Universiteit Leiden

Master Computer Science

Abstractive Summarization for Large Collections of Scientific Articles

Name:	Pavlos Zakkas
Student ID:	s3264009
Date:	12/07/2023
Specialisation:	Artificial Intelligence
1st supervisor:	Dr. Suzan Verberne
2nd supervisor:	Jakub Zavrel

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

We propose a new a prompt-based end-to-end system for extreme summarization of large collections of scientific documents. Digesting scientific information is becoming more challenging, especially in high volume fast-paced fields like AI. Our system helps researchers find information faster and stay up to date with their field of interest using summaries generated by instruction-tuned Large Language Models (LLMs). Although LLMs have already been applied to several downstream tasks including news summarization, its effectiveness in the scientific domain and especially in summarizing large collections of papers, is underexplored.

The input is a collection of scientific papers, e.g. thousands of papers published in a large conference. Our pipeline first groups these source documents using a graph-based clustering algorithm, selects representative papers per cluster based on their diversity and then it performs a two-step summarization. Using an instruction-tuned LLM as backbone, single-document summaries (SDS) of the selected papers are generated in the first step and their aggregation is performed in the second multi-document summarization (MDS) step. In the end, the cluster summaries are used to generate a full blog post, with the clusters as sections.

We evaluate the summarization components of our system in isolation using available SDS and MDS datasets in the scientific domain. For the SDS tasks, our system achieves better results than strong baselines: the fine-tuned models in the SciTLDR benchmark. For the MDS tasks, our two-step approach not only leads to a performance gain over a one-step approach where the full content of the source documents is passed in a single prompt to the LLM, but also reaches the performance of state-of-the-art fine-tuned models on the Multi-XScience dataset. For the end-to-end evaluation of our pipeline, we performed a small user study to assess the performance of our system in generating blog posts that summarize a whole AI conference. Preference judgments indicate that selecting representative papers before prompting the LLM for the summarization of a cluster is preferred by humans over using single-document summaries of all the papers of the cluster. Although a human-written blog post is still preferred over the automated one, the users appreciate the good informativeness and factuality of our method.

1 Introduction

The digestion of large amounts of text data is getting more and more emergent, especially in fast-paced fields such as Artificial Intelligence (AI). Meanwhile, the time that we can allocate to consume this amount of information is limited. For that reason, extreme summarization (Narayan et al., 2018) is an active area of research that aims at creating a short one-sentence summary about the main idea of an article. By transferring this idea to the multi-document task, we can argue that summarizing large collections of documents while keeping the output both short and informative would be a useful way to help researchers seek information faster and stay up-to-date with the domain of their interest. For instance, gathering insights from a conference with thousands of published scientific articles seems quite a challenging task. Producing an automated summary of such a conference could save a significant amount of time, as it can provide the general essence of the conference, helping the researcher to focus on the parts of their preference.

The rapid advancement in Natural Language Processing (NLP), especially with the release of transformers (Vaswani et al., 2017), has led to significant improvements in abstractive summarization of documents. Depending on the length and the domain of input documents, several approaches have been explored, varying from encoder-decoder transformer models which can be fine-tuned on the target dataset (Lewis et al., 2019; Zhang et al., 2020; Raffel et al., 2020), to transformer variants which are designed for longer input context (Xiao et al., 2021; Phang et al., 2022; Guo et al., 2021) or generative instruction-tuned Large Language Models (LLMs) summarizing documents based on a given prompt (Goyal et al., 2022).

In the news domain, the aforementioned methods have been explored more extensively either in Single-Document Summarization (SDS) tasks or in Multi-Document Summarization (MDS) tasks where the number of input documents is relatively small. On the other hand, in the scientific domain, there are several under-explored areas, especially in the application of generative instruction-tuned LLMs and in the summarization of large collections of documents. As opposed to the news domain, the source documents tend to be longer whereas the writing style is more scientific by nature. Specifically, although generative LLMs show strong summarization capabilities in the news domain, it is not clear if their performance in the scientific domain will be similar. Moreover, most work in the summarization of scientific articles is focused on SDS tasks, while MDS research usually explores datasets that contain a few documents (e.g. Multi-XScience (Lu et al., 2020) with around 4.4 source documents per case). Even BigSurvey (Liu et al., 2023a) with around 76 source documents per case is still far from the documents contained in large collections such as a conference, where we can have even thousands of source documents.

In general, due to both the huge input length, and the fact that a user or researcher has limited time to allocate to consuming new information, a system performing extreme summarization is required to provide the general intuition of the documents' collection. In other words, we can safely assume that the purpose of such a system would be not only to produce a summary of a large collection of papers but also to present it in a digestible way. For example, generating a long text output without any specific structure would not be appealing to a user interested in gathering insights into the given collection.

Based on the above, as the starting point of our work, we decided to first cluster the documents, and then summarize each cluster. In this way, the documents contained in a collection are organized in topics which facilitate the consumption of the information. For

the clustering of a collection of documents into meaningful topics, several algorithms have been proposed (Angelov, 2020; Grootendorst, 2022; Traag et al., 2019), but in this work, we are not aiming to compare the different variants of clustering algorithms. Instead, we decided to use the Leiden algorithm (Traag et al., 2019), because of its graph-based output, which can also be used as a visualization tool in our summaries.

The summarization of clusters of papers reveals various research questions, since to the best of our knowledge, it is a task that has not been explored in the past. Thus, our goal is to compare different ways of producing such summaries. Taking into consideration the rise of decoder-only instruction-tuned generative models, their promising performance in summarization in the news domain (Goyal et al., 2022), and their adaptability to different instructions that can be used to guide the generation of the summary, we decided to use such model as the backbone LLM of our system. In the end, we demonstrate the effectiveness of our system, by applying our proposed configuration to the source documents of a large AI conference, in order to generate an automated blog post that can be used as the summary of the whole conference.

Based on the above, we address the following research questions:

RQ1: How does an instruction-tuned generative LLM compare to fine-tuned LLMs in Single-Document (SDS) and in Multi-Document Summarization (MDS)?

RQ2: How does a two-step approach, where we firstly summarize the source documents and then we generate the multi-document summary of the generated summaries, compare to a one-step approach where the full content of the source documents is given as input to the model?

RQ3: What is the effect of selecting representative documents per cluster on the generation of the summary of a large collection such as the case of a conference?

In the current work, as the backbone instruction-tuned LLM of our system we use ChatGPT, that is the chat-based version of the gpt-3.5-turbo family of OpenAI models, which is optimized for conversations.[1] Additionally, for applying our pipeline to large collections of documents such as AI conferences, we use the Zeta Alpha[2] platform (Fadaee et al., 2020), where we can collect the papers published in any AI conference and also apply the Leiden clustering algorithm on them. By leveraging known open-source summarization datasets and by conducting a user study about preference judgment between different summaries of an AI conference, we are able to draw conclusions about the above questions. Our main contributions are summarized below:

- We propose an end-to-end system that uses a generative instruction-tuned LLM as the backbone in order to produce summaries of large collections of documents. We demonstrate the effectiveness of our system by generating the first fully-automated blog post summarizing a whole conference.
- We show that generative instruction-tuned LLMs can be competitive out-of-the-box solutions for both SDS and MDS tasks in the scientific domain, approximating or even outperforming state-of-the-art finetuned models in popular summarization datasets.
- We find that using a two-step approach in our prompt-based method leads to an

[1] https://platform.openai.com/docs/models/gpt-3-5
[2] https://www.zeta-alpha.com/

improvement in performance compared to a one-step approach where the full content of the source documents is given in a single prompt.

- We demonstrate that prompting our system for summaries based on only 10 representative documents per cluster produces summaries that are preferred by humans compared to those that were generated by prompting with the entire list of documents. Specifically, the last two points indicate the potential weakness of a prompt-based model to summarize a large set of longer and diverse inputs coming from different documents.

2 Related Work

2.1 Abstractive summarization

The undeniable progress in the field of NLP has led to significant improvements in abstractive summarization. Also, while the most common solution was to fine-tune a pretrained language model on a downstream task, the recent advancements in the field indicate that prompting an instruction-tuned Large Language Model (LLM), can yield better results without the need for a fine-tuning step (Brown et al., 2020; Wei et al., 2021). Of course, in the case of summarization, the length of the input documents affects the selection of the model, due to the limit in the input context of the tranformers architecture.

In general, when the input document can fit into a vanilla transformer architecture, which is usually the case for the news articles domain, language models optimized during pre-training for task generation tasks such as summarization can be used on the target dataset with a potential fine-tuning phase. Prominent examples of such models are BART (Lewis et al., 2019), PEGASUS (Zhang et al., 2020), BRIO (Liu et al., 2022), and T5 (Raffel et al., 2020).

However, with the rise of general-purpose instruction-tuned LLMs such as GPT-3 (Brown et al., 2020), FLAN (Brown et al., 2020), Alpaca (Taori et al., 2023) and the Anthropic model (Bai et al., 2022), prompt-based approaches are explored, since they can be applied out of the box and generate fluent and informative summaries without the need for a fine-tuning phase. The latter cases lead to zero-shot outputs that can even be preferred over the outputs of fine-tuned models (Goyal et al., 2022; Zhang et al., 2023b). Furthermore, in order to make the most out of such models, general prompting techniques can be tried, including In-Context Learning (ICL) (Dong et al., 2022), where few-shot examples of the target task are given in the prompt, or Chain-of-Thought (CoT) (Wei et al., 2022), where the LLM is prompted to think step-by-step before generating the final response. Also in the field of summarization, recent works research the iterative refinement of the summary using ChatGPT (Zhang et al., 2023a), or the distillation of a gpt-3.5-turbo model into a versatile and compact model that achieves similar or superior performance over the teacher model (Xu et al., 2023). It is important to note though, that a large challenge of using such models is to mitigate hallucinations that might contradict the source documents. While intrinsic hallucinations, that are based solely on the given documents, are not usually expected in abstractive summarization, extrinsic hallucinations which derive from the internal knowledge of the LLM may occur (Zhang et al., 2019a).

In addition, As the input length is getting larger, using a vanilla transformer requires truncation of the input, which may lead to information loss. This problem is crucial for domains where the input documents are long by nature. In the case of scientific articles summarization, using only specific parts of the whole document (e.g. abstract, introduction) can be a sufficient solution

for a general-purpose summary, since the main idea of the article is usually described well in these specific sections. Selecting informative parts of a document has also been followed as a way to construct summarization datasets for scientific articles: SciTLDR (Cachola et al., 2020), CiteSum (Mao et al., 2022), Multi-XScience (Lu et al., 2020). An alternative method to handle larger context is performing a two-step approach where an extractive summarization step is followed by an abstractive method (Liu and Lapata, 2019; Zhang et al., 2019c). The most promising research in summarizing long documents, however, explores the modification of the attention mechanism of the transformer architecture in order to fit larger context even up to 16K tokens, reporting state-of-the-art results in long documents summarization. Examples of such models and methods can be found in PRIMERA (Xiao et al., 2021), PEGASUS-X (Phang et al., 2022), LongT5 (Guo et al., 2021), HierEncDec (Shen et al., 2023), and RAMMER (Li et al., 2023). Similar increases in input length are explored by closed-source generative LLMs such as GPT-4 by OpenAI (Koubaa, 2023) which scales to 8K tokens and the Cohere Summarize API[3] which supports up to 50K characters.

2.2 Summarization of large collections

The aforementioned approaches can scale the input length, but as we move to even longer inputs as in the use cases of book summarization or Multi-Document Summarization (MDS) of large collections of documents, the same issues seem to appear. This is also the case for the BigSurvey (Liu et al., 2023a) dataset, where around 76 abstracts are used as input to the summarization model, which requires an unavoidable truncation step before passing the input to a long-context transformer architecture. Solving these cases is of high importance since they are much closer to real-world applications where users seek to get insights from large amounts of information. Recent approaches aim to scale the input context of transformers to significantly larger or even unlimited lengths (Bertsch et al., 2023; Martins et al., 2021). This direction is also explored with generative LLMs such as Claude by Anthopic (Bai et al., 2022) which claims to support 100K tokens. Besides increasing the input length, other approaches focus on creating a divide-and-conquer summarization system where parts of a long input are summarized, and their summaries are combined and summarized further in a bottom-up fashion till the full content is covered (Wu et al., 2021).

Even with the aforementioned methods, the summarization of a large collection of documents remains a challenging task since it contains a huge amount of diverse information which is difficult to summarize and present in a digestible way. Previous approaches to provide insights into such collections focus more on producing topic words or keywords that describe a large corpus (Thielmann et al., 2023; Zhang et al., 2022). Similar methods explore the generation of a set of questions that can be used to summarize a large amount of textual data (Surita et al., 2020). These ideas are quite helpful, but an automated summary is expected to serve better the purpose of digesting a large number of papers due to its fluency and user-friendliness.

