# Improving RAG Systems via Sentence Clustering and Reordering

Marco Alessio[1,*], Guglielmo Faggioli[2], Nicola Ferro[2], Franco Maria Nardini[1] and Raffaele Perego[1]

[1]*Institute of Information Science and Technologies (ISTI), National Research Council of Italy (CNR), Pisa, Italy*
[2]*Department of Information Engineering (DEI), University of Padua, Padua, Italy*

**Abstract**
Large Language Models (LLMs) have gained noteworthy importance and attention across different domains and fields in recent years. Information Retrieval (IR) is one of the domains they impacted the most, as witnessed by the recent increase in the number of IR systems incorporating generative models. Specifically, Retrieval Augmented Generation (RAG) is the emerging paradigm that integrates existing knowledge from large-scale document corpora into the generation process, enabling the model to generate more coherent, contextually relevant, and accurate text across various tasks. Such tasks include summarization, question answering, and dialogue systems. Recent studies have highlighted the significant positional dependence exhibited by RAG systems. Such studies observed how the placement of information within the LLM input prompt drastically affects the generated output. We ground our study on this property by investigating alternative strategies for ordering sentences within the LLM prompt to improve the average quality of the generated responses in the user and conversational system dialogues. We propose the architecture of an end-to-end RAG-based conversational assistant and empirically evaluate our strategies using the TREC CAsT 2022 collection. Our experiments highlight significant differences between distinct arrangement strategies. By employing an evaluation methodology based on RankVicuna, we show that our best approach achieves improvements up to $54\%$ in terms of overall response quality over baseline methods.

**Keywords**
Retrieval Augmented Generation, Conversational Search, Positional Bias, Arrangement Strategy

## 1. Introduction

Retrieval Augmented Generation (RAG) is an emerging paradigm in the field of Artificial Intelligence (AI) to enhance the accuracy and reliability of generative models by exploiting external data sources. In recent years, RAG has gained noteworthy importance and attention across different domains and fields [1] as it allows to combine the strengths of Information Retrieval (IR) systems and generative models to overcome each other's limitations.

RAG can improve the output of a generative model in several ways. First, it allows the generation process to be grounded on information from trusted knowledge sources incorporated in the provided prompt, thus avoiding or at least mitigating the well-known Large Language Model (LLM) hallucination problem, i.e., when the model generates contents not factually true or that do not concern the prompted text [2, 3, 4]. Second, RAG allows for continuous knowledge updates and integration of domain-specific information: the LLM can successfully respond to facts and topics not covered in its training data; moreover, it is easily adapted to different scenarios and contexts, without retraining or fine-tuning the entire model using datasets that might be unavailable or limited in scope or size. Finally, grounding the generation process on external knowledge incorporated in the input permits linking the output to verifiable external documents, thus enhancing trustworthiness and transparency [2, 3, 4].

Current RAG systems, however, suffer of some drawbacks highlighted in the literature. One of these issues originates from the notable positional sensitivity shown by LLMs. The placement of information within the input prompt significantly impacts the resulting output. Previous research [5, 6, 7] has highlighted biases towards "primacy" and "recency", suggesting that generative models tend to prioritize information placed at the beginning or end of the input while neglecting the central portion.

In this paper, we advance over previous studies by investigating the positional bias in the context of RAG-based conversational systems. Specifically, we propose a novel strategy for arranging sentences within the input prompt of the LLM to improve the average quality of the generated responses over simpler methods. Our approach is based on the intuition that as coherent, fluent, and well-structured text are critical factors for successful communication between human beings, the same should also apply to LLMs: among all the possible arrangements of the input, those having sentences with similar meaning placed closer in the LLM prompt should generate, on average, better quality output. Therefore, we propose an end-to-end RAG architecture to test our hypothesis. The components of this architecture allow us to precisely identify which sentences are likely useful for answering user queries. To this end, we cluster sentences by their similarity and we define alternative strategies for ordering them both inter and intra-cluster. In this way, we can study the effect on the generated response of these alternatives for prompting the generative LLM. To our knowledge, this is the first work that explicitly considers this aspect and allows us to fine-tune in a principled way the ordering of input sentences provided to the generative component of a RAG system. We compare our proposed approach against competitive baselines that represent the solutions employed by current RAG systems. We experimentally evaluate the performance of our proposed approach using the TREC Conversational Assistance Track (CAsT) 2022 collection [8], which allows us to compare the results that different arrangement strategies can achieve in a widely accepted Conversational Search (CS) scenario. Results highlight remarkable differences among the tested sentence placement strategies, with improvements up to $8.66\%$ w.r.t. the best baseline and $54.94\%$ w.r.t. random ordering.

The remainder of this work is organized as follows: Section 2 surveys the current state-of-the-art about RAG systems and quality evaluation for their responses. Section 3 details the architecture of our RAG system. Section 4 and Section 5 detail the results of an experimental analysis,

*Information Retrieval's Role in RAG Systems (IR-RAG) - 2024, July 18, 2024, Washington, DC*
*Corresponding author.
marco.alessio@isti.cnr.it (M. Alessio); guglielmo.faggioli@unipd.it (G. Faggioli); nicola.ferro@unipd.it (N. Ferro); francomaria.nardini@isti.cnr.it (F. M. Nardini); raffaele.perego@isti.cnr.it (R. Perego)

which aims to highlight how the ordering of clusters and sentences affects the quality of the generated response. Finally, Section 6 draws some conclusions and outlines future directions and extensions of our research.

## 2. Related Work

In the following, we survey the main works dealing with LLM positional dependencies and the difficulties of RAG systems in conciliating internal and external knowledge. Then, we analyze the challenges related to the evaluation of the quality of RAG responses and to the use of an "LLM-as-a-judge".

### 2.1. Retrieval Augmented Generation

RAG enhances LLMs by retrieving additional information from an external knowledge source, enabling them to successfully answer queries beyond the scope of the training data. At the same time, RAG mitigates the hallucination problem, which is generating factually incorrect text, by referencing the provided external knowledge.

The RAG paradigm is organized into two main stages: retrieval and generation. Upon receiving a query from the user, the relevant information is retrieved from an external knowledge source. This task is undertaken by a standard IR pipeline that outputs a ranked list of documents. Afterwards, in the generation phase, the LLM synthesizes the response to answer the user query using the information carried by the selected documents.

