial-OR	ConvMix
NORMY[ours]	$\mathbf{0.0782}^{\ddagger}$	$\mathbf{0.1625}^{\ddagger}$	$\mathbf{0.1723}^{\ddagger}$
NORMY w/o decay	$0.0668^{\ddagger}$	$0.1323^{\ddagger}$	0.1490
NORMY w/o sim	$0.0695^{\ddagger}$	$0.1431^{\ddagger}$	$0.1562^{\ddagger}$
OrConvQA [10], WS-OrConvQA [11](BM25)	0.0478	0.0955	0.1314
OrConvQA, WS-OrConvQA (DPR)	0.0466	0.0948	0.1298
ConvADR-QA [12]	0.0454	0.0897	0.1244

perform significantly worse indicating we need more context while retrieving from millions of passages.

D. Reranker Results

Evaluation Methodology. We use the same metrics as the Retriever as both of these modules produce top k passages.
Results. From Table IV we see that the Reranker module significantly improves the ranks of relevant passages. Using a fixed window, which sits between a broader context like Full History and a narrower context like Rewriting produces the overall best results. We conduct experiments with different history window sizes w and show the results in Figure 3b. Fixed window of 6 works best for two of our datasets, which is supported by the literature [10]. For ConvMix, we see that YAKE performs slightly better than all other methods and the results vary very little. This is because the passages in the collection are single sentences only, leading to multiple passages being relevant to the question. The *Reranker* ranks such passages similarly whereas only one contains the gold answer. In the real world, it is unusual for the documents to be single sentences. Note that our hypothesis that the *Reranker* needs less context than the *Retriever* still holds, as YAKE has less context than Full History.

E. Reader Results

Evaluation Methodology. We treat the evaluation of Reader module as a span selection task and adopt token level F1 as the evaluation metric. F1 calculates the similarity between the ground answer and the predicted span, where higher means better.
Results. From Table V we can see significant performance drops when more contexts are added for all three datasets. As the candidate passages have been reduced to 1, transformer-based reader model performs significantly better when only one query is used. Narrower contexts like adding no history and question rewriting perform much better than broader context models further proving our hypothesis. Among them, question rewriting produces a better result. For ConvMix we can see very high F1 scores for appropriate history models as the answers are on average two tokens only, which makes the *Reader* model easily predict answer spans. However, history models with broader context have poor F1 scores for this same dataset indicating the model needs proper history models even in simpler scenarios.

F. End-to-end Evaluation

In this section, we compare our pipeline with the state-of-the-art models [10]–[12], which either use a fixed window of 6 or full history with predicted answers uniformly in all modules. We see in Table VI that SOTA models perform poorly in a fully unsupervised setting. We also see that, ConvADR-QA performs worse than OrConvQA, as in an unsupervised setting, a wrongly predicted answer could misdirect the context to retrieve irrelevant passages. Our pipeline with non-uniform history modeling performs significantly better. SOTA models use fine-tuned dense retriever model (DPR) for their retriever module instead of BM25 which is used by NORMY. We use the vanilla pre-trained version of DPR here as we don't have access to training data. We also compare to a variant of OrConvQA that uses BM25 instead of DPR for completeness. We see that BM25 performs better than dense retriever models. Note that, uniformly using other baseline history modeling techniques evaluated in previous subsections does not produce better results than NORMY in the end-to-end pipeline. We do not show these in Table VI for brevity.

G. Ablation Studies

The effectiveness of our model relies on some of the design choices we made. We investigate such choices our novel retriever *NORMYRetr* has from equation (2). We present the

ablation results in Table VI. Specifically, we show two ablation settings as follows:

NORMY w/o decay. We showed that if we disregard previously returned passages we may miss out on some relevant information required for subsequent modules. However, if we gave the same weight to previous passages as passages returned from current turn n, we see a degradation in performance. This is due to the current turn holding the most amount of information. Thus passages returned from the current turn should be given the most weight.

NORMY w/o sim. If a history turn is related to its previous turn, the passages returned will also have some similarities. Here, we disregarded the average pairwise similarity with the previous turn's returned passages from equation (2). We again see a decrease in model performance. By refining retriever scores with similarity score, we compute the relevance of passages with relevant conversational history. The ablation studies further verify that both decay and similarity scores are crucial for NORMY to perform best.

V. Related Work

Machine Reading Comprehension (MRC). MRC task typically includes a single-turn query where the answer grounds in a short passage. It started with TREC [5] in the early days where the goal was to retrieve the appropriate passage for 200 factoid questions and advanced to recent high-quality datasets like NQ [4], SQuAD [6], NewsQA [36].

Open Domain QA. Open domain QA introduces large corpus as grounded documents and the task is to retrieve the appropriate documents and then try to extract the answer span. For this task, high-quality datasets have been proposed such as TriviaQA [18], WikiQA [8], QuAX [37]. Some previous work [38], [39] selects answers from a closed set of passages or learns to rerank them. End-to-end open domain pipelines like DrQA [19] and BERTserini [40] use TFIDF/BM25 for the retrieval of passages and a neural reader to select the answer span. ORQA [25] and DPR [26] introduce a learnable retriever module with a dual encoder architecture. These works are all single-turn QA's whereas we target multi-turn conversations.