Furthermore, as already mentioned, when dealing with substantial amounts of information, particularly in scientific contexts like conference papers, it becomes crucial to divide the content into digestible sections in order to enhance user comprehension. For that reason, clustering the documents into meaningful topics and then summarizing each topic seems to be a worth-exploring idea. In the context of multi-document summarization, this approach was also examined by using an encoder transformer architecture to cluster the documents

[3] https://txt.cohere.com/summarize-beta

and then generating the summaries using a decoder (Trabelsi and Uzunalioglu, 2023). Our work though differs not only in the clustering method, but mostly in enabling the usage of generative LLMs to produce summaries according to the requirements of each specific usecase.

Regarding the clustering of the documents and their assignment to specific topics, methods designed for topic modeling seem promising (Angelov, 2020; Grootendorst, 2022). However, there are alternative approaches that focus on clustering the input documents based on a given graph structure which uses the edges between documents to communicate their similarity. The latter method and specifically the Leiden algorithm (Traag et al., 2019) will be used in our system due to its visualizable graph output, where each node is a document and each edge represents the similarity between two documents. In general, different networks can be used as input including citation-based networks where documents are connected to their cited documents (Van Eck and Waltman, 2017), or networks which are based on the textual similarity between documents. In the latter case, which is adopted by the Zeta Alpha platform [4], the edges between papers are created if the similarity score exceeds a predefined threshold. This similarity score can be calculated with various approaches including cosine similarity between the document embeddings or the 'More Like This' query functionality of Elastic Search[5]. The latter is used in our case where for each document, the terms with the highest TF-IDF score are used to formulate a disjunctive query, which will be executed to return similar documents based on those terms (Kuc and Rogozinski, 2013).

2.3 Evaluation of summaries

Concerning the evaluation of our system, it is clear that there are several challenges due to the volume of the input documents that characterizes our task. At first, the datasets used in the scientific domain focus either in SDS tasks (Cachola et al., 2020; Mao et al., 2022), or in MDS tasks with a small number of input documents (Lu et al., 2020). Even the datasets with a larger number of source documents (Liu et al., 2023a) do not meet the needs of our use case which focuses on summarizing large collections of thousands of papers. Based on the above, we need to evaluate the components of our pipeline separately by using the aforementioned datasets, before evaluating the end-to-end system with a user survey.

Another important challenge arises from the lack of a generally accepted metric that can be used for summarization. To begin with, there is a variety of metrics that are used to evaluate a summary based on a set of given reference summaries (i.e. gold summaries). ROUGE score (Lin, 2004), which seems to dominate the field, is a recall-oriented metric, while BLEU (Graham, 2015) which is another known overlap-based metric focuses on precision and seems to correlate stronger overall with human judgment (Graham, 2015). Using both metrics could provide a sufficient overview at first glance, but their inability to evaluate the meaning of a summary led to the exploration of metrics, which are based on semantic similarity, with BERTScore being the most popular (Zhang et al., 2019b).

Since we want to compare the use of instruction-tuned LLMs to existing fine-tuned models for known SDS and MDS tasks, we ended up using the ROUGE scores as those are consistently reported by all summarization models, but we decided to also report BLEU and BERTScore metrics in order to provide a more extensive evaluation. Based on these comparisons, we can

[4] https://github.com/zetaalphavector/VOSviewer-Online

[5] https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html

then determine which methods are more promising and conduct a user survey to observe the preference judgment of users. Even though it is impossible for a human evaluator to fully comprehend all the input information given to our system, we drew inspiration from existing evaluation protocols (Fabbri et al., 2021; Kryściński et al., 2019), and we adapted them for our use case.

It is important to note also, that recent studies explore the use of instruction-tuned LLMs for the evaluation of summaries. These studies include methods that prompt LLMs (Brown et al., 2020; Koubaa, 2023) to evaluate a summary based on specific criteria or even provide a score that can be used as a rating of the summary (Fu et al., 2023; Liu et al., 2023b). Other approaches are also examining the factual consistency of a generated summary, in order to find potential hallucinations of the models (Chen et al., 2023). Although these methods point to a promising direction, they cannot be applied directly to our case due to the large context that needs to be allocated for the input documents. Of course, we can still use these methods to test components of the pipeline in isolation, but we decided to explore this as future work.

3 Methods

The goal of our work is to provide an end-to-end system that summarizes a potentially large collection of documents by using a instruction-tuned LLM as backbone. The overview of the system is presented in Figure 1, where we can see all the components of our pipeline, which will be explained in detail in the upcoming sections. Those components include the clustering of the documents (see Section 3.1), followed by the summarization module. The latter, which is the main focus of our research, contains the selection of representative documents per cluster (see Section 3.2) and then a two-step summarization approach where the single-document summaries of the papers are generated in the SDS stage (see Section 3.3) and are given as input in the MDS stage (see Section 3.4) that produces the final cluster summaries. Finally, to apply our system to a real-world scenario, such as the summarization of a conference in AI, we made one further step in order to generate a fully automated blog post using the blog post creator component of the end-to-end system. Details about the application of our pipeline on conference summarization are presented in Section 3.5.

It is important to note that since our main focus is the summarization module, we decided to keep the clustering method and the blog post creator fixed and leave their further exploration for future work.

3.1 Clustering documents

The first step of our end-to-end system is to cluster the text documents using a graph-based algorithm that produces as output a network, where each node is a document assigned to a cluster, and each edge represents the similarity between two documents. This is an important prerequisite for our proposed summarization pipeline, because the structure of the graph and the similarities between documents will be used by the detector of representative documents. In case we use a clustering algorithm that does not produce such graph as output, it would also be necessary to adapt the selection of representative documents accordingly.

Based on the above, we decided to use the Leiden graph-based clustering algorithm (Traag et al., 2019), and especially the version that is implemented by VosViewer (Van Eck and Waltman, 2010). Firstly, as already mentioned in Section 2.2, we find the similarity score between documents using the 'More Like This' query functionality of Elastic Search. Then,

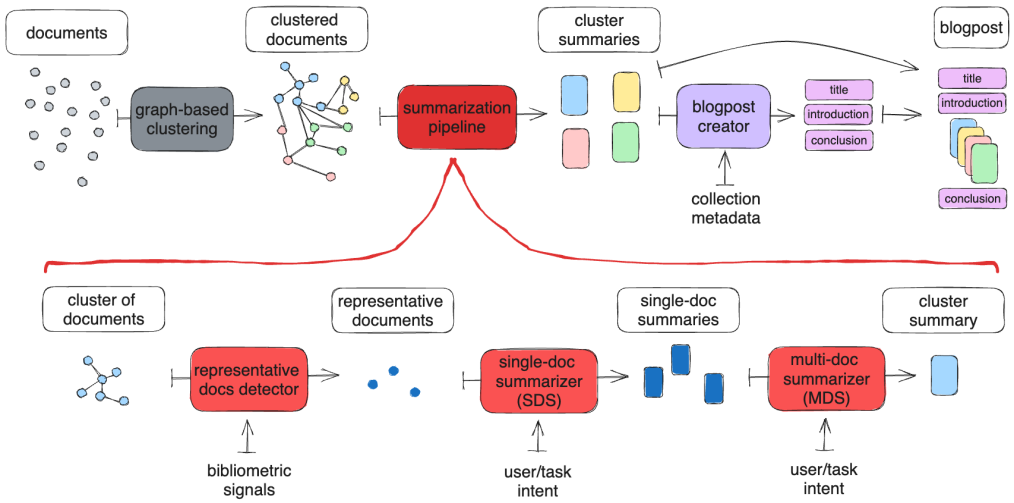

Figure 1: Pipeline overview of automatic blog post generation for a large collection of documents, such as conference proceedings. The clustering component is fixed, while for the blog post creator component, a simple prompt-based approach was used to generate the blog post title, introduction and conclusion. The focus of our work is given to the summarization module, which is responsible for taking as input a set of clusters of documents and producing a summary for each cluster based on the intent of the user or the task at hand.

we create the network of the collection at hand, where the similarity scores are used as edges between the documents after applying max-normalization. The network is then passed as input to the VosViewer implementation of the Leiden algorithm which returns the same network back, enhanced with the cluster attributed to each document. In order to optimize a clustering quality function known as modularity, the algorithm follows an iterative process where the nodes are moved from one community to another in order to create partitions, which are also refined during the process, till no further improvement can be made (Traag et al., 2019).

The generated graph structure is now composed of clustered nodes, which represent the documents of the collection, and of the aforementioned edges, which are still assigned with the similarity score of the documents they connect. This structure is then passed to the first component of the summarization pipeline, which is responsible to find the most representative documents per cluster.

3.2 Representative documents detector

In order to find representative documents per cluster, three variants of this component are explored, where the first performs no selection of representative documents, the second uses the graph structure produced by the clustering algorithm to select a set of documents that covers as much as possible the whole cluster, and the third incorporates bibliometric signals to the second approach.

3.2.1 No selection of representative documents

This variant basically suggests omitting this component and attempting to summarize all the documents of a cluster at once, as long as the tokens limit of the LLM is not exceeded. With the two-step summarization approach that is described in the Sections 3.3 and 3.4, we can take even a hundred documents in the input of LLMs with 4K tokens limit. Thus, based on

the ability of our summarization method to scale up to this number, we decided to inspect the effect of not detecting representative documents per cluster.

3.2.2 Selection of representative documents that maximize the coverage of the cluster

Our idea to select representative documents derives from the intuition that it will be hard for an LLM to process and summarize in a robust way a large amount of information coming from different documents. As a result, this variant aims to select a set of documents that represent as much as possible the whole cluster.

The number of selected documents can vary and depends on the desired length of the output summary. We assume that the length of a summary limits the amount of information and consequently the number of references that can be included in the summary. Also, a desired feature of our summarization approach when using instruction-tuned LLMs is to allow the end user to decide the length of the summary in terms of the number of words ($target\_length$). Based on the $target\_length$ given by the user of the blogpost, we decided to select $target\_length/10$ documents as representatives of each cluster, since we believe it is an intuitive amount of information to be presented in a summary of such length, taking into consideration that 10 words is a realistic length for a one-sentence summary about one paper. Of course, delving into the impact of this tenfold factor would be valuable, but that task was left for future research.

Moreover, although the representativeness of a document can be interpreted in various ways, we believe that a promising starting point would be to create a portfolio of documents that cover the underlying network of the cluster as much as possible. This idea is firstly inspired by the Dominating Set concept in graph theory, which is a set of vertices such that any vertex of the graph is either in the set or has a neighbor in the set (Cormen et al., 2022). Specifically, if for each cluster we select a set of documents that form a minimal dominating set, that means that if we take all the neighbors of these documents, we will end up covering the whole cluster. Taking into consideration that the edges connect papers that are similar to each other, we could safely assume that the selected set is representative of the cluster.

However, each edge has a specific strength value based on the calculated similarity score, which means that some connections are weaker than others. As a result, we believe that it would be risky to assume that all the neighbors of a selected node are fully represented. For instance, in the graph presented in Figure 2a, node A belongs to the dominating set, even though its connections are weak. Instead, we would prefer node B to be selected, since it has much stronger similarity with its neighbors, although it is connected to one less node compared to A.

Due to the aforementioned observation, we need to adjust our approach accordingly. A different direction to solve this problem would be to assign a similarity strength to each node, which is based on the similarity scores of its neighbors. This value would indicate how well a node represents its neighbors. A logical next step of this approach would be to select the top k nodes with the highest similarity strength. However, in the case of graphs where we have a strongly connected subnetwork in the cluster (see Figure 2b), this method would yield inferior results, since the subnetwork would be over-represented, leaving no space available for the remaining nodes of the cluster.

By combining the above two approaches, we ended up with the following iterative steps in order to mitigate these downsides and create a portfolio of documents per cluster:

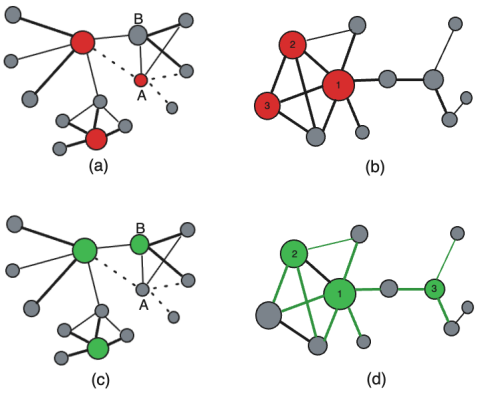

Figure 2: Examples of selecting $k = 3$ representative documents in a cluster, where: (a) uses a minimal dominating set, (b) selects the nodes with the highest similarity strength, and (c), (d) use our suggested algorithm where in each iteration we calculate the similarity strength of each node, then select the node with the highest strength and finally remove this node along with its edges. The size of the nodes indicate their initial similarity strength, whereas the thickness of the edges indicate the strength of the similarity score between two nodes.

- Firstly, we assign a *SimilarityStrength* value to each node which is equal to the sum of the similarity scores of its edges. We do not normalize this strength value, because we can safely assume that the more edges a node has, the more likely it is to represent a larger portion of the cluster.
- Secondly, we select the node with the highest *SimilarityStrength* and we put it into our initially empty portfolio of documents.
- Thirdly, we remove this node from the graph, along with its edges. The removal of the edges is the key step of our algorithm since it will lead to a decrease in the similarity strengths of the neighbors of the selected node. This is a desired effect, as these nodes are already likely to be represented by the selected node.