Despite its clear advantages, RAG has drawbacks and limitations, which spark several challenges. First, RAG systems employ the external knowledge as their main source of information, disregarding the internal knowledge memorized within the LLM [9, 10]. This, in turn, may determine a decrease in the quality of the generated output when the provided content is not high-quality [10]. It is not uncommon for RAG to obtain worse outputs w.r.t. what the LLM can achieve in the closed-book scenario, i.e., without supplying retrieved results [10]. In this line, it has been observed that the LLM produces better results without injecting external knowledge when the topic popularity is very high [9]. In general, state-of-the-art LLMs provide good quality responses for a wide range of questions but require assistance from an IR system when the internal knowledge of the model lacks information about the current topic. This phenomenon is likely to occur if the topic is not very popular, requires exceptional expertise, or when scaling the number of parameters of the generative model produces little to no effect [9]. Another challenge lies in the significant positional dependence [5, 6, 7] exhibited by LLMs, whereby the placement of information within the input prompt drastically affects the generated output. Prior research [5] has identified "primacy" and "recency" biases, indicating the tendency of generative models to focus toward information positioned either at the beginning or the end of the input while disregarding the central part. Therefore, the performance degrades significantly when LLMs should rely on information in the middle of its input context, showing a characteristic U-shaped performance curve [5]. This, in turn, means that most state-of-the-art generative models do not use effectively their longer contexts w.r.t. smaller and earlier counterparts. These phenomena can be observed both in open-source, e.g., Llama [11, 12] by Meta, and closed-source, e.g., GPT-4 [13] by OpenAI, models. It is not advisable to directly input all the retrieved information to the LLM for generating the response. Redundant information and very long contextual data can interfere with the generation quality, leading to repetitive, disjointed, or incoherent outputs [1]. Therefore, the retrieved content is typically further processed before being given in input to the LLM [14]. A recent work in this direction systematically examines the retrieval strategy of RAG systems [15]. The authors consider multiple retrieval factors affecting the generation process, such as the relevance of the passages in the prompt context, their position, and their number. One counter-intuitive finding is that the retriever's highest-scoring documents that are not directly relevant to the query, e.g., do not contain the answer, negatively impact the effectiveness of the LLM. Moreover, the authors discover that adding random documents in the prompt improves the LLM accuracy by up to 35%.

In this work, we rely on the intuition that the use of coherent, fluent, and well-structured inputs can improve RAG and we propose an end-to-end architecture for selecting and structuring the external information included in the LLM prompt for response generation.

### 2.2. Quality Evaluation

Another line of research is how to evaluate the overall quality of the generation output. Despite human assessment providing the most accurate and reliable measure for evaluating model performance, the high time and cost requirements severely limit the application. Therefore, there exists an ever-increasing demand for automated evaluation techniques that consistently align with human judgements while offering enhanced efficiency and cost-effectiveness.

In this paper, we focus on textual-based generative models. Classical automatic evaluation metrics, such as BLEU [16], ROUGE [17], and METEOR [18], are designed to quantify the degree of similarity between a candidate text and one or more reference texts, by assessing their n-grams matching. The simplicity and explainability, along with the good correlation with human judgements, make these metrics widely used as baselines. However, these metrics exhibit several limitations [19]: firstly, they cannot account for lexical diversity; secondly, they penalize variations in the semantic ordering of words; thirdly, they struggle to capture and match paraphrases effectively; lastly, they inadequately account for distant dependencies within the text. With the advent of word embeddings [20, 21] and neural models [22, 23, 11, 12, 24] based on Transformers [25], new learned metrics [19, 26] have been developed. For example, BERTScore [19] can capture the semantic similarity between the candidate and reference texts employing the contextual embeddings generated by an encoder model, such as BERT [22].

In recent years, the rapid advancements of LLMs showing remarkable performance across many tasks have gained considerable interest in their potential application also as annotators and evaluators. Due to their training using Reinforcement Learning from Human Feedback (RLHF), these models demonstrate significant human alignment. Many research have investigated leveraging state-of-the-art LLMs to automatically produce assessments serving as proxies for human judgments, a paradigm known as "LLM-as-a-judge".

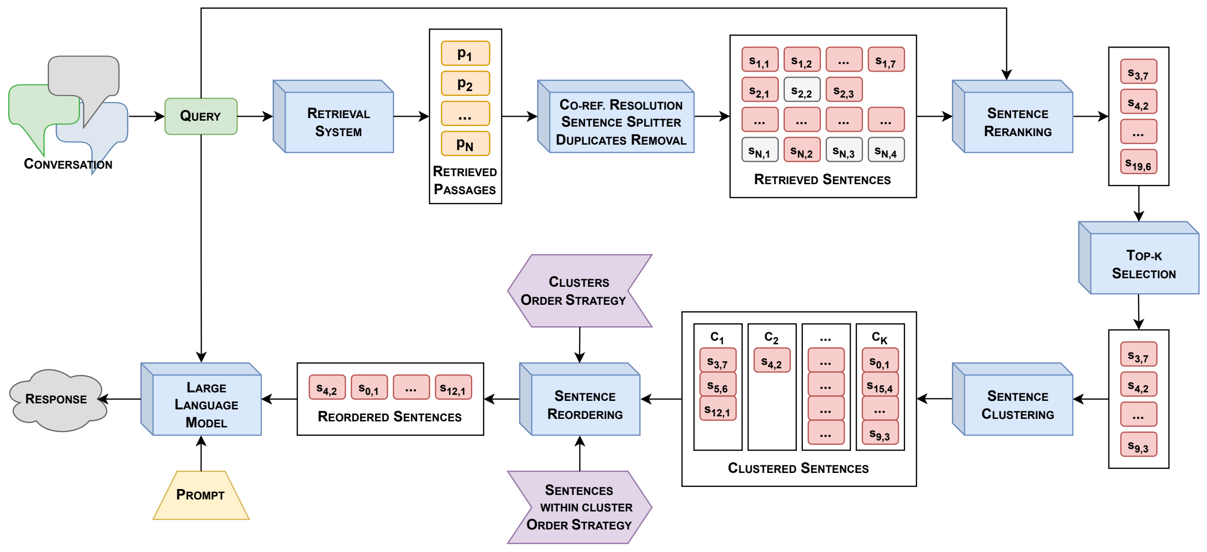

**Figure 1:** Architecture of our proposed RAG system.

Furthermore, in recent years LLMs have gained popularity also as evaluators. For example, Zheng et al. [24] assessed the quality of conversations with various LLMs, both open and closed source, employing GPT-4 [13] as judge. They experimented with various prompts and different approaches, such as single answer grading and pairwise comparisons both between responses and against a reference text. GPT-3.5 Turbo and GPT-4 [13] have been employed as listwise rerankers [6, 7] for the TREC Deep Learning 2019 and 2020 [27, 28] and BEIR [29] experimental collections, obtaining state-of-the-art performance [6]. The same LLMs have also been employed as teacher models to fine-tune smaller open-source student models, such as Llama and Vicuna [30, 31] (i.e.: RankVicuna [32]).

