Research Report

Query-Focused Document Summarization

Aaditya Bodke[1] **and Kevin Chang**[2]

[1,2]*Forward Data Lab, University of Illinois Urbana-Champaign*

This report was compiled on August 24, 2024

Abstract

Query-focused summarization (QFS) is a specialized area within automatic text summarization, where the objective is to generate summaries tailored to address a specific query by selecting and condensing relevant information from the given document(s). This research introduces a novel QFS framework that synergizes the precision of Elasticsearch with the generative capabilities of large language models (LLMs) to enhance information retrieval and abstractive summarization within the context of grant documentation. By leveraging Elasticsearch's robust indexing and semantic search capabilities, combined with LLMs' ability to produce coherent, contextually aware summaries and judgements, the framework aims to deliver concise, query-specific insights from extensive grant records.

Keywords: *query-focused summarization, QFS, search, LLM, elasticsearch, funding, grant*

1. Introduction

Query-Focused Summarization (QFS) is a specialized task in natural language processing (NLP) designed to generate summaries that directly address specific user queries by distilling the most relevant information from extensive documents. This task is particularly crucial in scenarios where users need to quickly access the most pertinent information from large datasets or lengthy texts, such as in academic research, legal document analysis, or grant application processes.

There are two primary approaches to query-focused summarization:

- **Extractive Summarization**: This method involves directly selecting sentences or phrases from the source document that are deemed most relevant to the query. While widely used in web browsers and search engines to generate snippets, extractive summarization often produces results that feel disjointed or lack coherence, as the selected sentences may not flow smoothly when combined.
- **Abstractive Summarization**: Unlike extractive methods, abstractive summarization generates new sentences that convey the essential information from the source text, often through paraphrasing, generalization, or simplification. This approach aims to produce summaries that are not only informative but also logically structured, well-organized, and grammatically sound, closely mimicking human-written text.

While extractive summarization is more prevalent due to its simplicity and computational efficiency, it often fails to capture the fluidity and coherence of human writing. In contrast, abstractive summarization, though technically more challenging and less widely implemented, offers the potential for more natural, readable summaries that better meet the needs of users. Given these advantages, this research focuses on abstractive summarization, leveraging its flexibility and coherence to enhance the quality of query-focused summaries, particularly in the context of grant search engines.

1.1. The Problem

Generating accurate and useful query-focused snippets in search engines and online marketplaces remains a critical challenge with significant implications for improving user experience. Effective QFS can greatly enhance document-based retrieval systems, enabling users to quickly identify the most relevant results. This capability is particularly valuable when users are searching for precise information within a large corpus of documents, such as lengthy articles, legal texts, or academic papers, where traditional search methods may fall short in delivering concise and contextually relevant information.

The need for effective query-focused summarization is especially pronounced in the domain of grant applications. Popular grant search engines, such as Grants.gov and GrantForward, serve as essential resources for identifying potential funding opportunities. However, these platforms currently lack the ability to provide users with abstractive snippets that quickly convey the relevance of each opportunity. This absence of concise, query-specific summaries can hinder users from efficiently evaluating the suitability of various funding options, ultimately affecting their ability to make informed decisions.

To address this challenge, our research proposes an advanced QFS framework that combines the robust search capabilities of Elasticsearch with the generative power of large language models (LLMs). This approach aims to improve the relevance, coherence, and readability of query-focused summaries, thereby enhancing the overall user experience in grant search applications.

2. Proposed Solution

To address the challenges of query-focused summarization in the context of funding opportunities, we propose GrantQuest, a comprehensive semantic search engine that enhances the browsing experience for users seeking grants. GrantQuest combines advanced retrieval and summarization technologies to deliver precise, relevant, and easy-to-understand information, enabling users to quickly assess the suitability of various funding options.

The architecture of GrantQuest comprises three main components: the Retriever, the Snippet Generator, and the Application Interface. Together, these components work in concert to provide a seamless and efficient grant search experience.

2.1. Architecture Overview

GrantQuest operates through a modular architecture where:

- **The Retriever** handles the retrieval of relevant documents using a combination of traditional and semantic search methods.
- **The Snippet Generator** processes the retrieved documents to generate concise, query-focused summaries using abstractive summarization techniques.
- **The Application Interface** serves as the front-end, allowing users to interact with the system, submit queries, and view results.

This architecture ensures that users receive not only relevant search results but also clear and concise summaries that directly address their queries. Figure 1 illustrates the architecture.