By iteratively applying the aforementioned steps for k iterations, where k is the desired number of representative documents, we end up with the portfolio of documents per cluster which seems to align more with our intent as observed in Figures 2c, 2d, where the previous issues are mitigated.

3.2.3 Incorporation of bibliometric signals

Although the aforementioned idea seems to be a good starting point for selecting representative documents, it takes into account only the graph structure and the similarities between documents in order to cover the collection topics as widely as possible. However, when a user reads the summary of a large collection, such as a conference, they would expect not only to be informed about all the diverse topics of the collection, but also about the papers that seem more prominent and impactful. Consequently, we have made the decision to integrate bibliometric signals into our approach, aiming to capture the trending and attention-grabbing papers within the collection.

Incorporating such signals though is a challenging task, where extensive research should be done to further understand which metrics seem to capture more accurately the aforementioned trend, especially when applied to recently published papers. Related work in this direction includes citation count prediction (Yan et al., 2011; Li et al., 2019), the journal impact factor

of a conference (Garfield, 2006), or altmetrics (Warren et al., 2017; Priem et al., 2011), such as online interactions and social media attention. The latter seems to be more interesting in the case of a recent conference, where the published papers are new.

In our case, we did not allocate our research focus to this area, but instead, we used the metrics monitored by Zeta Alpha about the citations, the h-index of the authors of a paper, and finally the trend of the papers on Twitter. Specifically, the latter is measured by retrieving from the Twitter API the number of tweets mentioning the arXiv URL of the paper. Combining the aforementioned metrics, we designed a formula in order to account for the recency of a publication and a potential time difference between the release dates of different papers. At first, we normalized the values of each metric by dividing with the maximum value of the collection. Then, we separated the metrics into two types. The first type refers to metrics that can be used as an early indicator of the paper's impact, either by capturing the trend or the influence of the authors in the research community. Those include firstly the $Popularity$ of the paper on Twitter, which is measured as the normalized number of tweets, and secondly the $Influence$, which is the normalized average h-index of the authors, both calculated using max-normalization. The second type of metrics refers to those which may need a significant amount of time, as opposed to the first type, in order to capture the impact of a paper. $Citations$ is such an example, since for newly published papers, we need to wait for several months or years in some cases, before this metric stabilizes and aligns with the impact of a paper.

Based on the above, our intuition is that for a short period after the publication of a paper, the first type of metrics can be useful, but as more time passes by, it is preferable to switch to the second type of metrics which seems more reliable in the long-term. For that reason, we decided to find the average value over popularity and influence, and then interpolate them exponentially, based on the age of a paper (i.e. the time passed from its publication) in order to focus more on the second type of metric as time passes. Thus, we ended up with the following formula:

$$\text{Impact} = \alpha \times Avg(\text{Popularity}, \text{Influence}) + (1 - \alpha) \times \text{Citations} \tag{1}$$

with $\alpha = e^{(textExpFactor \times \text{PaperAge})}$, where $PaperAge$ is the number of days passed from the publication of the paper and $ExpFactor$ defines the steepness of the exponential interpolation. In our case, we decided to start favoring more the 'Citations' after one year from publication and also make it dominate the $Impact$ score after two years. Thus, we empirically estimated that an $ExpFactor = 0.0025$ results in a $Citations$ factor equal to 0.6 for the first year and 0.84 for the second year, which is aligned with our intuition. Of course, this formula is arbitrary and takes into account the metrics we have at hand at the moment. In the future, it would be worth researching and collecting more metrics and formulas proposed for the measurement of the impact of a paper, especially if it is newly published.

Last but not least, to incorporate the impact score into the metric used to select the most representative papers, we decided to interpolate it linearly with the similarity strength:

$$\text{Representativeness} = \beta \times \text{SimilarityStrength} + (1 - \beta) \times \text{Impact} \tag{2}$$

where $\beta \in [0, 1]$ is a factor that indicates how much we should rely on the *SimilarityStrength* and consequently on covering the largest part of the cluster's graph. For $\beta = 1$, we do not take into account the *Impact* at all, making the selection of representatives equivalent to the

method described in Section 3.2.2. For $\beta = 0$, we focus only on the estimated $Impact$ of each paper and we use no information from the graph structure. The latter is not suggested due to the challenges in measuring *Impact* and the small number of resources we have available for estimating this metric, as already mentioned.

3.3 Single-Document Summarization (SDS)

Although our goal to summarize a large collection of documents is an MDS task by nature, our proposed solution is based on a two-step approach, where firstly a summary for each paper is produced and then those summaries are aggregated in the MDS component. As a result, in this section, we are going to explore different configurations of the SDS stage (see Figure 1), including: a pass-through setting that does not summarize the input documents, the usage of pre-trained models fine-tuned on the target dataset, and the out-of-the-box usage of instruction-tuned LLMs.

3.3.1 No SDS step

Since we are dealing with an MDS task, it is reasonable to try summarizing the full content of the source documents directly at the MDS component. Thus, a vanilla approach is to omit the SDS component of the pipeline, and pass through the documents directly to the MDS component.

Methods suggested by related work in multi-document summarization usually follow this direction by passing all the documents to a transformer architecture separated by a document separator token. Specifically, different approaches so far focus on training alternative transformer architectures by aligning the training objective with the objectives of a summarization task and by suggesting architectures with sparse attention in order to increase the context length and fit more documents in the input (Xiao et al., 2021; See et al., 2017; Guo et al., 2021). Such models are described in detail in Section 3.4, where the MDS component is presented. As a result, for these cases it is not expected to use an SDS component, since those models are already trained with the full content of the source documents.

3.3.2 Fine-tuned models

In order to use our two-step summarization approach, we need an SDS model to summarize each paper given its abstract. For that reason, we can use a pre-trained LLM, designed for summarization, which can also be fine-tuned on the target dataset. In our case, the documents belong to the scientific domain. Moreover, since our goal is to produce compressed summaries, where the output length is much smaller compared to the size of the input documents, we are interested in models that can perform well in extreme summarization of scientific articles. Based on the recent research in SDS and on the performance of fine-tuned models in tasks related to extreme summarization of scientific articles such as SciTLDR (Cachola et al., 2020), we are going to explore the following models which have already been fine-tuned on our downstream task:

- BART (Lewis et al., 2019): a sequence-to-sequence encoder-decoder model, which is pretrained in order to adapt for downstream text generation and understanding tasks like summarization.
- PEGASUS (Zhang et al., 2020): a popular family of summarization models, based on large encoder-decoder transformer models, which are pretrained with a self-supervised objective tailored for abstractive text summarization. Specifically, important sentences

are masked from the input document and are then generated as one output sequence from the remaining sentences.

- T5 (Raffel et al., 2020): a powerful encoder-decoder model pretrained on a mixture of unsupervised and supervised tasks, which are distinguished based on a text prefix followed by the input corresponding to each task. For summarization, we need to prefix the source document with the prompt 'summarize: '.

3.3.3 Instruction-tuned models

Our main inspiration for the two-step summarization method is to use an instruction-tuned LLM as backbone, as it will be easier to adapt it for a downstream task by modifying the instructions of the prompt. For that reason, besides the document that we aim to summarize, we are also giving the user/task intent as input to the prompt of the model (see Figure 1) in order to adapt the summary accordingly.

Since the rise of generative LLMs, several prompting techniques (Dong et al., 2022; Wei et al., 2022) have emerged, revealing the potential of powerful instruction-tuned LLMs. In our case, we decided to explore the In-Context Learning (ICL) approach which can be applied by incorporating document-summary example pairs in the prompt. Additionally, when using LLMs that can operate as chat bots, there may be an option to set up the chat bot in a specific way by giving it a persona before starting the actual conversation which will contain the task. This is also the suggested approach by Open AI for their chat-based models gpt-3.5-turbo and GPT-4, where with the system message, we can introduce high-level instructions beforehand.[6]

Based on the above, we explore different prompt variants. Firstly, inspired by the summarization of news articles using instruction-tuned LLMs (Goyal et al., 2022), we decided to use a simple prompt as a vanilla approach in order to observe its effectiveness in the scientific domain (see Table 1). This prompt does not take into account details about the user intent which might be related to the style of the summary. Secondly, taking into account the aforementioned ideas and also the flexibility of such models in following the user's instructions, we created a prompt template (see Table 1), which contains also a setup message for the chat bot, which of course can be adapted to the SDS task at hand. Moreover, the potential usage of ICL to guide the generated summaries is allowed by setting up the conversation with pairs of user-bot messages which correspond to the suggested user message template and to the gold summary respectively. It is important to point out also that the prompt includes instructions about the style and the length of the summary, the target audience, and general guidelines about the tone of the writing. These instructions can be amended based on the task at hand as well.

In this work, we use gpt-3.5-turbo in order to compare these different variants, including the vanilla prompt and our suggested prompt with and without ICL. Specifically, based on the intent of the summary, the instructions of the prompt may be slightly adjusted along with the few-shot examples. When dealing with a known SDS dataset, we can select the examples from its training fold. However, when the task at hand is not related to a specific dataset, we pick our examples by iteratively prompting GPT-4 to generate and adjust the summary of a sample document that is similar to our target collection. Finally, by running the model on different tasks, we can acquire a sense of its adaptability to the instructions.

[6] https://platform.openai.com/docs/guides/chat

 # Vanilla prompt: {{*document*}} Summarize the above document in 1 sentence.
 # Our suggested SDS prompt: ***### system message***: You are the most famous research journalist in writing summaries of scientific articles. Your summaries are concise, informative, and of high quality. As an expert in grammar and vocabulary, you possess the ability to adapt your writing style according to provided instructions. ***### user message***: Write a short and concise sentence summarizing the provided document in {{*target_length*}} words.The summary should be informative for {{*target_group*}} {{*document*}}

Table 1: Prompt Templates for SDS instruction-tuned LLM, where '{{document}}' has the content of the source document, '{{target_length}}' the target length of the summary in words count, and '{{target_group}}' the description of the intent of the reader, which is set to 'a reader who is an experienced researcher in this field'.

3.4 Multi-Document Summarization (MDS)

The final step of the summarization module focuses on producing a short summary out of a set of documents. The content of these documents depends on the SDS variant of the previous step. For instance, if no SDS component is used, the MDS stage will receive as input the full content of the documents. If an SDS model is used, however, the summaries of the input documents will be given to the MDS component. The different models that can be used to summarize multiple documents are divided again in two categories, where the first one includes finetuned LLMs and the second one instruction-tuned LLMs as the ones used in SDS step.

3.4.1 Fine-tuned models for MDS

Similarly to the SDS component, a pretrained LLM which can be fine-tuned on a downstream multi-document summarization task, is an approach that has been tried frequently. As already mentioned, in the case of multiple documents, increasing the allowed context length is crucial in order to fit more documents and consequently larger amount of information in the input of a transformer architecture.

Although most work in this field focuses on the news domain, there is also research applied to the summarization of scientific articles, especially with MDS datasets in this domain (Lu et al., 2020; Liu et al., 2023a). In our case, since our proposed summarization approach is based on summarizing a specific number of representative papers, we decided to use the following fine-tuned models, which have already shown significant performance in the Multi-XScience dataset, which contains a small number of documents as well (Lu et al., 2020; Xiao et al., 2021).

- PRIMERA (Xiao et al., 2021): a pre-trained model for MDS, which is based on an objective that guides the model to aggregate information across documents and uses encoder-decoder transformers to facilitate the processing of concatenated input documents.
- Pointer-Generator (See et al., 2017): an architecture that enhances the standard sequence-to-sequence transformers, by using a pointer-generator network that firstly copies words from the source to reproduce information and secondly amends the summary via the

generator.

As already noted, these models are already designed for handling the full content of the documents in the input. For that reason, when using a fine-tuned model as our MDS component, the SDS step is omitted. As a result, by using the aforementioned models fine-tuned on our target task, we will be able to compare them against our two-step approach that uses an instruction-tuned LLM as the backbone.

3.4.2 Instruction-tuned models

As already explained in Section 3.3.3, a powerful instruction-tuned LLM is a worth trying alternative to fine-tuned models due to its ability to adapt to prompts and to be used out-of-the-box without any fine-tuning on the target task.

As with the fine-tuned models for MDS, instruction-tuned models can also be applied directly to the full content of the source documents. It is clear though, that this limits the number of documents that can be given in the prompt due to the context limit of the transformer models. Nevertheless, based on our intuition that including a large amount of information from different documents will make it harder for the LLM to produce a summary in a robust way, our suggested solution uses as the MDS component a prompt that contains only the summaries of the source documents. This approach can scale to more documents and also enables the LLM to focus on the key points of each document. Finally, due to the significant decrease in the prompt length, the latter method can allow examples in the prompt in order to guide the summary style and intent through ICL.

Based on the above, it is useful to explore the two aforementioned approaches and observe if using the two-step approach is more beneficial. In our experimentation, we are going to use gpt-3.5-turbo as the backbone LLM model, exactly as in the SDS component. The prompt templates used for our MDS tasks include a system message, followed by a user message that contains the task definition, instructions about the output format and the writing style, and a list of instructions to handle common issues, observed when experimenting with the model.