In this work, we rely on state-of-the-art assessment methods and evaluate the quality of the responses generated by the different methods using RankVicuna [32].

## 3. The Proposed RAG Architecture

Generative models exhibit strong biases towards information positioned at the start or the end of the input while disregarding the middle part [5]. This phenomenon motivates our research effort to determine how the order of the input sentences provided to a RAG-based conversational system affects the quality of the generated output and, in turn, the optimal ordering strategy to achieve the best response. This section describes each method and all variations considered in our experiments.

The architecture of our proposed RAG system is illustrated in Figure 1. It includes an IR pipeline, which retrieves top-$k$ documents $D = \{d_1, d_2, ..., d_k\}$ in response to each user utterance $q$. The retrieved documents are then processed by additional components responsible for splitting them into sentences, identifying the most relevant sentences, clustering such sentences based on their semantic similarity, and ordering them according to the various strategies analyzed. Finally, the selected—re-ordered—sentences are provided as input to the LLM for response generation. These components are the focus of our research. Their functionalities are detailed in the remainder of this section.

### 3.1. Document Pre-processing and Splitting

As observed in literature [33, 34], the entire text of a relevant document rarely contains meaningful knowledge to satisfy the user information need expressed by a query $q$. In most cases, only one or a few portions of the document are relevant to the query, while the remaining parts contain irrelevant information. The proposed architecture aims to precisely identify the key information in the retrieved documents, i.e., the sentences, to reduce the noise in the prompt used for response generation.

Hereinafter, we consider sentences in the documents as the atomic units of information. Our pipeline, illustrated in Figure 1 works as follows. First, for each query $q$ we consider only the top-$k$ documents $\{d_1, d_2, ..., d_k\}$ retrieved by the IR system. Then, a state-of-the-art co-reference resolution model is applied to all documents to replace pronouns and other generic terms within a sentence with the fully specified entity mentioned in a previous sentence. This allows us to remove the contextual dependencies among sentences in a document so they can be considered self-explanatory. The third step splits each document $d_i$ into a sequence of sentences $\{s_{i,1}, s_{i,2}, ..., s_{i,n_i}\}$. Afterwards, near-duplicate removal is employed to the sentences originated by all documents by discarding sentences with a Jaccard similarity $\geq 0.9$ between their Bag-of-Words (BoW) representations[1].

### 3.2. Sentence Selection

After the first pre-processing phase, we obtain a sentence candidate set for each query to be included in the LLM prompt of our RAG system (see Figure 1). Since the cardinality of this set can be large and not all the sentences are useful for answering the query, we employ the BERT-based cross encoder answer-in-the-sentence classifier[2] developed by Lajewska and Balog [35] to rank the candidate sentences

[1]This step is particularly important in our setting because the CAsT 2022 corpus contains a multitude of near-duplicate documents. In particular, the same Wikipedia article is often replicated in documents retrieved from the KILT and MS-MARCO collections.

[2]The model named “squad_snippets_unanswerable” is available at https://iai.group/downloads/emnlp2023-answerability_prediction.

according to their predicted usefulness to (at least partially) answer the query and we retain the top-$n$ ranked sentences thus discarding the remaining ones. As a possible limitation, please note that the model by Lajewska and Balog [35] employed have been trained on queries and passages used in our experiments. Therefore, it is very likely that the model performs significantly better on our data w.r.t. any other model, ensuring that top-ranked sentences are indeed relevant to the query. Even though such a model is not available in a real practical scenario, this choice is justified by our research effort being focused exclusively on comparing the ordering strategy for sentences in the LLM input rather than on the absolute results achievable by our RAG system.

### 3.3. Sentence Clustering and Ordering

The previous steps of the pipeline constrain the number of sentences per query while increasing their expected utility in answering the query. Furthermore, they allow us to control other noise sources, such as the number or the variable length of the retrieved documents. Therefore, we can assess how the positional bias affects the generation process. We highlight again that the positional bias of LLM has already been observed in prior research [5, 6, 7]. However, it has been considered exclusively as a limitation of LLMs and RAG systems. Our research moves a step forward by investigating the best ordering strategy to maximize, on average, the quality of the generated responses over a testing query set $Q$. We believe that logically organized text where sentences with akin meanings are positioned closer in the LLM prompt should, on average, yield superior output quality. Consequently, our sentence ordering strategies exploit the similarities among sentences selected by the sentence selection step. To measure semantic inter-sentence similarity, we resort to the contextualized embeddings generated with the *tct-colbert* model[3] [36]. We generate the representation of the $n$ selected sentences for each query and measure their pair-wise cosine similarity. Then, we progressively aggregate the most similar sentences by employing a hierarchical clustering algorithm. The maximum value of Silhouette statistic is used as the criteria to determine the optimal clustering among all possible. As a result, for each query $q \in Q$, the top-$n$ sentences are grouped in a variable number $N_c \geq 1$ of clusters, each composed of one or more sentences with similar semantic meaning. To devise different strategies for ordering input sentences, we leverage the above clustering that allows us to study the impact of sentence placement variations occurring in both inter and intra-clusters.

More formally, given a query, the set $S$ of the $n$ previously selected sentences, and the prompt $p$, we aim to find the ordering $ord^*$ of $S$ such that:

$$ord^* = \underset{ord}{\operatorname{argmax}} \sum_{q \in Q} s(q, LLM(p, q, ord(S))),$$

where $ord(S)$ is a sentence ordering strategy that returns an ordering of the sentences in $S$, $LLM(p, q, ord(S))$ is the response generated by the LLM used for prompt $p$, query $q$ and sentence ordering $ord(S)$, and, finally, $s(q, r)$ is a scoring function evaluating the perceived quality of the generated response $r = LLM(p, q, ord(S))$ for query $q$.