Conversational QA. Conversational QA is a variant of MRC where the queries are no longer single turn and the role of retrieval is disregarded [41]. The multi turn questions can be interconnected (CoQA [1], DoQA [42]), can depend on the previous history answer(QuAC [2]) or only limited to binary answers (ShARC [20]). A better understanding of the context of conversation history is needed to answer the grounded question. To capture the context, FlowQA [43] and GraphFlow [44] use each word as nodes in a graph and use an attention mechanism to represent the history; HAE [21] considers the history ground answers as context which is impractical for real life dialogue agents; Pos-HAE [45] considers the history turn positions as additional encoding. There are also backtracking based [14] and query rewriting based [15], [33] models as mentioned in previous sections.

Open Domain Conversational QA. ODQA models like OrConvQA [10], WS-OrConvQA [11], ConvADR-QA [12] do not perform any history modeling and use the same history window in all of their modules for the end-to-end system. Other ODQA works like TopiOCQA [46] do not focus on history modeling and use gold training data to train neural models for their pipeline. Similar to *Extractive QA* like this task (OrConvQA), there are *Abstractive* pipelines [47] where the answer is generated using transformers like BART [48] rather than being extracted from passages. Such pipelines are sequence-to-sequence tasks and not span predictions.

VI. Conclusion

We have presented the first end-to-end pipeline that uses non-uniform history modeling for open retrieval conversational question answering. We show that existing systems are suboptimal due to modeling the context in the same way for all modules and not utilizing previously returned passages for the *Retriever* module. We have also proposed a novel algorithm to utilize such passages to output higher-quality passages for subsequent modules. We further updated the doc2dial dataset to make it appropriate for OrConvQA task. Extensive experimental evaluation from various history modeling techniques with different types of data shows that NORMY significantly outperforms the state-of-the-art in each individual module and the entire pipeline.

Acknowledgment

This work was partially supported by NSF Grants IIS-1901379 and IIS-2227669.

References

[1] S. Reddy, D. Chen, and C. D. Manning, "Coqa: A conversational question answering challenge," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 249–266, 2019.

[2] E. Choi, H. He, M. Iyyer, M. Yatskar, W.-t. Yih, Y. Choi, P. Liang, and L. Zettlemoyer, "Quac: Question answering in context," *arXiv preprint arXiv:1808.07036*, 2018.

[3] M. Zaib, W. E. Zhang, Q. Z. Sheng, A. Mahmood, and Y. Zhang, "Conversational question answering: A survey," *Knowledge and Information Systems*, pp. 1–45, 2022.

[4] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee *et al.*, "Natural questions: a benchmark for question answering research," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 453–466, 2019.

[5] E. M. Voorhees, D. M. Tice *et al.*, "The trec-8 question answering track evaluation," in *TREC*, vol. 1999. Citeseer, 1999, p. 82.

[6] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," *arXiv preprint arXiv:1606.05250*, 2016.

[7] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen *et al.*, "Ms marco: A human generated machine reading comprehension dataset," *arXiv preprint arXiv:1611.09268*, 2016.

[8] D. Cohen, L. Yang, and W. B. Croft, "Wikipassageqa: A benchmark collection for research on non-factoid answer passage retrieval," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, 2018, pp. 1165–1168.

[9] B. Dhingra, K. Mazaitis, and W. W. Cohen, "Quasar: Datasets for question answering by search and reading," *arXiv preprint arXiv:1707.03904*, 2017.

[10] C. Qu, L. Yang, C. Chen, M. Qiu, W. B. Croft, and M. Iyyer, "Open-retrieval conversational question answering," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 539–548.

[11] C. Qu, L. Yang, C. Chen, W. B. Croft, K. Krishna, and M. Iyyer, "Weakly-supervised open-retrieval conversational question answering," in *European Conference on Information Retrieval*. Springer, 2021, pp. 529–543.

[12] H.-C. Fang, K.-H. Hung, C.-W. Huang, and Y.-N. Chen, "Open-domain conversational question answering with historical answers," *arXiv preprint arXiv:2211.09401*, 2022.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[14] M. Qiu, X. Huang, C. Chen, F. Ji, C. Qu, W. Wei, J. Huang, and Y. Zhang, "Reinforced history backtracking for conversational question answering," in *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI*, 2021, pp. 13 718–13 726.

[15] S. Vakulenko, S. Longpre, Z. Tu, and R. Anantha, "Question rewriting for conversational question answering," in *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021, pp. 355–363.

[16] I. Mele, C. I. Muntean, F. M. Nardini, R. Perego, N. Tonellotto, and O. Frieder, "Adaptive utterance rewriting for conversational search," *Information Processing & Management*, vol. 58, no. 6, p. 102682, 2021.

[17] H. Su, X. Shen, R. Zhang, F. Sun, P. Hu, C. Niu, and J. Zhou, "Improving multi-turn dialogue modelling with utterance rewriter," *arXiv preprint arXiv:1906.07004*, 2019.