For different use cases, we are modifying the prompt accordingly in order to adjust it for the given task. For instance, for the cluster summaries provided by our system, our goal is to summarize the input documents while providing references when needed. Then, these summaries will be used as building blocks of our automated blog post. On the other hand, due to the lack of available MDS datasets in the scientific domain, during the systematic evaluation of our system, we are going to deal with a different task, proposed by the Multi-XScience dataset, where the related work section of a paper is requested based on its abstract and the articles it references (Lu et al., 2020). The prompts used for these two tasks are presented in Table 2.

Refinement prompts At this stage, due to the increased complexity of the task compared to the SDS tasks, we are also interested in exploring different ways to help the LLM follow the instructions. This intent is mostly focused on the format of the output and the way the information is presented in the summary. Besides the aforementioned ICL method, where we will give as input in the prompt an exemplar with the expected format, we are also considering refining the output summary with follow-up prompts. Specifically, after checking the quality of the output based on implemented functions that evaluate various criteria, we can continue the chat-based interaction with gpt-3.5-turbo in order to explicitly ask for specific improvements. This idea is similar to the recent work of SummIt, where the authors

 # Multi-Doc Summary with references:
system message:
You are the most famous research journalist in writing summaries of scientific articles. Your summaries are concise, informative, and of high quality. As an expert in writing, you possess the ability to adapt your summaries according to the provided instructions.
user message:
Write a short and concise paragraph of at most {{*target_length*}} words that summarizes the given documents. The summary should be informative and appealing to {{*target_group*}}.
Refer to the documents using 'd' plus their index in square brackets and cite them wherever needed. All documents should be cited. Ensure completely that each citation is supported by the information provided in documents.
Use only information from the given documents.
Do not use generic sentences that do not refer to any document.
Do not mention how many documents are given.
Do not mention anything related to the order and the position of the documents in the list.
Do not use the citation as the subject of any sentence.

Documents:
[d1]: {{*document_1*}}
[d2]: {{*document_2*}}
...

 # Multi-XScience:
system message:
You are a researcher who is an expert in writing the related work section of a given scientific article. Your writing is concise, informative, and of high quality. As an expert in writing, you possess the ability to adapt your summaries according to the provided instructions.
user message:
You are the author of a scientific article. You have already written the abstract of the article, and you are currently writing the related work section of the article. You want to create a paragraph of at most {{*target_length*}} words, which will be used without modification as a paragraph in the related work section that refers to the referenced documents, either to base on their ideas or to challenge them.
Be fluent. Avoid repetitive information. Refer to the referenced documents of the list using their $id in the given format: "@cite_$id". All documents should be cited. You are encouraged to cite more than one document in one place if you are sure that the citation is supported by their content.

Scientific Article abstract:
{{*main_document_abstract*}}
Referenced documents:
@cite_i: {{*document_i*}}
@cite_j: {{*document_j*}}
...

Table 2: Prompt Templates for MDS instruction-tuned LLM, where {{document}} placeholders have the content of the documents, '{{target_length}}' refers to the target length of the summary in words count, and '{{target_group}}' refers to the target audience of the summary and is set to 'a reader who is an experienced researcher in this field'.

make a further step to automate the refinement of the output by using an LLM to evaluate it and propose modifications (Zhang et al., 2023a).

To examine these ideas, we focused on the task of creating cluster summaries of 100 words out of the single-document summaries of 10 papers. This task is highly important, since such summaries will eventually be used as a building block for our final conference blog post. Based on this task, the quality validations and refinement prompts that we used are presented below:

- *Too long summary*: Since chat-based LLMs like gpt-3.5-turbo tend to produce longer text than expected, it is important to calculate the length of the output. We implement a function that counts the number of words in the generated text and if it is larger than a specific threshold, we prompt the model with a follow-up message to shorten the summary. Thus, if the cluster summary was more than 130 words, we interact with gpt-3.5-turbo with the following refinement prompt:
  'Shorten the summary to fit in at most 100 words, while keeping it informative and fluent. Keep in mind to include citations to all documents.'

- *Missing citations*: While exploring the capabilities of gpt-3.5-turbo in our task, one instruction that was ignored in several cases was when we asked the model to include accurate references to the papers (i.e. citations). Based on the expected output format of the references, where the index of the document is used in square brackets, we create a function to count the number of citations included in the generated summary, and if this number is below a specific threshold percentage, then we prompt the model to add citations to all the given papers. For example, if the cited documents are fewer than the 80% of the input documents, then the the following refinement instruction is given to the conversation with the model:
  'Not all documents are cited in the summary. You should cite all documents while keeping the length of the summary at most at 100 words.'

- *No aggregated sentences*: Another issue, observed in the generated summaries, was that in several cases the output contained one sentence for each input document without any aggregation at all, resulting in a less user-friendly output with insufficient connections between sentences. To avoid this problem, we create a function to check the content of each sentence, and if all sentences contain only one citation each, then we decide to prompt the model to merge more sentences together in order to improve readability. The follow-up message used for that purpose is presented below: *'Merge together several sentences in order to make the summary more readable and fluent. Keep the length of the summary at 100 words at most.'*

The aforementioned validation checks and refinements prompts are used in every generated summary till no validation check is triggered or till we reach three interactions with the LLM, meaning that we only allow three refinement cycles. This was done because the refinement process increases the response time since we have to wait for the model's output before validating the quality of the summary. Thus, each round of refinements requires the generation of the full summary which can take several seconds.

Taking into consideration the described approach, we generated 20 cluster summaries using the following variants for the MDS component, while keeping the SDS stage fixed:

- Prompt: The model is prompted only with the instructions of the task without ICL and

refinement follow-up prompts.

- Prompt + One example: The task prompt is used along with one exemplar in order to guide the summary output using ICL. The exemplar was created with the help of GPT-4 where the same task was given, and the output of the model was manually refined with follow-up messages till the quality of the summary met the requirements of our prompt instructions.
- Prompt + Refinement: The task prompt is used, and after checking the model output, we have the chance to refine further the summary with the described validation checks and refinement prompts. Three retries are allowed.
- Prompt + One example + Refinement: A combination of the last two variants is used, where we use both, a one-stot setting in the initial prompt and the refinements messages till we reach three retries.

These variants are analyzed in terms of the following criteria which were designed in order to observe how much the summary respects the input instructions:

- *avg_number_of_words*: The average number of words in the summary. This criterion will evaluate the ability of the LLM to adapt to the instruction about the target length of the summary.
- *avg_number_of_sentences*: The average number of sentences. This aspect will be used as an indicator of the complexity of the sentences and the ability of the model to combine the information in single sentences.
- *percentage_of_docs_cited*: The percentage of the input documents that are cited in the summary. This measurement will show us how much the model follows the instruction about citing all the documents of the input.

The outcome of this comparison is presented in Table 3, where we can see that refinement prompts and one-shot setting both improve significantly the ability of the model to follow the instructions, even when just one of the two is applied. Based on these results, and also taking into consideration that refinement prompts lead to a significant increase in response time, we decided to use the one-shot prompt type in our blog generation, where the prompt will be enhanced with one example. When there is no gold summary available, we create this example by interacting with GPT-4 in OpenAI playground.[7] Specifically, we give the same task with a predefined set of documents and we ask GPT-4 to generate the summary. Then, we manually inspect the output summary and iteratively refine it by prompting the model to adapt it, until it meets the needs we have described in the instructions.

3.5 Conference summarization

While the components described in the previous sections are sufficient to cluster a large collection of documents and also provide one summary for each cluster, we believe that it would be useful to aggregate these summaries into a single document that will be used as the summary for the whole collection. One potential application of our pipeline would be to generate a blog post that summarizes the whole collection. Such an example is the summarization of a large conference in AI which is a fast-paced field and the need to stay up-to-date with the latest advancements is emergent.

[7] https://platform.openai.com/playground

Method	# of words	# of sentences	% of cited docs
Prompt	157.3	9.25	89.0%
+Refinement	126.4	8.80	92.2%
+One Exemplar	119.3	5.70	96.5%
+Refinement+One Exemplar	112.7	5.50	97.2%

Table 3: Heuristic evaluation of four different variants of the MDS prompt for generating a summary of the input documents. The results were gathered after generating 20 cluster summaries of various conference papers for each variant. In each variant, we instructed GPT-3.5 to write a summary of 100 words including references to all 10 input documents.

For that reason, in this section, we are going to present the missing components that will enable us to produce a fully automated blog post. Then we are going to examine two variants of our system, one baseline, and one suggested version.

3.5.1 Blog generation

As shown in Figure 1, an additional 'blog post creator' component is needed in order to combine all cluster summaries together and along with specific metadata about the collection, generate a full blog post.

We believe that a blog post format that could be followed for a large collection, needs to include a title, an introduction, sections about the main topics of the collection, and a conclusion. Also, including figures to avoid a monotonous text structure would also benefit the readability of the blog post.

Based on the pipeline described so far, we have the clusters of the collection, which could be used as the sections of our blog post. In addition, by using the MDS prompt for summarizing a set of documents including references, we have a summary for each section. We can assume though that as observed in human-written blogs, providing a short introductory paragraph for each section, before delving into the details of the referenced papers would make the blog post more appealing. Thus, we need to run our MDS component for a second time, by adapting the prompt to generate a short introductory paragraph as shown in Table 4. It is important to note that these two executions of the MDS stage can be done in parallel after the SDS stage, leading respectively to the generation of an introductory paragraph and of a more detailed cluster summary with references.

Once we have the introductory paragraph and the summary for each cluster, we can concatenate them and create the content of the blog post sections. Also between these two paragraphs, we can add a visualization of the cluster that was generated by VosViewer during clustering (see Section 3.1).

At this point, the titles of each section, and the blog post surroundings (i.e. the general blog post title, the introduction and the conclusion) are missing. Although this is not the main research focus of this work, we decided to create another set of prompts in order to generate these missing parts and fully-automate the generation of the blog post. Further exploration of these prompting is left for future work in order to improve the generated blog post.

Specifically, we designed two more prompts that take as input the introductory paragraphs of each cluster, and generate respectively the section titles and the blog post surroundings. The latter also takes as input the metadata of the collection, which in terms of a conference refers to its name, location, website and date.

 # Multi-Doc Summary used as introductory paragraph:
system message:
You are the most famous research journalist in writing summaries of scientific articles. Your summaries are concise, informative, and of high quality. As an expert in writing, you possess the ability to adapt your summaries according to the provided instructions.
user message:
Write a short and concise paragraph of at most {{*target_length*}} words that can be used as an informative introductory paragraph for the given documents. The paragraph should be informative and appealing to {{*target_group*}}.
Focus on what the given documents have in common and reveal a potential underlying trend in the field.
Do not reference or cite any documents. Rely on the information provided in the documents, but do not use explicitly any text from them.
Do not mention how many documents are given. Do not mention anything related to the order and the position of the documents in the list.

Documents:
[d1]: {{*document_1*}}
[d2]: {{*document_2*}}
..

Table 4: Prompt Template for generating a short paragraph for a given set of documents that will be used as an introductory paragraph for a blog post section about these documents. The {{target_length}} was set to 30 for our pipeline, and the {{target_group}} to 'a reader who is an experienced researcher in this field'.

The reason of prompting the model to generate all the cluster titles at once by giving all the introductory paragraphs as input was based on our intent to avoid generating the same title for different clusters. Specifically, some of the clusters might be semantically close to each other, thus prompting for a title for each of them separately increases the risk to produce the same generic title for both clusters. By giving both of the clusters in the prompt's input though, and instructing the model to produce different titles, it is more likely to get more specific titles that both represent the cluster and differentiate it from the other clusters.

Furthermore, regarding the generation of the blog post surroundings (title, introduction, conclusion), we decided to create a single prompt as well, in order to help the model generate more consistent content for these parts, since due to its auto-regressive nature, it will have access to all the already generated parts.

The prompt templates for the generation of the cluster titles and the blog post surroundings can be found in Table 5.

3.5.2 Proposed end-to-end setting

Based on the components explained in detail in the previous sections, we end up with a version of our pipeline (see Figure 1), which can summarize a large conference of papers and produce a blog post out-of-the-box. After the experimentation of the components in isolation which was conducted for the SDS and the MDS stages in Sections 4.2.2 and 4.3.2, and based on our intuition about what would make a blopost pleasant, we propose the following setting for our end-to-end system for conference summarization:

- Clustering of the whole collection using the VosViewer implementation of the Leiden algorithm

 # (a) Cluster titles:
Given the following summaries, create one very short topic title for each summary using at most 3 words. Avoid using verbs. The topic titles should be representative of the summary. Avoid repetition across topic titles and make each title unique. Focus on what makes each topic different than the others.

1: {{*cluster_1_intro*}}
2: {{*cluster_2_intro*}}
...