[3]https://huggingface.co/castorini/tct_colbert-v2-hnp-msmarco

The order of clusters and the order of the sentences within the same cluster uniquely determine the possible global ordering of the $n$ sentences we consider for inputting the LLM. Our experimental assessment will evaluate six different ordering strategies for placing the clusters of sentences in the input, and four different methods for ordering sentences within the same cluster. Cluster placements consider different aspects, such as the clusters' cardinality and similarity to the query. The ordering tested includes the random one and those obtained by decreasing/increasing the value of each aspect. Finally, the U-shaped order suggested in [5] is also tested. Regarding the ordering within clusters, we consider random order, order by reranker score, visiting order, and the clustering aggregation order.

## 4. Experimental Evaluation

We can now formulate the research questions we aim to answer with our experimental framework.

**Research Questions**. Given the sentence selection and clustering steps discussed above, the two main aspects to consider for defining our ordering strategies $ord(\cdot)$ are the order of placement in the LLM prompt of the clusters and of the sentences within the same cluster. They uniquely determine the global ordering $ord(\cdot)$ of the top-$n$ sentences given in input to the LLM for response generation. Our research questions assess which is the best solution among these alternatives considered. Specifically,

RQ1 What is the best cluster ordering strategy?
RQ2 What is the best ordering strategy for sentences within the same cluster?
RQ3 Can our proposed strategy enhance the effectiveness of the RAG system w.r.t. baseline methods?

**Experimental Settings**. We experiment with the TREC CAsT 2022 dataset, a standard experimental collection for CS [8]. This choice is due to prior research that released additional datasets, models, and human judgments for this benchmark [34, 35]. The corpus is composed of three documents collections, MS-MARCO v2 [37], KILT [38], and Washington Post v4, which are subdivided into $106M$ short documents. CAsT 2022 includes 18 information needs (topics) and 205 user utterances (queries), with an average length of $11.39$ user utterances per topic. The number of utterances for which relevance judgements are provided is $163$.

For our experiments, as the retrieval system, we employ as the output of the retrieval pipeline the best-performing run originally submitted to TREC CAsT 2022[4] [39]. This allows us to focus exclusively on the following steps of our pipeline. In all our experiments, we consider only the top-20 retrieved documents, leaving the investigation about the implications of this choice and possible alternatives as future work. To provide meaningful results, all queries where $Precision@20 < 0.2$, that is, having at most 3 relevant passages in the top-20 results, are discarded[5], ensuring that enough relevant information is retrieved to answer the considered queries successfully.

[4]The run is identified as "udinfo_mi_b2021" from the "udel_fang" group, University of Delaware (USA)

[5]The number of queries considered in these experiments is 115 out of 163 evaluated in the official relevance judgments.

**Table 1**
Comparisons between the six approaches proposed for RQ1: “What is the best ordering strategy for clusters?”. In the top half, each row reports three numbers, which are the wins for the approach in the column label, the ties, and the wins for the approach in the row label, respectively. In the bottom half, the overall results are reported.

| | **A vs.** | **B vs.** | **C vs.** | **D vs.** | **E vs.** | **F vs.** |
|---|---|---|---|---|---|---|
| **A** | — | 56-4-51 | 57-2-52 | **62-4-45** | 51-1-59 | 55-2-54 |
| **B** | 51-4-56 | — | 47-8-56 | **61-4-46** | 52-5-54 | 52-2-57 |
| **C** | 52-2-57 | 56-8-47 | — | **58-3-50** | 55-0-56 | 59-4-48 |
| **D** | 45-4-62 | 46-4-61 | 50-3-58 | — | 44-1-66 | 47-1-63 |
| **E** | 59-1-51 | 54-5-52 | 56-0-55 | **66-1-44** | — | 57-5-49 |
| **F** | 54-2-55 | 57-2-52 | 48-4-59 | **63-1-47** | 49-5-57 | — |
| **Overall** | 261-13 | 269-23 | 258-17 | **310-13** | 251-12 | 270-14 |
| **Avg. Score** | 0.5723 | 0.5844 | 0.5510 | **0.6219** | 0.5736 | 0.5969 |

**Table 2**
Comparisons between the four approaches proposed for RQ2: “What is the best ordering strategy for sentences within the same cluster?”. In the top half, each row reports three numbers, which are the wins for the approach in the column label, the ties, and the wins for the approach in the row label, respectively. In the bottom half, the overall results are reported.

| | **A vs.** | **B vs.** | **C vs.** | **D vs.** |
|---|---|---|---|---|
| **A** | — | 53-3-59 | 48-8-59 | **55-4-56** |
| **B** | 59-3-53 | — | 54-3-58 | **60-7-48** |
| **C** | 59-8-48 | 58-3-54 | — | **57-6-52** |
| **D** | 56-4-55 | 48-7-60 | 52-6-57 | — |
| **Overall** | **174-15** | 159-13 | 154-17 | **172-17** |
| **Avg. Score** | 0.6281 | 0.6143 | 0.6124 | **0.6451** |

Furthermore, in the steps of the pipeline where the query text is needed, i.e., sentence ranking and response generation, we employed the manually rewritten text for every query. This allows us to account for the possible bias introduced by different query rewriting approaches. Future developments will investigate the relationship between query rewriting approaches and RAG solutions.

For co-reference resolution at the document level, i.e., removing co-references across different sentences in the “document processing” step, we use the “F-Coref” model[6] [40] based on the “LingMess” architecture [41]. After this step, we use the well-known SpaCy Python library to divide each document into a sequence of independent sentences.

In the following section, we report two different metrics for each comparison. The former is the average score of every approach when assessing all 10 random permutations using RankVicuna. The latter, instead, is a pairwise metric, assessing the number of queries for which the first approach obtains higher/the same/lower score w.r.t. the other one. This information should better highlight the differences and provide a more comprehensive view than a single average value.

**Response Generation**. For the response generation, we employ Vicuna 7B[7] [24], a LLM based on Llama 2 [11, 12] fine-tuned on 125K user conversations with ChatGPT gathered using public APIs from the ShareGPT.com website.

**Quality Evaluation**. To evaluate the quality of the generated responses, we employ RankVicuna [32] to perform listwise ranking between all responses being compared. To mitigate the positional bias intrinsic in RankVicuna, we assess 10 different random permutations of the same responses, averaging the results obtained. This is a reasonable trade-off between evaluation accuracy and the computational runtime required. For each assessment, we assign $\frac{N+1-i}{N}$ points to the i-th ranked response, where $1 \leq i \leq N$ and $N$ is the number of responses being compared. Furthermore, we also evaluate the number of wins and ties between pairs of responses considered. Whether a valid judgment from the LLM can not be determined, the entire comparison is discarded from the evaluation.