[18] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension," *arXiv preprint arXiv:1705.03551*, 2017.

[19] D. Chen, A. Fisch, J. Weston, and A. Bordes, "Reading wikipedia to answer open-domain questions," *arXiv preprint arXiv:1704.00051*, 2017.

[20] M. Saeidi, M. Bartolo, P. Lewis, S. Singh, T. Rocktäschel, M. Sheldon, G. Bouchard, and S. Riedel, "Interpretation of natural language rules in conversational machine reading," *arXiv preprint arXiv:1809.01494*, 2018.

[21] C. Qu, L. Yang, M. Qiu, W. B. Croft, Y. Zhang, and M. Iyyer, "Bert with history answer embedding for conversational question answering," in *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, 2019, pp. 1133–1136.

[22] S. Feng, K. Fadnis, Q. V. Liao, and L. A. Lastras, "Doc2dial: a framework for dialogue composition grounded in documents," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 09, 2020, pp. 13 604–13 605.

[23] P. Christmann, R. S. Roy, and G. Weikum, "Conversational question answering on heterogeneous sources," *arXiv preprint arXiv:2204.11677*, 2022.

[24] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, M. Gatford *et al.*, "Okapi at trec-3," *Nist Special Publication Sp*, vol. 109, p. 109, 1995.

[25] K. Lee, M.-W. Chang, and K. Toutanova, "Latent retrieval for weakly supervised open domain question answering," *arXiv preprint arXiv:1906.00300*, 2019.

[26] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," *arXiv preprint arXiv:2004.04906*, 2020.

[27] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[28] R. Campos, V. Mangaravite, A. Pasquali, A. M. Jorge, C. Nunes, and A. Jatowt, "A text feature based automatic keyword extraction method for single documents," in *European conference on information retrieval*. Springer, 2018, pp. 684–691.

[29] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[30] P. M. Htut, S. R. Bowman, and K. Cho, "Training a ranking function for open-domain question answering," *arXiv preprint arXiv:1804.04264*, 2018.

[31] K. Clark and C. D. Manning, "Improving coreference resolution by learning entity-level distributed representations," *arXiv preprint arXiv:1606.01323*, 2016.

[32] A. Elgohary, D. Peskov, and J. Boyd-Graber, "Can you unpack that? learning to rewrite questions-in-context," *Can You Unpack That? Learning to Rewrite Questions-in-Context*, 2019.

[33] R. Anantha, S. Vakulenko, Z. Tu, S. Longpre, S. Pulman, and S. Chappidi, "Open-domain question answering goes conversational via question rewriting," *arXiv preprint arXiv:2010.04898*, 2020.

[34] F. Boudin, "pke: an open source python-based keyphrase extraction toolkit," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, Osaka, Japan, December 2016, pp. 69–73. [Online]. Available: http://aclweb.org/anthology/C16-2015

[35] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng, "Ms marco: A human generated machine reading comprehension dataset," in *CoCo@ NIPs*, 2016.

[36] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman, "Newsqa: A machine comprehension dataset," *arXiv preprint arXiv:1611.09830*, 2016.

[37] M. S. Rashid, F. Jamour, and V. Hristidis, "Quax: Mining the web for high-utility faq," in *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2021, pp. 1518–1527.

[38] B. Kratzwald and S. Feuerriegel, "Adaptive document retrieval for deep question answering," *arXiv preprint arXiv:1808.06528*, 2018.

[39] J. Lee, S. Yun, H. Kim, M. Ko, and J. Kang, "Ranking paragraphs for improving answer recall in open-domain question answering," *arXiv preprint arXiv:1810.00494*, 2018.

[40] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin, "End-to-end open-domain question answering with bertserini," *arXiv preprint arXiv:1902.01718*, 2019.

[41] P. Christmann, R. Saha Roy, A. Abujabal, J. Singh, and G. Weikum, "Look before you hop: Conversational question answering over knowledge graphs using judicious context expansion," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 729–738.

[42] J. A. Campos, A. Otegi, A. Soroa, J. Deriu, M. Cieliebak, and E. Agirre, "Doqa–accessing domain-specific faqs via conversational qa," *arXiv preprint arXiv:2005.01328*, 2020.

[43] H.-Y. Huang, E. Choi, and W.-t. Yih, "Flowqa: Grasping flow in history for conversational machine comprehension," *arXiv preprint arXiv:1810.06683*, 2018.

[44] Y. Chen, L. Wu, and M. J. Zaki, "Graphflow: Exploiting conversation flow with graph neural networks for conversational machine comprehension," *arXiv preprint arXiv:1908.00059*, 2019.

[45] C. Qu, L. Yang, M. Qiu, Y. Zhang, C. Chen, W. B. Croft, and M. Iyyer, "Attentive history selection for conversational question answering," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1391–1400.

[46] V. Adlakha, S. Dhuliawala, K. Suleman, H. de Vries, and S. Reddy, "Topiocqa: Open-domain conversational question answering with topic switching," *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 468–483, 2022.

[47] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[48] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.