 # (b) blog post surroundings (title, introduction, conclusion):
You are asked to write a blog post about an upcoming conference, that will be published in a very popular online journal read by experienced researchers. You have already written all the main sections of the blog post, but you are still missing the title, the introduction and the conclusion.
Given the following conference general information and the given topic summaries, complete the following tasks:
A. Write a short but appealing title for the blog post, that differentiates it from other conferences.
B. Write a couple of paragraphs as the introductory section of the blog post about the upcoming conference that will be placed before the blog post sections you have already written. The introductory section should include: the link to the conference website, the most important information about the conference. Focus also, on the underlying trends and the common aspects of the conference. Make it informative and appealing in order to motivate the reader continue reading the blog post. Avoid repetitive information.
C. Write a very short conclusion about the upcoming conference that will be placed after the blog post sections you have already written.
- Conference information:
name: {{*name*}}
website: {{*website*}}
location: {{*location*}}
start_date: {{*start_date*}}
end_date: {{*end_date*}}

- Topic summaries:
{{*cluster_1_intro*}}
{{*cluster_2_intro*}}
...

Table 5: Prompt Templates for generating (a) a title for each cluster using the generated cluster introductory paragraphs, (b) the blog post surroundings that include the title, the introduction and the conclusion of the blog post, given the conference metadata and the introductory paragraphs of the clusters used as summaries of the topics that are discussed in the conference.

- Selection of 10 representative documents per cluster
- Parallel SDS of the selected documents using gpt-3.5-turbo as the instruction-tuned backbone LLM with the prompt suggested in Table 1 in order to generate short summaries of 10 words.
- Parallel MDS of clusters given the single-document summaries of the previous step, using gpt-3.5-turbo as the backbone LLM, in order to generate cluster introductory paragraphs of 30 words and summaries with references of 100 words with the prompt templates presented in Tables 2 and 4 respectively. One exemplar generated through our iterative interaction with GPT-4 was used in both prompts to guide the format of the output.
- Parallel generation of cluster titles and blog post surroundings (title, introduction, conclusion) using gpt-3.5-turbo with the prompts presented in Table 5.

3.5.3 Baseline end-to-end setting

As already mentioned, the lack of MDS datasets that scale to a large collection of input documents, such as the case of a conference, creates a challenge in evaluating the impact of selecting representative documents per cluster before generating the summary. Taking also into consideration that the clustering of the documents and the 'blog post_creator' components are kept fixed since they are not our main research focus, we decided to use a baseline setting of our system that omits the selection of representative documents and keeps the remaining components the same. In this way, we can compare our proposed setting to the baseline variant and inspect if the selection of representative documents can lead to improvements.

As a result, our baseline setting contains the exact same steps, but instead of selecting representative documents per cluster, it passes directly to the summarization components, where it summarizes all the papers with the same SDS setting as the proposed variant, and then uses these summaries in the MDS stage with the exact same prompts. The difference is that due to the increased number of input documents in the prompt, there is no space for examples, so the baseline variant uses the zero-shot setting of the prompts and passes as many documents as possible.

4 Experiments and Results

As already mentioned, in this work we are not going to compare different clustering algorithms. Instead, the Leiden algorithm described in Section 3.1 is used, and the produced graph output of the algorithm is given as input to the next stages of our system. Taking into consideration that we have a multi-stage summarization module whose components include the selection of representative documents, the SDS stage and then the aggregation MDS stage, the evaluation of the whole system becomes a challenging task. This becomes even harder when the desired output is a blog post about a conference, since the summaries of each cluster are not enough, and the generation of titles, introduction, and conclusion of the blog post is also needed. Moreover, the current summarization datasets, especially in the scientific domain, are limited. Specifically, they are focusing only on SDS tasks, or on MDS tasks that have just a few source documents per sample. Even the BigSurvey (Liu et al., 2023a) dataset, which scales to more than 50 documents, does not fully meet our needs, since there are conferences with thousands of published articles.

Based on the above, to evaluate our proposed system, we decided to evaluate smaller

components in isolation starting from the smallest to the largest set of components that can be tested using the summarization datasets at hand. Firstly, we are going to evaluate the SDS stage using a respective dataset from the scientific domain, where the abstract of an article is given as input, and a short summary is expected as output, focusing on the main idea of the article. Secondly, we evaluate the MDS stage on a dataset where the number of documents that are given as input can fit into the transformer architecture. Thirdly, we need to scale to larger collections of documents, where the system will be tested end-to-end since the remaining stages will also be used. In this case, since there are no relevant datasets at hand, we are going to perform a qualitative user study, where annotators are asked for their preferences on blog posts about a large conference. Thus, we will examine the impact of selecting representative documents and using an exemplar in the prompt in order to guide the style of the summary.

4.1 Evaluation setup and metrics

For the evaluation on the target summarization datasets, we record ROUGE scores, but also BLEU and BERTScore since they seem to align more with human judgment (Zhang et al., 2019b; Graham, 2015).

For ROUGE (Lin, 2004), the following formula was used to calculate the metric for a given set of reference summaries:

$$\text{ROUGE-N} = \frac{\sum_{\text{S} \in \text{reference summary}} \sum_{\text{n-gram} \in \text{S}} \text{count}_{\text{match}}(\text{n-gram})}{\sum_{\text{S} \in \text{reference summaries}} \sum_{\text{n-gram} \in \text{S}} \text{count}(\text{n-gram})}$$

where n represents the n-gram order, $count_{match}$ represents the number of n-grams that co-occur in the generated summary and the reference summaries, and $count$, the total number of n-grams in the reference summaries. For n, values $1, 2, L$ were used, annotated as ROUGE_1, ROUGE_2, ROUGE_L respectively, where L refers to the longest common sequence. There are also several variants based on the preprocessing applied to the summaries, such as stemming and handling of stopwords. In our case, to provide a fair comparison, we used the same variants as the ones recorded by the baseline fine-tuned models on each summarization dataset.

Regarding BLEU, the formula suggested by the authors (Graham, 2015) is:

$$\text{BLEU} = \text{BP} \times \exp\left(\sum_{n=1}^{N} \text{weight}_n \cdot \log(\text{precision}_n)\right)$$

where BP is the brevity penalty term, N is the maximum n-gram order, while weight_n and precision_n represent respectively the weight and the precision for the corresponding n-gram order.

Finally, BERTScore is based on the meaning similarity between the generated and the reference summary and is given by the formula:

$$\text{BERTScore} = \frac{1}{N} \sum_{i=1}^{N} \max_{j=1}^{M} \text{cosine\_similarity}(\mathbf{C}_i, \mathbf{R}_j) \tag{3}$$

where $\mathbf{C}_i$ represents the BERT embedding of the i-th token in the generated summary, $\mathbf{R}_j$

represents the BERT embedding of the j-th token in the reference summary, whereas N and M are respectively the number of tokens in the generated and the reference summaries. For each token in the generated summary, we calculate the cosine similarity between its embedding and all token embeddings of the reference summary, and we keep the maximum similarity. BERTScore is the average of these maximum similarities. To produce the embeddings, the SciBERT language model[8] was used since it is pre-trained on scientific articles. We note though that similar trends in the results were observed with the RoBERTa model as well.

4.2 Single-document summarization

Given that the whole pipeline can be applied to collections of thousands of documents, where one short summary will be generated for each cluster, it is safe to assume that only a small portion of the information contained in the source documents will end up in the output summary. Thus, the summary is expected to focus on the main idea of the documents in a cluster. This use case is closely related to extreme summarization, which as an SDS task, aims at creating a short one-sentence summary.

4.2.1 Experimental setup

In the scientific domain, SciTLDR is a popular dataset of extreme summarization, which introduces the idea of TLDR generation. As stated by the creators of this dataset, this task involves high source compression and expert background knowledge in order to generate a TLDR summary of a scientific article. SciTLDR is a multi-target dataset of 3.2K papers, which contains 5.4K TLDRs which are both author-written and expert-derived, collected using the annotation protocol proposed by the authors (Cachola et al., 2020).

In order to investigate whether an instruction-tuned LLM can outperform finetuned models, we are going to compare the SDS stage of our pipeline using the prompting variants presented in Table 1 and gpt-3.5-turbo as the backbone LLM, against the following baseline summarization models finetuned on SciTLDR dataset:

- CATTS (Cachola et al., 2020): a learning strategy proposed by the authors of the SciTLDR dataset, which uses control codes to incorporate scaffold tasks such as title generation for denoising autoencoders like BART. Combining BART with the CATTS learning method leads to promising results in SDS tasks such as SciTLDR. In our experiments, the BART-large variant of the model was used, finetuned on the SciTLDR dataset[9]. As suggested by the authors, the number of beams was set to 4, and the length penalty factor to 0.2.
- PEGASUS (Zhang et al., 2020): we use the PEGASUS-large variant of this model, finetuned on the SciTLDR dataset[10].
- T5 (Raffel et al., 2020): we use the T5-base variant of the model in our experiments finetuned on the SciTLDR dataset[11], whereas the proposed configuration included the length penalty set to 2.0, the max and min length to 200 and 30 respectively, and the number of beams set to 4. As an extension of the experiment, we can also fine-tune the T5-large model on SciTLDR, but at the moment this is left as future work, since

[8] https://huggingface.co/allenai/scibert_scivocab_uncased
[9] https://huggingface.co/lrakotoson/scitldr-catts-xsum-ao
[10] https://huggingface.co/alk/pegasus-scitldr
[11] https://huggingface.co/HenryHXR/t5-base-finetuned-scitldr-only-abstract

Method	ROUGE_1	ROUGE_2	ROUGE_L	BLEU	BERTScore
CATTS (ours)	0.416	0.212	0.359	0.027	0.696
CATTS (reported)	0.438*	0.213*	0.359*	-	-
PEGASUS	0.399	0.208	0.346	0.125	0.692
T5	0.403	0.208	0.351	0.143	0.696
gpt-3.5-turbo					
vanilla prompt	0.408	0.200	0.339	0.110	0.699
our 0-shot prompt	0.429	0.205	0.354	0.124	0.704
our 2-shot prompt	**0.444**	**0.223**	**0.374**	**0.144**	**0.712**

Table 6: SDS results on SciTLDR dataset. Max ROUGE scores with stemming and stopword removal, BLEU and BERTScore are recorded. The '*' indicates that the results are presented as reported by the authors (Cachola et al., 2020). Regarding gpt-3.5-turbo results, the 'vanilla prompt' was based on a simple prompt for summarization, previously tried in the news domain (Goyal et al., 2022), 'our 0-shot prompt' refers to the prompt described in Section 3.3.3, and 'our 2-shot prompt' refers to the same prompt enhanced with two exemplars from the training set.

the effort to evaluate our system end-to-end should also be allocated to the remaining components of the pipeline.

Regarding the preprocessing of the summaries for the ROUGE-max calculation, stemming and stopword removal were used in order to follow the same configuration as the authors of the CATTS method (Cachola et al., 2020). This configuration also aligns with the suggested usage of the metric according to correlations with human judgment (Graham, 2015).

4.2.2 Results

The results of the SDS experiment are presented in Table 6. Firstly, we observe that the ROUGE scores presented by our reproduced version of CATTS match with high precision the scores reported by the authors (Cachola et al., 2020). Secondly, it seems that switching from the vanilla summarization prompt to our prompt, described in Section 3.3, leads to an important improvement, whereas using a two-shot setting with static exemplars from the training set of SciTLDR increases further the performance over the zero-shot prompt leading to the highest scores across all metrics. Among the fine-tuned models, CATTS and T5 performed similarly in terms of ROUGE metrics, with PEGASUS being slightly worse. However, regarding BLEU and BERTScore, CATTS scored quite lower than PEGASUS and T5, with the latter almost reaching the performance of the best '2-shot gpt-3.5-turbo' method.

At this point, it is important to note that summarization metrics are not always aligned with human preference. Previous work comparing news summaries produced by the vanilla prompt against summaries by fine-tuned models has indicated that even when automatic metrics favor summaries from fine-tuned models, human judgment tends to favor summaries generated by instruction-tuned models (Goyal et al., 2022). In terms of ROUGE scores, the vanilla prompt used by the authors in the news domain follows the same trend in the scientific domain, resulting in inferior results compared to the fine-tuned models. However, we show that aligning the prompt with the task at hand is crucial since both 0-shot and 2-shot prompts lead to better ROUGE scores for the summaries of the instruction-tuned LLM.

Method	average length in words
CATTS	7.48
PEGASUS	33.62
T5	29.21
gpt-3.5-turbo (vanilla prompt)	33.38
gpt-3.5-turbo (our 0-shot prompt)	19.22
gpt-3.5-turbo (our 2-shot prompt)	18.92

Table 7: Average summary length in words per SDS model on SciTLDR test set. Our suggested prompt leads to length of summaries that aligns with the average summary length of SciTLDR training set which is approximately 20 words.

Since the main focus of our work is not to evaluate extensively the single-document summaries but to provide a system that can summarize large collections of documents, we did not conduct a separate human evaluation on the SDS stage. Instead, we decided to inspect further the structure of the summaries produced by each model.