## 4.1. RQ1: Order of Clusters

For the first experiment, we evaluate the effects of different ordering of the clusters while keeping the order of sentences within the same cluster (based on the clustering aggregation order) fixed. We test six different strategies for ordering clusters: clusters selected in random order (strategy A); clusters selected in descending order of cardinality (strategy B); clusters selected in ascending order of similarity with the query[8] (strategy C); clusters selected in descending order of similarity with the query (strategy D); clusters selected in descending order by similarity with the query using a ping-pong layout from top to bottom (strategy E)[9]; clusters selected by similarity with the query in descending order, using a ping-pong layout from bottom to top (strategy F)[10].

As shown in Table 1, sorting the clusters in descending order by their similarity with the query (strategy D) is the clear winner in this comparison, in terms of both score and pairwise wins. This approach performs $18.77\%$, $15.24\%$, $20.16\%$, $23.51\%$, and $14.81\%$ better than other options. This figures suggest that the LLM used to generate the responses exhibit a much stronger “primacy” rather than “recency” biases, as highlighted by option C being overall the worst performing among those considered. Instead, methods E and F were designed to place the least important clusters towards the center, since LLMs struggle to utilize the information in the middle of their prompt effectively. However, we can see that both approaches are ineffective: we suspect this is due to the length of the input text being much smaller than the maximum context window of the model. Different results may be observed when varying the amount of input data provided to the LLM for generation.

## 4.2. RQ2: Order of Sentences within the same Cluster

In this second experiment, we evaluate different sorting schemes for sentences within the same cluster, keeping the cluster’s order fixed at the best strategy determined in RQ1. We test four different strategies for ordering sentences within the same cluster: sentences selected in random order (strategy A); sentences selected in descending order by reranker score (strategy B); sentences selected by visiting order[11] (strategy C); sentences selected by aggregation order (strategy D).

As shown in Table 2, the best results are achieved by two

[6] https://huggingface.co/biu-nlp/f-coref
[7] https://huggingface.co/lmsys/vicuna-7b-v1.5

[8] The similarity between a cluster $C$ and the query is defined as the maximum cosine similarity between the query $q \in Q$ with any sentence $s_{i,j} \in C$ belonging to the cluster.
[9] The clusters are placed first, last, second, second-to-last, third, and so on, e.g., [A, B, C, D, E] becomes [A, C, E, D, B].
[10] The clusters are placed last, first, second-to-last, second, third-to-last, and so on, e.g., [A, B, C, D, E] becomes [B, D, E, C, A].
[11] The sentences are sorted based on the order in which they appear when sequentially scanning through the set of top-$k$ retrieved documents.

**Table 3**
Comparisons between the five approaches considered for RQ3: "Can our proposed strategy enhance the effectiveness of the RAG system w.r.t. baseline methods?". In the top half, each row reports three numbers, which are the wins for approach in the column label, the ties, and the wins for approach in the row label, respectively. In the bottom half, the overall results are reported.

| | **A vs.** | **B vs.** | **C vs.** | **D vs.** | **CL vs.** |
|---|---|---|---|---|---|
| **A** | — | 45-4-62 | 54-1-56 | 54-0-57 | **66-2-43** |
| **B** | 62-4-45 | — | 71-1-39 | 64-8-39 | **67-5-39** |
| **C** | 56-1-54 | 39-1-71 | — | 50-4-57 | **59-3-49** |
| **D** | 57-0-54 | 39-8-64 | 57-4-50 | — | **59-3-49** |
| **CL** | 43-2-66 | 39-5-67 | 49-3-59 | 49-3-59 | — |
| **Overall** | 218-7 | 162-18 | 231-9 | 217-15 | **251-13** |
| **Avg. Score** | 0.5882 | 0.5533 | 0.6177 | 0.6016 | **0.6392** |

different strategies: option D, sorting sentences within the same cluster based on aggregation order, and interestingly, option A, randomly sorting the sentences. Both strategies are preferable to the other two methods considered, performing $8.18\%$ and $11.69\%$ better w.r.t. options B and C, respectively. We note however that the difference in performance of the various strategies are not large as the sentences are grouped in the clusters by their similarity. The LLM response appears to be more impacted by the order of the clusters than by the order of sentences within each cluster.

## 4.3. RQ3: Comparison with Baselines

Our last experiment investigates whether our proposed approach is beneficial in enhancing the overall effectiveness of the RAG system w.r.t. four simpler baseline methods that may be used in practice by current state-of-the-art RAG systems. We test five different strategies: i) the top-5 retrieved documents (A), ii) the top-40 sentences taken in random order (B), iii) the top-40 sentences taken in descending order by re-ranker score (C), iv) the top-40 sentences selected by visiting order (D), v) the best clusterization-based approach determined from RQ1 and RQ2 (CL).

The results obtained are shown in Table 3. The clusterization-based approach demonstrate superior performance, resulting as the best strategy in this comparison. The four baselines yield notably lower results: $15.14\%$, $54.94\%$, $8.66\%$, and $15.67\%$, respectively. Among the methods considered in this work, randomly sorting the top-$h$ sentences is by far the least performing approach. This, in turn, proves our starting intuition about coherent, fluent, and well-structured text being critical factors for LLMs to generate high quality output.

## 5. Additional Experiments

The clusterization-based ordering strategy proposed in this work is designed to position sentences sharing analogous semantic content close together in the LLM prompt. Given the results obtained in Section 4.3, we have shown its effectiveness in our experimental settings. Nevertheless, we answer two additional research questions in this section to gain additional insights. Specifically,

RQ4 Is there a correlation between the similarity of subsequent sentences in the LLM prompt and the quality of the generated response?

**Table 4**
Comparisons between the seven approaches proposed for RQ4: "Is there a correlation between the similarity of subsequent sentences in the LLM prompt and the quality of the generated response?". In the top half, each row reports three numbers, which are the wins for the approach in the column label, the ties, and the wins for the approach in the row label, respectively. In the bottom half, the overall results are reported.

| | **1.000 vs.** | **0.625 vs.** | **0.500 vs.** | **0.375 vs.** | **0.250 vs.** | **0.125 vs.** | **0.000 vs.** |
|---|---|---|---|---|---|---|---|
| **1.000** | — | 46-2-43 | 38-2-51 | 45-0-46 | 40-2-49 | 40-1-50 | 38-1-52 |
| **0.625** | 43-2-46 | — | 37-2-52 | 42-2-47 | 41-1-49 | 36-0-55 | 35-1-55 |
| **0.500** | 51-2-38 | 52-2-37 | — | 51-2-38 | 52-0-39 | 37-0-54 | 44-2-45 |
| **0.375** | 46-0-45 | 47-2-42 | 38-2-51 | — | 42-2-47 | 37-3-51 | 37-1-53 |
| **0.250** | 49-2-40 | 49-1-41 | 39-0-52 | 47-2-42 | — | 43-3-45 | 42-1-48 |
| **0.125** | 50-1-40 | 55-0-36 | 54-0-37 | 51-3-37 | 45-3-43 | — | 44-1-46 |
| **0.000** | 52-1-38 | 55-1-35 | 45-2-44 | 53-1-37 | 48-1-42 | 46-1-44 | — |
| **Overall** | 291-8 | **304-8** | 251-8 | 289-10 | 268-9 | 239-8 | 240-7 |
| **Avg. Score** | 0.5731 | **0.5866** | 0.5480 | 0.5617 | 0.5516 | 0.5349 | 0.5143 |

RQ5 Is the proposed clusterization strategy more effective than directly optimising the similarity of subsequent sentences?

**Experimental Settings**. We determine heuristically the two ordering $ord^{+}$ and $ord^{-}$, which maximize and minimize the overall similarity between subsequent sentences. Let $sum^{+}$ and $sum^{-}$ be the sum of similarity between subsequent sentences for $ord^{+}$ and $ord^{-}$ respectively. The similarity $sim(p)$ for a sentence permutation $p$ is given by the following equation, where min-max normalization is used, and $s_i$ are the embedding representations of the respective sentences:

$$sim(p) = \frac{\left(\sum_{i=2}^{h} cos(s_{i-1}, s_i)\right) - sum^{-}}{sum^{+} - sum^{-}}$$

In our experiments, for each query, we generate one million random permutations, then we determine which is the permutation with similarity closer to each of the following thresholds: $0.125$, $0.250$, $0.375$, $0.500$, and $0.625$. We decided to stop at $0.625$ because higher values are unlikely to be observed given that the average similarity of these permutations is $0.3433$ with standard deviation $0.0530$.

**Results**. We determine how the quality of the generated response is influenced when varying the similarity between subsequent sentences at various predefined thresholds, as shown in Table 4. It is interesting to note that the highest results are obtained by permutations with $0.625$ normalised similarity, rather than $1.000$ which is the ordering maximising the similarity between subsequent sentences ($ord^{+}$). This method achieves 4.47% and 26.67% more pairwise wins w.r.t. $ord^{+}$ and $ord^{-}$, respectively. To answer RQ5, we assess the responses generated using the best clustering strategy against the approach defined above. The average scores are $0.7652$ and $0.7348$ while the pairwise wins and ties are 38 - 46 - 31, respectively.

From these experiments, we can conclude that a positive correlation exists between similarity between subsequent sentences and response quality, while proving that sentence similarity may not be the only factor that should be considered. Moreover, subdividing and explicitly grouping together sentences by subtopic is beneficial w.r.t. considering the sentence similarity only in a pairwise fashion and thus lacking a global vision of the retrieved knowledge.

## 6. Conclusions and Future Work

In this work, we presented a novel pipelined RAG architecture aimed at selecting a set of relevant sentences for each query and arranging them in a specific order to optimize the quality of responses generated by a LLM. For this purpose, sentences are first extracted from the top documents retrieved. Then, they are reranked, and the most relevant sentences are organized in clusters by similarity. We proposed different strategies for ordering clusters and the sentences within clusters in the input given to the LLM for response generation. To the best of our knowledge, this is the first work investigating sentence clustering and re-ordering to improve the quality of the response generated by RAG systems. Our empirical assessment is based on a well-known—public—framework for conversational search. The results of the experiments show that different sequences of sentences in the LLM prompt significantly impact response quality despite all methodologies processing identical information from the same set of sentences. Random permutations yield the lowest results, whereas our proposed approach based on sentence clusterization yields superior results. Additionally, we examined whether maximizing the similarity between consecutive sentences in the LLM prompt enhances response quality. While a positive correlation between these factors was observed, it is not the exclusive determinant. Consequently, while we infer that sentence similarity constitutes a pivotal aspect, other contributing factors remain unidentified, warranting further investigation. Moreover, although our experimental evaluation employs a well-known conversational collection, the methodology and results shown in this work are general. They could also be applied to other scenarios, such as ad-hoc search.

In future work, we intend to evaluate the impact of the number of clusters selected by our method for generating the response. Our intuition is that the number of clusters identified for a given query is a proxy of the difficulty of the query itself. Fewer clusters or even a single large should characterize simple and close queries. In contrast, difficult—multi-faceted—queries are possibly characterized by more clusters, each addressing a different facet of the query. This intuition paves the way for the extension of the evaluation methodology by adopting diversification-based metrics [42], allowing us to understand how well the generated answers cover the query facets and the topical distribution of the clusters.

## References

[1] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, M. Wang, H. Wang, Retrieval-augmented generation for large language models: A survey, 2024. arXiv:2312.10997.

[2] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, T. Liu, A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, CoRR abs/2311.05232 (2023). URL: https://doi.org/10.48550/arXiv.2311.05232. doi:10.48550/ARXIV.2311.05232. arXiv:2311.05232.

[3] Y. Zhang, Y. Li, L. Cui, D. Cai, L. Liu, T. Fu, X. Huang, E. Zhao, Y. Zhang, Y. Chen, L. Wang, A. T. Luu, W. Bi, F. Shi, S. Shi, Siren's song in the AI ocean: A survey on hallucination in large language models, CoRR abs/2309.01219 (2023). URL: https://doi.org/10.48550/arXiv.2309.01219. doi:10.48550/ARXIV.2309.01219. arXiv:2309.01219.

[4] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, P. Fung, Survey of hallucination in natural language generation, ACM Comput. Surv. 55 (2023) 248:1–248:38. URL: https://doi.org/10.1145/3571730. doi:10.1145/3571730.

[5] N. F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, CoRR abs/2307.03172 (2023). URL: https://doi.org/10.48550/arXiv.2307.03172. doi:10.48550/ARXIV.2307.03172. arXiv:2307.03172.