Specifically, we can observe the length of the produced summaries in Table 7 and two sample summaries per model in Table 8. As shown, CATTS method tends to generate very short summaries which are interpreted more as headlines of the papers and less as summaries. This is attributed not only to the fact that several summaries in the training set of SciTLDR have this format, but also to the objective of the CATTS learning strategy which uses a title generation task to help the summarization model. Also, T5 and PEGASUS produce longer summaries, which do not respect always the goal of extreme summarization. In addition, gpt-3.5-turbo with the vanilla prompt, where a summary of one sentence is requested, results in longer summaries too, although the instruction to produce one sentence is respected. On the other hand, prompting the model to produce summaries of specific length aligns the model more with the target dataset. In other words, after finding that the average word length of the summaries in the training set is approximately 20 words, we prompted the model to generate summaries of such length. The outcome reveals the ability of gpt-3.5-turbo to adapt to this instruction, which is a decisive factor in the improvement of the summary quality. Last but not least, we can observe that in the 2-shot setting, there are also summaries that correspond mostly to headlines and not to full sentences, which is expected taking into consideration that one of the two exemplars given to the prompt had this format due to the gold summaries of SciTLDR.

Given the slightly better performance of gpt-3.5-turbo against the finetuned versions of the aforementioned summarization models, we can conclude that gpt-3.5-turbo has strong summarization abilities and can be used out-of-the-box for SDS of scientific articles, without the need for a finetuning stage on a downstream supervised task. Combined with its ability to adapt the output in a specific style based on the given prompt, gpt-3.5-turbo and in general instruction-tuned models seem to be an appealing option for our use case.

4.3 Multi-document summarization

In the proposed system, there are two summarization stages where the first produces short summaries (TLDRs) of the documents of a cluster, and the second aggregates these TLDRs into a single paragraph that summarizes the whole cluster. The power of instruction-tuned LLMs lies in their ability to follow instructions, which can adapt the output summary according to specific use cases. In order to see the effectiveness of our two-step approach, we are going

Method	sample summary
CATTS	**#1**. Multi-view Learning of Sentence Representations. **#2**. Two-Step Defense against Adversarial Attacks.
PEGASUS	**#1**. Motivated by the asymmetry in the two hemispheres of the human brain as well as the observation that different learning architectures tend to emphasise different aspects of sentence meaning, we present two multi-view frameworks for learning sentence representations in an unsupervised fashion. **#2**. We show how strong adversarial examples can be generated only at a cost similar to that of two runs of the fast sign gradient method (FGSM), allowing defense against adversarial attacks with a robustness level comparable to that of the adversarial training with multi-step adversarial examples.
T5	**#1**. We present two multi-view frameworks for learning sentence representations in an unsupervised fashion. one uses a generative objective and the other a discriminative one. **#2**. In this paper, we show how strong adversarial examples can be generated only at a cost similar to that of two runs of the fast gradient sign method (FGSM).
gpt-3.5-turbo vanilla prompt	**#1**. The article presents two multi-view frameworks for learning sentence representations in an unsupervised fashion, which provide improved representations over their single-view learnt counterparts and demonstrate solid transferability on standard downstream tasks. **#2**. Researchers have developed a two-step defense approach that generates strong adversarial examples at a cost similar to the fast gradient sign method, allowing for defense against adversarial attacks with a robustness level comparable to that of multi-step adversarial examples.
our 0-shot prompt	**#1**. Multi-view learning with generative and discriminative frameworks improves sentence representations using self-supervision and distributional hypothesis. **#2**. Adversarial attacks on deep neural networks can be defended with a two-step approach that generates strong adversarial examples at low computational cost.
our 2-shot prompt	**#1**. Multi-view learning with distributional hypothesis for unsupervised sentence representation learning, with improved performance over single-view representations. **#2**. A two-step defense approach generates strong adversarial examples at a cost similar to that of the fast gradient sign method.

Table 8: Two sample summaries per SDS method on SciTLDR test set.

to use an MDS task of small collections of scientific articles, before scaling to large collections of documents.

4.3.1 Experimental setup

In the scientific domain, Multi-XScience (Lu et al., 2020) contains summaries of a few articles, with an average of 4.42 documents per summary. The purpose of this dataset though is not a generic summarization, but the generation of a related work paragraph of a paper, that refers to a given set of other papers. Thus, we adapted the prompts as described in Section 3.4 and we used this dataset to inspect the quality of our approach in order to guide our decisions about the summarization stages of our pipeline.

A vanilla prompt to generate the summaries for the Multi-XScience dataset would be to provide the abstracts of all the source documents in the prompt and ask for a generic summary. However, since the underlying task is closely related to the related work sections of papers, we decided to adapt the prompt and instruct the model with the details of the exact task, as shown in the prompt templates in Table 2. As mentioned, our pipeline suggests a two-step approach for summarizing multiple documents, but somebody could argue that for a small collection of source documents, giving the full content in a single prompt could be more effective and by default less expensive since we do not have to perform multiple calls to the instruction-tuned model. For that reason, we decided to compare both variants and examine if the two-step approach performs better.

We compare our instruction-tuned GPT model against fine-tuned models which are designed for MDS tasks. Based on the results reported on the Multi-XScience dataset, we use as baselines PRIMERA and Pointer-Generator which are explained in more detail in Section 3.4. For each model, the following results are reported:

- PRIMERA (Xiao et al., 2021): the authors reported the ROUGE scores on Multi-Xscience for two variants of PRIMERA, where one is fine-tuned and the other one is zero-shot indicating the generalization of the model. Results from both variants are presented.
- Pointer-Generator (See et al., 2017): the authors of PRIMERA (Xiao et al., 2021), reproduced the experiments of the Pointer-Generator model which achieves state-of-the-art in Multi-XScience dataset (Lu et al., 2020). They recorded the ROUGE scores using the same variant of the metric as in the evaluation of the PRIMERA model[12].

4.3.2 Results

The outcome of our experimentation is shown in Table 9, where the baseline models and the two gpt-3.5-turbo approaches are presented. It should be noted that the BLEU and BERTScore metrics are not reported for the PRIMERA and Pointer-Generator models, but we decided to include them for our methods since they are more likely to align with human preference (Graham, 2015; Zhang et al., 2019b). As observed, Pointer-Generator remains at the top with regards to ROUGE_L score, while in terms of ROUGE_1 and ROUGE_2, the gpt-3.5-turbo two-step approach achieves the highest scores. The fine-tuned variant of PRIMERA achieves comparable results to Pointer-Generator, while the zero-shot version seems to be quite worse than the gpt-3.5-turbo variants.

The most important observation though is that by using the two-step approach of our system, we can improve the performance over the one-step approach, with regard to all the reported

[12]They used https://github.com/google-research/googleresearch/tree/master/rouge with default stemmer settings

metrics. This could be attributed to our initial intuition that giving the full content of all source documents in a single prompt might confuse the model, as it would be harder to focus on the parts that really matter for the task at hand. Instead, based on the fact that in extreme summarization a small portion of the source information reaches the end summary, providing only the summaries of the papers in a final MDS prompt makes it easier for gpt-3.5-turbo to handle the input and facilitates the generation of the multi-document summary.

Method	$ROUGE_1$	$ROUGE_2$	$ROUGE_L$	**BLEU**	**BERTScore**
Pointer-Generator	0.339*	0.068*	**0.182***	-	-
PRIMERA (finetuned)	0.319*	0.074*	0.180*	-	-
PRIMERA (0-shot)	0.291*	0.046*	0.157*	-	-
gpt-3.5-turbo one-step	0.347	0.076	0.169	0.070	0.598
gpt-3.5-turbo two-step	**0.354**	**0.084**	0.171	**0.092**	**0.605**

Table 9: Multi-Document Summarization results on Multi-XScience. Max ROUGE scores with stemming, BLEU and BERTScore are recorded. The '*' indicates that the results are presented as reported by the authors (Xiao et al., 2021). Regarding gpt-3.5-turbo results, the 'one-step' approach refers to a single prompt where the full content of all source documents is given at once, whereas the 'two-step' approach refers to an SDS stage followed by an MDS step, which takes as input the summaries produced during the SDS stage and generates the final multi-document summary.

4.4 Conference summarization

As we move to larger collections of documents, such as the publications of a conference, the evaluation is becoming more challenging not only due to the lack of related datasets but also due to the difficulty to judge what is actually a good summary. For that reason, we decided to generate fully automated blog posts of a large conference in AI, and conduct a user study asking for preference judgment and feedback. Since our main concern is the generation of a summary for each cluster of documents in a large collection, it is important to understand the impact of selecting representative papers per cluster because otherwise, it is not possible to evaluate this stage with the Multi-XScience dataset which contains only a few documents per sample.

Specifically, since it is preferred to use the two-step approach, as demonstrated in Section 4.3.2, the prompt given to the MDS stage is much shorter than passing the full content of the source documents. Based on that, a logical thought would be to scale the input of this prompt to contain hundreds of document summaries. Nevertheless, our intuition, which is also supported by the results of the MDS experiments, indicates that as we increase the information given to the MDS prompt, it will be harder for gpt-3.5-turbo to consume all the necessary details and produce a summary that covers all the input documents. On top of that, this significant increase in the length of the prompt does not allow us to include exemplars in the prompt, that would guide gpt-3.5-turbo to generate summaries of specific style and structure.

4.4.1 User study design

To validate the aforementioned hypothesis about the selection of representative papers, we compare the proposed setting of our system presented in Section 3.5.2 against the baseline setting presented in Section 3.5.3. Moreover, as explained in Section 3.2, we can also incorporate bibliometric signals into the selection of the representatives per cluster, thus it would be interesting to observe the effect of this addition too. Finally, as far as we know, there

is no other system that generates automated blog posts of large collections of documents, which makes the comparison with an external baseline system impossible. For that reason, we decided to select a *'human written'* blog post which could also be considered as the gold standard of our case.

To perform these comparisons, we are going to setup the following user study to gather preference judgments and feedback from annotators:

- We use the ICLR 2023[13] conference as a case, which was hosted in Kigali of Rwanda in May 2023, where more than 2300 papers were published. The volume of this collection makes this conference a sufficient use case for the evaluation of our system. Moreover, the recency of the conference is another important factor, since it is unlikely for gpt-3.5-turbo or any other of the used models to have seen these papers during pre-training, which enables us to evaluate the generalization of our system in new information.
- The first part of the user study is to compare the users' preference between two variants of our proposed system (see Section 3.5.2), where the first one uses only the similarities between documents to select representative papers per cluster as described in Section 3.2.2, while the second one incorporates bibliometric signals as well (see Section 3.2.3). We call the first variant [14] *'coverage'* and we use $\beta = 1$ in the formula 2, while for the second variant, we use $\beta = 0.5$ in order to assign equal weights to *SimilarityStrength* and *Impact*. Thus, we name the latter variant [15] *'coverage+impact'*.
- The second part is to compare the preferred choice of the user from the first part, against a *'human written'* blog post. Taking into account that the clustered documents given as input to our system, are based on the clustering implementation of Zeta Alpha, we decided to use as the *'human written'* variant, a blog post that was written by an AI analyst of Zeta Alpha, who is also using the same clustering feature to gain insights into the conference. It should be noted of course that although the quality of this *'human written'* conference blog post[16] is expected to be high, a significant amount of time is required by an AI analyst to digest the whole information and compile this blog post.
- The third and final comparison of the user study is to ask for preference between the variant selected by the user in the first part, and the baseline variant of our system, where no representative documents are selected, as explained in Section 3.5.3

For each of the above comparisons, we are going to present the three pairs to the annotator and ask for their preference based on the below criteria, which are inspired by previous work in human evaluation of summaries (Fabbri et al., 2021; Dang, 2005). It should also be noted that a 'No preference / I do not know' option was given.

- Overall: the general preference of the annotator between the two blog posts
- Fluency: the quality of individual sentences
- Factuality: Which blog would you trust more that provides a factually consistent summary of the conference?

[13] https://iclr.cc/

[14] https://zav-vos-viewer.s3.eu-central-1.amazonaws.com/data/automated-blog/ICLR-2023-coverage-final/VOS-ICLR-2023.html

[15] https://zav-vos-viewer.s3.eu-central-1.amazonaws.com/data/automated-blog/ICLR-2023-balanced-temperature/VOS-ICLR-2023.html

[16] https://www.zeta-alpha.com/post/a-guide-to-iclr-2023-10-topics-and-50-papers-you-shouldn-t-miss

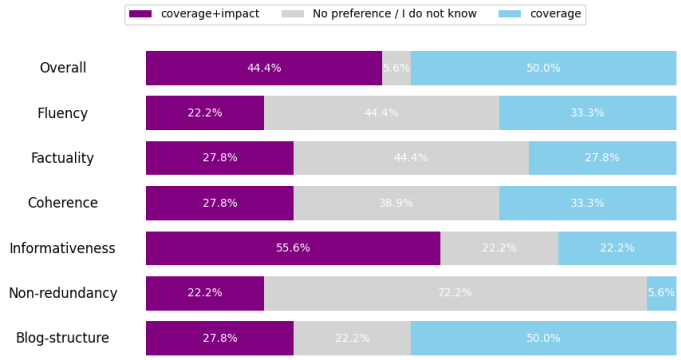

Figure 3: Preference judgment comparison between *'coverage+impact'* and *'coverage'* variants of the proposed setting of the system described in Section 3.5.2.