[6] W. Sun, L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Is chatgpt good at search? investigating large language models as re-ranking agents, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023, Association for Computational Linguistics, 2023, pp. 14918–14937. URL: https://doi.org/10.18653/v1/2023.emnlp-main.923. doi:10.18653/V1/2023.EMNLP-MAIN.923.

[7] R. Tang, X. Zhang, X. Ma, J. Lin, F. Ture, Found in the middle: Permutation self-consistency improves listwise ranking in large language models, CoRR abs/2310.07712 (2023). URL: https://doi.org/10.48550/arXiv.2310.07712. doi:10.48550/ARXIV.2310.07712. arXiv:2310.07712.

[8] P. Owoicho, J. Dalton, M. Aliannejadi, L. Azzopardi, J. R. Trippas, S. Vakulenko, TREC cast 2022: Going beyond user ask and system retrieve with initiative and response generation, in: I. Soboroff, A. Ellis (Eds.), Proceedings of the Thirty-First Text REtrieval Conference, TREC 2022, online, November 15-19, 2022, volume 500-338 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 2022. URL: https://trec.nist.gov/pubs/trec31/papers/Overview_cast.pdf.

[9] A. Mallen, A. Asai, V. Zhong, R. Das, D. Khashabi, H. Hajishirzi, When not to trust language models: Investigating effectiveness of parametric and non-parametric memories, in: A. Rogers, J. L. Boyd-Graber, N. Okazaki (Eds.), Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023, Association for Computational Linguistics, 2023, pp. 9802–9822. URL: https://doi.org/10.18653/v1/2023.acl-long.546. doi:10.18653/V1/2023.ACL-LONG.546.

[10] R. Ren, Y. Wang, Y. Qu, W. X. Zhao, J. Liu, H. Tian, H. Wu, J. Wen, H. Wang, Investigating the factual knowledge boundary of large language models with retrieval augmentation, CoRR abs/2307.11019 (2023). URL: https://doi.org/10.48550/arXiv.2307.11019. doi:10.48550/ARXIV.2307.11019. arXiv:2307.11019.

[11] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, G. Lample, Llama: Open and efficient foundation language models, CoRR abs/2302.13971 (2023). URL: https://doi.org/10.48550/arXiv.2302.13971. doi:10.48550/ARXIV.2302.13971. arXiv:2302.13971.

[12] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Canton-Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, T. Scialom, Llama 2: Open foundation and fine-tuned chat models, CoRR abs/2307.09288 (2023). URL: https://doi.org/10.48550/arXiv.2307.09288. doi:10.48550/ARXIV.2307.09288. arXiv:2307.09288.

[13] OpenAI, GPT-4 technical report, CoRR abs/2303.08774 (2023). URL: https://doi.org/10.48550/arXiv.2303.08774. doi:10.48550/ARXIV.2303.08774. arXiv:2303.08774.

[14] F. Xu, W. Shi, E. Choi, RECOMP: improving retrieval-augmented lms with compression and selective augmentation, CoRR abs/2310.04408 (2023). URL: https://doi.org/10.48550/arXiv.2310.04408. doi:10.48550/ARXIV.2310.04408. arXiv:2310.04408.

[15] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonellotto, F. Silvestri, The power of noise: Redefining retrieval for rag systems, arXiv preprint arXiv:2401.14887 (2024).

[16] K. Papineni, S. Roukos, T. Ward, W. Zhu, Bleu: a method for automatic evaluation of machine translation, in: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA, ACL, 2002, pp. 311–318. URL: https://aclanthology.org/P02-1040/. doi:10.3115/1073083.1073135.

[17] C.-Y. Lin, ROUGE: A package for automatic evaluation of summaries, in: Text Summarization Branches Out, Association for Computational Linguistics, Barcelona, Spain, 2004, pp. 74–81. URL: https://aclanthology.org/W04-1013.

[18] S. Banerjee, A. Lavie, METEOR: an automatic metric for MT evaluation with improved correlation with human judgments, in: J. Goldstein, A. Lavie, C. Lin, C. R. Voss (Eds.), Proceedings of the Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization@ACL 2005, Ann Arbor, Michigan, USA, June 29, 2005, Association for Computational Linguistics, 2005, pp. 65–72. URL: https://aclanthology.org/W05-0909/.

[19] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, Y. Artzi, Bertscore: Evaluating text generation with BERT, in: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020, OpenReview.net, 2020. URL: https://openreview.net/forum?id=SkeHuCVFDr.

[20] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: Y. Bengio, Y. LeCun (Eds.), 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings, 2013. URL: http://arxiv.org/abs/1301.3781.

[21] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: A. Moschitti, B. Pang, W. Daelemans (Eds.), Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, ACL, 2014, pp. 1532–1543. URL: https://doi.org/10.3115/v1/d14-1162. doi:10.3115/V1/D14-1162.

[22] J. Devlin, M. Chang, K. Lee, K. Toutanova, BERT: pre-training of deep bidirectional transformers for language understanding, in: J. Burstein, C. Doran, T. Solorio (Eds.), Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), Association for Computational Linguistics, 2019, pp. 4171–4186. URL: https://doi.org/10.18653/v1/n19-1423. doi:10.18653/V1/N19-1423.

[23] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, J. Mach. Learn. Res. 21 (2020) 140:1–140:67. URL: http://jmlr.org/papers/v21/20-074.html.

[24] L. Zheng, W. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. P. Xing, H. Zhang, J. E. Gonzalez, I. Stoica, Judging llm-as-a-judge with mt-bench and chatbot arena, in: A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, S. Levine (Eds.), Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023, 2023. URL: http://papers.nips.cc/paper_files/paper/2023/hash/91f18a1287b398d378ef22505bf41832-Abstract-Datasets_and_Benchmarks.html.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, R. Garnett (Eds.), Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, 2017, pp. 5998–6008. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

[26] E. Clark, S. Rijhwani, S. Gehrmann, J. Maynez, R. Aharoni, V. Nikolaev, T. Sellam, A. Siddhant, D. Das, A. P. Parikh, SEAHORSE: A multilingual, multifaceted dataset for summarization evaluation, in: H. Bouamor, J. Pino, K. Bali (Eds.), Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023, Association for Computational Linguistics, 2023, pp. 9397–9413. URL: https://doi.org/10.18653/v1/2023.emnlp-main.584. doi:10.18653/V1/2023.EMNLP-MAIN.584.