- Coherence: Each blog should build from sentence to sentence to a coherent body of information about a topic. Based on that, which blog is more coherent for you?
- Informativeness: Each blog should cover the conference as broadly as possible in order not to miss any potentially relevant and important information. Which blog would you prefer based on that?
- Non-redundancy: There should be no unnecessary repetition in the summary. Unnecessary repetition might take the form of whole sentences that are repeated, or repeated facts, or the repeated use of a noun or noun phrase.
- Blog Structure: Which blog would you prefer in terms of blog structure (sections, section titles, length, overall structure)?

At this point, it is worth mentioning that some of these criteria are more challenging to evaluate, because it is infeasible for the annotator to consume all the source documents of the ICLR conference and judge if the blog post is as informative or factual as possible. Nonetheless, given that any reader consuming text content can intuitively assess its informativeness and factuality, we can still seek the reader's opinion on which blog post provided them with more useful information according to their intuition, and which one they trusted more.

4.4.2 Results

Given the described user study, we gathered responses from 18 annotators with experience in the field as MSc or PhD students or industry employees in AI or Data Science.

Comparison of 'coverage+impact' vs 'coverage'

In Figure 3 we can observe the results of the first pair that was presented, as explained in Section 4.4.1. As depicted from the figure, the annotators seem to slightly prefer the *'coverage'* blog post variant, indicating that the way we incorporated bibliometric signals did not favor the output. Nevertheless, after delving further into the feedback comments, this conclusion does not seem to be justified in most of the cases. To be more specific, annotators who prefer one of these two variants are not basing their choice on the quality of the selected papers, but on the writing style and how each blog post is presenting and explaining the provided information.

As a consequence, all of these observations cannot be attributed to the selection of the representative documents, but to the randomness of the model when generating summaries.

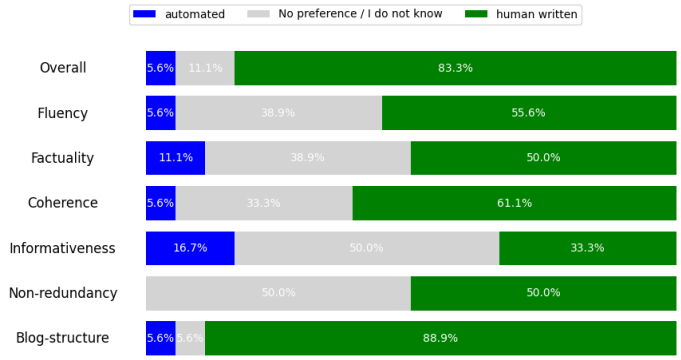

Figure 4: Preference judgment comparison between *'automated'* and *'human written'* blog posts. The *'automated'* blog post is either the *'coverage'* or the *'coverage+impact'* variant, depending on the user's choice in the first pair comparison. The *'human written'* version is the blog post[17] written by an AI analyst of Zeta Alpha.

That suggests that if we scale our experiment to more collections of documents, the aforementioned comments are expected to be evenly distributed between these two variants. Based on the above, we believe that it is not safe to draw conclusions regarding the preference of the annotators between the *'coverage'* and the *'coverage+impact'* versions. However, we can conclude that judging these variants is a very hard task for humans, since it is impossible to digest the information from thousands of papers and decide which papers are more represenative than others. An approach worth considering entails assigning annotators who possess expertise in specific document collections to participate in the same user study. Nevertheless, it is important to acknowledge that even in such cases, subjective biases will influence the quality of preference judgments.

Comparison of 'automated' vs 'human written'

In Figure 4, we present the comparison between the *'human written'* and the automated blog post that was selected by each annotator during the first pair. As observed, it is clear that the *'human written'* blog post dominates the preference judgments, even in terms of 'Fluency' and 'Coherence' which are known to be solved issues for LLMs such as gpt-3.5-turbo. After manually inspecting the blog posts and the comments from the annotators, we believe that a preference based on these aspects cannot be clearly justified by the content of the summaries. Most comments refer to the user-friendliness of the *'human written'* blog post, but this aspect should not affect the fluency judgment. Probably, we could attribute this difference to the fact that an annotator favoring overall one variant is more likely to favor this variant in terms of every other aspect as well.

Despite the clear preference of the *'human written'* blog post, some of the provided aspects such as 'Informativeness', 'Non-redundancy', and 'Factuality' posed a challenge to several annotators making it more difficult for them to decide which blog post they prefer. This resulted in a higher percentage of 'No preference / I do not know' choices and even in a few choices of the *'automated'* blog post, which is a promising sign since for at least 50% of the annotators, it seems that the *'automated'* version can approach the quality of the *'human written'* version in these aspects. Taking also into consideration the significant amount of effort and time needed by an AI analyst to create such content, we can conclude that automating this process using instruction-tuned models can be effective and time efficient.

Of course, there are more improvements to consider in order to surpass human-level performance, which can be guided by the received feedback that revealed the weaknesses of the *'automated'* blog post when compared to the *'human written'*. Specifically, annotators tend to prefer the *'human written'* variant, because it is more readable as they mention, and has a structure that is easier to follow. Despite the fluency of gpt-3.5-turbo, it seems that the generated summaries tend to follow a similar format which is also encouraged by the design of our pipeline to control the output in order to mitigate hallucinations. Thus, even though the content might be informative, the user-friendliness of the blog post could not approach the user-friendliness of the *'human written'* blog post. As a result, conducting further research in this direction could lead to significant improvements.

Last but not least, several comments mentioned that they preferred the information of each paper to be presented as a list of bullet points as in the *'human written'* version because it was easier for them to follow the content. For that reason, it would be a useful amendment to keep the first generic paragraph of each section unchanged and present the more detailed paragraph about the papers of the cluster in a list format.

Comparison of 'automated' vs 'baseline'

Finally, in Figure 5, we show the preference judgments for the third pair of our survey, where the *'automated'* variant is compared to the *'baseline'* blog post. As expected, prompting the model with a set of representative documents and providing an exemplar in the prompt leads to summaries that are preferred over prompting the model with the entire list of documents without ICL. There is still though, an important number of annotators selecting the 'No preference / I do not know' choice, or even the *'baseline'* version. From the received feedback, they liked the fact that the *'baseline'* method could include a description of a larger number of documents. We believe though that the difficulty in judging if the presented content is representative of the whole conference led to these comments.

To be more specific, our intuition is that the papers that were chosen by the model to be in the final summary of the *'baseline'* variant are highly affected by their order in the prompt, and less by their actual content. To further support this view, we investigated the underlying distribution of the papers that were selected by the model as references in the final summary. We produced 500 random perturbations of 100 papers of a cluster and prompted the MDS stage of the system with their summaries. The result of this experiment was 500 summaries for the same cluster that contain references to specific documents.

If the selection of the papers was robust against different perturbations, we would expect the same papers to be picked every time. However, in Figure 6a, we can observe that this assumption does not hold, since even the most cited paper is cited in less than 35% of the generated summaries. On the other hand, if we examine the distribution of the indices that are selected by the model independently of which document corresponds to each index (Figure 6b), we can see that the model is highly biased towards the papers presented first, even if we explicitly mention in the prompt that the given order is random. For instance, the first 5 papers in the prompt seem to be picked around 50% of the times by the model. As a result, the order of the papers seems to highly affect the output of the LLM, which reveals the weakness of such model to process a large amount of diverse information in a robust way.

Regarding the other aspects of the evaluation, even though the same LLM is used to generate both blog posts, the 'Fluency' and the 'Coherence' of the output were in favor of the

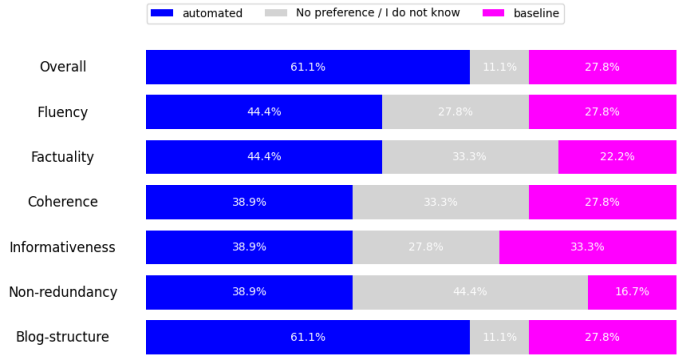

Figure 5: Preference judgment comparison between 'automated' and *'baseline'* blog posts. The 'automated' blog post is either the 'coverage' or the 'coverage+impact' variant, depending on the user's choice in the first pair comparison. The *'baseline'* version is the blog post generated using the baseline setting described in Section 3.5.3.

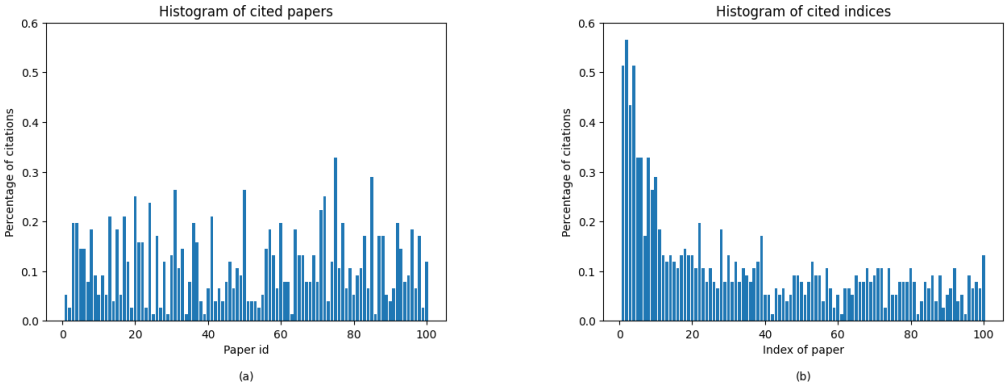

Figure 6: Histogram of papers (a) and indices (b) that were cited, after 500 iterations of the MDS stage of our pipeline in a single cluster, where 100 papers were included in the prompt.

automated version. This time, we can attribute this outcome to several cut off paragraphs in the *'baseline'* blog post which revealed the weakness of the model in generating all the summaries using less than the 512 tokens which was set as the maximum limit. This was mentioned by several annotators who preferred the automated variant, indicating that when the model is prompted with a large number of documents and without ICL, it cannot follow accurately the instructions about the length of the summary. It is also worth noting that the annotators that did not declare any preference or preferred the *'baseline'* blog post, did not mention anything about the cut off paragraphs, making us wonder if they observed this issue. Concerning factuality, the *'baseline'* blog post was also dominated, since the annotators did not like the fact that two of the cluster summaries did not contain references, as the model could not follow the relevant instruction in all cases. This issue, combined with the lengthy output of the *'baseline'* approach, resulted also in a higher preference of the automated version in terms of 'Blog structure'.

On the other hand, the 'Informativeness' aspect divided the annotators. Based on the feedback, those who preferred the *'baseline'* commented that it usually contained more referenced documents, whereas the annotators that preferred the 'automated' focused also on the usability of the summary. We would also agree that providing larger amounts of information by presenting summaries of numerous papers in each cluster would result in an unusable and consequently, non-informative summary, compared to a summary that thoroughly selected which papers to present in order to facilitate the digestion of the information. Probably, if we rephrased the explanation of the 'Informativeness' aspect to also point out the need for usability, we could reproduce the aforementioned conclusion in our preference judgment results as well. In general, it is observed that there is a correlation between the preference of the users in different aspects, which may affect the objectivity in the way they answer the questions, a phenomenon already known in the literature as anchoring bias (Echterhoff et al., 2022).

5 Discussion

5.1 Answers to the Research Questions

RQ1. How does a instruction-tuned generative LLM compare to fine-tuned LLMs in Single-Document (SDS) and in Multi-Document Summarization (MDS)?
In this work, we demonstrated that instruction-tuned LLMs like gpt-3.5-turbo show strong summarization abilities and can be used out-of-the-box in the scientific domain without the need for fine-tuning on downstream supervised tasks. Specifically, in SDS, adapting the prompt to the task at hand and using In-Context Learning (ICL) leads to significant improvements that outperform state-of-the-art fine-tuned models. Furthermore, in MDS, instruction-tuned LLMs perform better compared to summarization models used in a zero-shot setting, and comparable to models that are fine-tuned on the target dataset. Last but not least, the ability of instruction-tuned LLMs to adjust the summary according to the given task is prominent and useful when summarizing large collections of papers, such as in the case of generating a conference blog post.

RQ2. How does a two-step approach, where we firstly summarize the source documents and then we generate the multi-document summary of the generated summaries, compare to a one-step approach where the full content of the source documents is given as input to the model?
We conclude that using a two-step approach leads to a slight improvement over the one-step

approach. We attribute this behavior to the fact that it will be easier for an LLM to handle a prompt with more concise content. Specifically, in the case of extreme multi-document summarization where a small portion of the source documents ends up in the output summary, the information loss caused by feeding the LLM with only the summaries of the source documents instead of the full content does not harm the performance. Instead, it facilitates the handling of information by the model achieving a performance gain. The two-step approach, of course, depends on the quality of the single-document summaries, which was shown to be high in our SDS experimentation, thus we can safely assume that providing the MDS step with accurate single-documents summaries justifies the aforementioned performance gain.

RQ3. What is the effect of selecting representative documents per cluster on the generation of the summary of a large collection such as the case of a conference? Through our user study containing preference judgments and feedback from the annotators, we indicated that selecting a handful of representative papers per cluster before generating the summary for each cluster leads to blog posts that are slightly preferred by humans. While we were expecting the difference to be larger, we understood from the comments of the user study that judging several important aspects of the evaluation, such as informativeness and factuality is a difficult task and sometimes infeasible when the collection is so large. By performing a further analysis on the generated summaries, we indicate that instruction-tuned LLMs such as gpt-3.5-turbo are biased towards the papers that are presented in the first positions of the prompt, exactly after the task instructions. Thus, the selection of the papers by the model is driven mostly by their order in the provided list and not by their actual content.

From the findings of the previous two research questions, it can be inferred that when it comes to summarization, giving the instruction-tuned LLM a vast quantity of information from various documents poses a challenge for the model to handle effectively. Instead, by controlling the input provided in the prompt, the generation of a summary becomes easier. This is not only because concise information is given as input, but also because it allows the utilization of ICL that can guide the output format of the summary.

5.2 Conference summarization

An interesting application of our end-to-end system is to perform conference summarization where a fully automated blog post is generated to provide insights into a large collection of recently published papers.

To observe the effectiveness of this approach we conducted a user study with pairwise comparisons of different blog posts. When comparing our suggested pipeline (see Section 3.5.2) against a human-written blog post, we showed that even though there is a clear preference towards the human-written blog post, the comparable informativeness and factuality of our system demonstrate the potential of our approach. In addition, taking into consideration that the largest percentage of comments was focused on the readability and the user-friendliness of the human-written blogpost over the automated, we believe that iterating further in adjusting the blog structure can yield significant improvements. In other words, we can approximate the quality of the human-written blog post, just by making the output of our system more user-friendly, while keeping the content similar to the current version.

Finally, we also examined the impact of incorporating bibliometric signals in the selection of representative documents per cluster. This experiment was inconclusive because it was

infeasible for annotators to evaluate the differences in the selection of representative papers, due to the huge size of source documents. Thus, their preference was based on different criteria which cannot be attributed to the papers selected by the system.

5.3 Limitations and Future work

In this section, we delve into the limitations of our work and the potential future directions to mitigate these limitations and further enhance our system.

To begin with, the selection of representative papers is based on graph-based clustering where the edges of the graph represent the similarity between papers. Thus, it is not possible to use out-of-the-box a different clustering algorithm that is not graph-based, because it would require a modification in the representative documents detector as well. However, finding different ways to select representative documents would enable the usage of any clustering algorithm.

Moreover, incorporating bibliometric signals in the selection of representative documents was based on an arbitrary calculation of the impact score of a paper, which was designed based on the data we have at hand. Exploring further this field, and exploiting better ways to measure the potential impact of a paper is expected to yield better results in the selection process that precedes the summarization components.

Regarding the instruction-tuned LLM used as the backbone for the generation of the summaries and the blog post, our research is constrained to utilizing only the closed-source gpt-3.5-turbo LLM. On the other hand, the integration of different models and especially open-source LLMs would be an interesting and important direction. This approach might also require the adaptation of the prompts to the expected format for each LLM, but it would still be worth exploring in order to evaluate their effectiveness in summarization tasks.

In addition, increasing the number of participants in our user study would yield more reliable results and potentially safer conclusions. It is important to note though that from our analysis it seems challenging for humans to evaluate the selection of representative documents when the collection of source documents is so large. Thus, finding also different ways and automatic metrics to evaluate this part of the system would be beneficial in order to observe how the selection of papers affects the final summaries.

Besides the selection of representative papers, the evaluation of the factuality of a summary or a blog post is another hard task. Once again, due to the large number of input documents, it is infeasible for annotators to inspect this aspect in the evaluation process. Alternative ways to solve this issue, focus on using LLMs in the evaluation process, by asking explicitly an instruction-tuned model if a summary is factually correct based on a given source document (Liu et al., 2023b; Fu et al., 2023). In our case, this approach is not directly applicable due to the large input context, but evaluating in isolation the SDS and MDS steps would be probably worth exploring.

6 Conclusion

To sum up, we propose an end-to-end system for abstractive summarization of large collections of scientific documents and we demonstrate its effectiveness by producing a fully automated blog post that summarizes a large AI conference.

We show that using a two-step prompt-based summarization approach where the single-document summaries of the selected papers are generated in the first step and their aggregation

is performed in the second step leads to performance gain over prompting an LLM with the full content of source documents at once. We also evaluate the SDS and MDS stages of our pipeline in SciTLDR and Multi-XScience datasets respectively, demonstrating the strong summarization abilities of instruction-tuned LLMs when prompted with task-specific instructions. Specifically, our approach achieves similar or better results when compared to state-of-the-art fine-tuned models.

Moreover, we conduct a user study asking for preference judgment and feedback on pairs of blog posts that summarize a large AI conference. We show that selecting representative papers per cluster before prompting the LLM to produce the multi-document summary is preferred by humans over passing the summaries of all documents of a cluster in the prompt. Additional analysis also shows that the selection of papers by an LLM in the latter case is affected highly by the order of the papers in the given prompt and not much by their content.

In addition, we observe that incorporating bibliometric signals in the selection of representative documents does not lead to any improvement in the preference judgments by humans. However, after inspecting their feedback, we attribute this result to the challenge associated with evaluating this aspect in a large collection of papers.

Finally, even though it is clear that a human-written blog post is preferred over the automated one, there are promising signs that indicate the good informativeness and factuality of the summaries produced by our system. Most of the user feedback pointed out the user-friendliness of the human-written blog post and not its content, indicating that further improving the structure and the output format of the automated blog post will yield significant improvements.

References

Dimo Angelov. 2020. Top2vec: Distributed representations of topics. *arXiv preprint arXiv:2008.09470*.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew R Gormley. 2023. Unlimiformer: Long-range transformers with unlimited length input. *arXiv preprint arXiv:2305.01625*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel S Weld. 2020. TLDR: Extreme summarization of scientific documents. *arXiv preprint arXiv:2004.15011*.

Shiqi Chen, Siyang Gao, and Junxian He. 2023. Evaluating Factual Consistency of Summaries with Large Language Models. *arXiv preprint arXiv:2305.14069*.

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2022. *Introduction to algorithms*. MIT press.

Hoa Trang Dang. 2005. Overview of DUC 2005. In *Proceedings of the document understanding conference*, volume 2005, pages 1–12. Citeseer.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Jessica Maria Echterhoff, Matin Yarmand, and Julian McAuley. 2022. AI-moderated decision-making: Capturing and balancing anchoring bias in sequential decision tasks. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–9.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Marzieh Fadaee, Olga Gureenkova, Fernando Rejon Barrera, Carsten Schnober, Wouter Weerkamp, and Jakub Zavrel. 2020. A new neural search and insights platform for navigating and organizing AI research. *arXiv preprint arXiv:2011.00061*.

Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. 2023. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*.

Eugene Garfield. 2006. The history and meaning of the journal impact factor. *jama*, 295(1):90–93.

Tanya Goyal, Junyi Jessy Li, and Greg Durrett. 2022. News summarization and evaluation in the era of gpt-3. *arXiv preprint arXiv:2209.12356*.

Yvette Graham. 2015. Re-evaluating automatic summarization with BLEU and 192 shades of ROUGE. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 128–137.

Maarten Grootendorst. 2022. BERTopic: Neural topic modeling with a class-based TF-IDF procedure. *arXiv preprint arXiv:2203.05794*.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. LongT5: Efficient text-to-text transformer for long sequences. *arXiv preprint arXiv:2112.07916*.

Anis Koubaa. 2023. GPT-4 vs. GPT-3.5: A concise showdown.

Wojciech Kryściński, Nitish Shirish Keskar, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Neural text summarization: A critical evaluation. *arXiv preprint arXiv:1908.08960*.

Rafal Kuc and Marek Rogozinski. 2013. *Elasticsearch server*. Packt Publishing Ltd.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Miao Li, Eduard Hovy, and Jey Han Lau. 2023. Towards Summarizing Multiple Documents with Hierarchical Relationships. *arXiv preprint arXiv:2305.01498*.

Siqing Li, Wayne Xin Zhao, Eddy Jing Yin, and Ji-Rong Wen. 2019. A neural citation count prediction model based on peer review text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4914–4924.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Shuaiqi Liu, Jiannong Cao, Ruosong Yang, and Zhiyuan Wen. 2023a. Generating a structured summary of numerous academic papers: Dataset and method. *arXiv preprint arXiv:2302.04580*.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023b. Gpteval: Nlg evaluation using gpt-4 with better human alignment. *arXiv preprint arXiv:2303.16634*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Yixin Liu, Pengfei Liu, Dragomir Radev, and Graham Neubig. 2022. BRIO: Bringing order to abstractive summarization. *arXiv preprint arXiv:2203.16804*.

Yao Lu, Yue Dong, and Laurent Charlin. 2020. Multi-XScience: A large-scale dataset for extreme multi-document summarization of scientific articles. *arXiv preprint arXiv:2010.14235*.

Yuning Mao, Ming Zhong, and Jiawei Han. 2022. CiteSum: Citation Text-guided Scientific Extreme Summarization and Domain Adaptation with Limited Supervision. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10922–10935.

Pedro Henrique Martins, Zita Marinho, and André FT Martins. 2021. Infinite-former: Infinite Memory Transformer. *arXiv preprint arXiv:2109.00301*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.

Jason Phang, Yao Zhao, and Peter J Liu. 2022. Investigating efficiently extending transformers for long input summarization. *arXiv preprint arXiv:2208.04347*.

Jason Priem, Dario Taraborelli, Paul Groth, and Cameron Neylon. 2011. Altmetrics: A manifesto.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Chenhui Shen, Liying Cheng, Yang You, and Lidong Bing. 2023. A hierarchical encoding-decoding scheme for abstractive multi-document summarization. *arXiv preprint arXiv:2305.08503*.

Gabriela Surita, Rodrigo Nogueira, and Roberto Lotufo. 2020. Can questions summarize a corpus? Using question generation for characterizing COVID-19 research. *arXiv preprint arXiv:2009.09290*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Anton Thielmann, Quentin Seifert, Arik Reuter, Elisabeth Bergherr, and Benjamin Säfken. 2023. Topics in the Haystack: Extracting and Evaluating Topics beyond Coherence. *arXiv preprint arXiv:2303.17324*.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From Louvain to Leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):5233.

Mohamed Trabelsi and Huseyin Uzunalioglu. 2023. Absformer: Transformer-based Model for Unsupervised Multi-Document Abstractive Summarization. *arXiv preprint arXiv:2306.04787*.

Nees Van Eck and Ludo Waltman. 2010. Software survey: VOSviewer, a computer program for bibliometric mapping. *scientometrics*, 84(2):523–538.

Nees Jan Van Eck and Ludo Waltman. 2017. Citation-based clustering of publications using CitNetExplorer and VOSviewer. *Scientometrics*, 111:1053–1070.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Hannah R Warren, Nicholas Raison, and Prokar Dasgupta. 2017. The rise of altmetrics. *Jama*, 317(2):131–132.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Jeff Wu, Long Ouyang, Daniel M Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. 2021. Recursively summarizing books with human feedback. *arXiv preprint arXiv:2109.10862*.

Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. 2021. PRIMERA: Pyramid-based masked sentence pre-training for multi-document summarization. *arXiv preprint arXiv:2110.08499*.

Yichong Xu, Ruochen Xu, Dan Iter, Yang Liu, Shuohang Wang, Chenguang Zhu, and Michael Zeng. 2023. InheritSumm: A General, Versatile and Compact Summarizer by Distilling from GPT. *arXiv preprint arXiv:2305.13083*.

Rui Yan, Jie Tang, Xiaobing Liu, Dongdong Shan, and Xiaoming Li. 2011. Citation count prediction: learning to estimate future citations for literature. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1247–1252.

Haopeng Zhang, Xiao Liu, and Jiawei Zhang. 2023a. SummIt: Iterative Text Summarization via ChatGPT. *arXiv preprint arXiv:2305.14835*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019a. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019b. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2023b. Benchmarking large language models for news summarization. *arXiv preprint arXiv:2301.13848*.

Xingxing Zhang, Furu Wei, and Ming Zhou. 2019c. HIBERT: Document level pre-training of hierarchical bidirectional transformers for document summarization. *arXiv preprint arXiv:1905.06566*.

Zihan Zhang, Meng Fang, Ling Chen, and Mohammad-Reza Namazi-Rad. 2022. Is neural topic modelling better than clustering? An empirical study on clustering with contextual embeddings for topics. *arXiv preprint arXiv:2204.09874*.