[27] N. Craswell, B. Mitra, E. Yilmaz, D. Campos, E. M. Voorhees, Overview of the TREC 2019 deep learning track, CoRR abs/2003.07820 (2020). URL: https://arxiv.org/abs/2003.07820. arXiv:2003.07820.

[28] N. Craswell, B. Mitra, E. Yilmaz, D. Campos, Overview of the TREC 2020 deep learning track, in: E. M. Voorhees, A. Ellis (Eds.), Proceedings of the Twenty-Ninth Text REtrieval Conference, TREC 2020, Virtual Event [Gaithersburg, Maryland, USA], November 16-20, 2020, volume 1266 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 2020. URL: https://trec.nist.gov/pubs/trec29/papers/OVERVIEW.DL.pdf.

[29] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, I. Gurevych, BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models, CoRR abs/2104.08663 (2021). URL: https://arxiv.org/abs/2104.08663. arXiv:2104.08663.

[30] X. Ma, X. Zhang, R. Pradeep, J. Lin, Zero-shot listwise document reranking with a large language model, CoRR abs/2305.02156 (2023). URL: https://doi.org/10.48550/arXiv.2305.02156. doi:10.48550/ARXIV.2305.02156. arXiv:2305.02156.

[31] W. Sun, Z. Chen, X. Ma, L. Yan, S. Wang, P. Ren, Z. Chen, D. Yin, Z. Ren, Instruction distillation makes large language models efficient zero-shot rankers, CoRR abs/2311.01555 (2023). URL: https://doi.org/10.48550/arXiv.2311.01555. doi:10.48550/ARXIV.2311.01555. arXiv:2311.01555.

[32] R. Pradeep, S. Sharifymoghaddam, J. Lin, Rankvicuna: Zero-shot listwise document reranking with open-source large language models, CoRR abs/2309.15088 (2023). URL: https://doi.org/10.48550/arXiv.2309.15088. doi:10.48550/ARXIV.2309.15088. arXiv:2309.15088.

[33] P. Ren, Z. Chen, Z. Ren, E. Kanoulas, C. Monz, M. de Rijke, Conversations with search engines: Serp-based conversational response generation, ACM Trans. Inf. Syst. 39 (2021) 47:1–47:29. URL: https://doi.org/10.1145/3432726. doi:10.1145/3432726.

[34] W. Lajewska, K. Balog, Towards filling the gap in conversational search: From passage retrieval to conversational response generation, in: I. Frommholz, F. Hopfgartner, M. Lee, M. Oakes, M. Lalmas, M. Zhang, R. L. T. Santos (Eds.), Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, CIKM 2023, Birmingham, United Kingdom, October 21-25, 2023, ACM, 2023, pp. 5326–5330. URL: https://doi.org/10.1145/3583780.3615132. doi:10.1145/3583780.3615132.

[35] W. Lajewska, K. Balog, Towards reliable and factual response generation: Detecting unanswerable questions in information-seeking conversations, in: N. Goharian, N. Tonellotto, Y. He, A. Lipani, G. McDonald, C. Macdonald, I. Ounis (Eds.), Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR 2024, Glasgow, UK, March 24-28, 2024, Proceedings, Part III, volume 14610 of *Lecture Notes in Computer Science*, Springer, 2024, pp. 336–344. URL: https://doi.org/10.1007/978-3-031-56063-7_25. doi:10.1007/978-3-031-56063-7_25.

[36] S. Lin, J. Yang, J. Lin, In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval, in: A. Rogers, I. Calixto, I. Vulic, N. Saphra, N. Kassner, O. Camburu, T. Bansal, V. Shwartz (Eds.), Proceedings of the 6th Workshop on Representation Learning for NLP, RepL4NLP@ACL-IJCNLP 2021, Online, August 6, 2021, Association for Computational Linguistics, 2021, pp. 163–173. URL: https://doi.org/10.18653/v1/2021.repl4nlp-1.17. doi:10.18653/V1/2021.REPL4NLP-1.17.

[37] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng, MS MARCO: A human generated machine reading comprehension dataset, in: T. R. Besold, A. Bordes, A. S. d'Avila Garcez, G. Wayne (Eds.), Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016, volume 1773 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2016. URL: http://ceur-ws.org/Vol-1773/CoCoNIPS_2016_paper9.pdf.

[38] F. Petroni, A. Piktus, A. Fan, P. S. H. Lewis, M. Yazdani, N. D. Cao, J. Thorne, Y. Jernite, V. Karpukhin, J. Maillard, V. Plachouras, T. Rocktäschel, S. Riedel, KILT: a benchmark for knowledge intensive language tasks, in: K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, Y. Zhou (Eds.), Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021, Association for Computational Linguistics, 2021, pp. 2523–2544. URL: https://doi.org/10.18653/v1/2021.naacl-main.200. doi:10.18653/V1/2021.NAACL-MAIN.200.

[39] D. Yang, Y. Zhang, H. Fang, An exploration study of mixed-initiative query reformulation in conversational passage retrieval, in: I. Soboroff, A. Ellis (Eds.), Proceedings of the Thirty-First Text REtrieval Conference, TREC 2022, online, November 15-19, 2022, volume 500-338 of *NIST Special Publication*, National Institute of Standards and Technology (NIST), 2022. URL: https://trec.nist.gov/pubs/trec31/papers/udel_fang.C.pdf.

[40] S. Otmazgin, A. Cattan, Y. Goldberg, F-coref: Fast, accurate and easy to use coreference resolution, in: Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - System Demostrations, Taipei, Taiwan, November 20 - 23, 2022, Association for Computational Linguistics, 2022, pp. 48–56. URL: https://aclanthology.org/2022.aacl-demo.6.

[41] S. Otmazgin, A. Cattan, Y. Goldberg, Lingmess: Linguistically informed multi expert scorers for coreference resolution, in: A. Vlachos, I. Augenstein (Eds.), Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2023, Dubrovnik, Croatia, May 2-6, 2023, Association for Computational Linguistics, 2023, pp. 2744–2752. URL: https://doi.org/10.18653/v1/2023.eacl-main.202. doi:10.18653/V1/2023.EACL-MAIN.202.

[42] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, I. MacKinnon, Novelty and diversity in information retrieval evaluation, in: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '08, Association for Computing Machinery, New York, NY, USA, 2008, p. 659–666. URL: https://doi.org/10.1145/1390334.1390446. doi:10.1145/1390334.1390446.