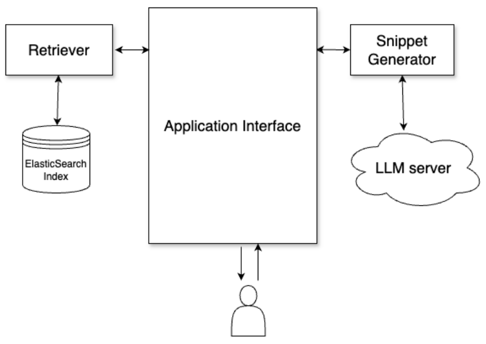


Figure 1. System Architecture

2.2. The Retriever - Elasticsearch

At the core of GrantQuest is Elasticsearch, a powerful and scalable search engine optimized for high-performance retrieval from large datasets. GrantQuest leverages Elasticsearch for both traditional full-text search and advanced semantic search, offering a comprehensive approach to retrieving relevant documents.

- **Full-Text Search**: In traditional full-text search, Elasticsearch retrieves documents based on keyword matching. It analyzes both the user's query and the indexed documents, breaking them down into tokens and matching these tokens against the content stored in the index. This method is highly effective for finding documents that contain exact or similar phrases to the user's input. However, it is limited by its reliance on the presence of specific keywords and may miss documents that are contextually relevant but do not contain those exact terms.
- **Semantic Search**: To overcome the limitations of full-text search, GrantQuest employs semantic search, which leverages embeddings stored in the Elasticsearch index. In semantic search, both the query and the documents are converted into dense vector representations (embeddings) that capture the underlying meaning of the text. By comparing these embeddings, the system can identify documents that are contextually similar to the query, even if they do not contain the exact keywords. This approach allows for a more nuanced and accurate retrieval of documents, ensuring that users receive results that are truly relevant to their informational needs.
- **Hybrid Search**: GrantQuest also integrates a hybrid search mechanism, combining the strengths of both full-text and semantic search. This hybrid approach allows the system to strike a balance between the capabilities of full-text and hybrid search, by taking a weighted sum of the socres returned by the two methods.

By combining these search capabilities, GrantQuest delivers more comprehensive and relevant search results, enhancing the user's ability to find the most pertinent funding opportunities.

2.3. The Snippet Generator - LLM Server

Once the relevant documents are retrieved, GrantQuest employs a Large Language Model (LLM) server to generate query-focused snippets. The LLM utilizes an abstractive summarization approach to create concise, coherent summaries tailored to the user's specific query. This method goes beyond traditional extractive techniques by generating new sentences that distill the most pertinent information, making the summaries more natural and easier to read.

Initially, the snippets generated by GrantQuest were composed of three components:

- A summary of the specific area or thing the grant will fund, that is, the purpose of the grant.
- A judgement on how well the grant matches the query.
- An explanation on how the grant can support the query/interest, if it does.

In the final iteration of the system, the last two components were combined into a single coherent judgment, providing users with a more streamlined and easily digestible summary.

2.4. The Application Interface - Flask

The GrantQuest application is built using Flask, a lightweight and flexible web framework for Python. Flask serves as the backbone of the user interface, handling user requests, delivering search results, and displaying generated snippets. The application seamlessly integrates with both the Elasticsearch backend and the LLM server, orchestrating the flow of data to provide a cohesive user experience.

By leveraging Flask's modularity, the GrantQuest application not only delivers a powerful search and summarization tool but also ensures that the user experience is intuitive, efficient, and adaptable to future enhancements.

3. Navigating to the Solution

This section outlines the iterative process that led to the development of our final solution, highlighting the challenges encountered and the improvements made at each stage.

3.1. Initial Solution - Full-Text Search

Our initial approach utilized full-text search within the Elasticsearch index. This method, while straightforward and fast, quickly proved insufficient for the task of query-focused summarization. Full-text search relies heavily on keyword matching, which fails to capture the deeper semantic meaning of both the query and the document. As a result, the search results were often irrelevant.

3.2. Transition to Semantic Search

Recognizing the limitations of full-text search, we implemented a semantic search approach in the next iteration of our solution. This involved selecting important fields from each document and generating embeddings for them using OpenAI's *text-embedding-3-small* model. The embeddings were aggregated and normalized to create a single document representation, which was then indexed in Elasticsearch.

To perform a search, the user's query was similarly converted into an embedding, and approximate k-Nearest Neighbors (kNN) search was used to retrieve the most relevant documents based on cosine similarity. This transition significantly improved the relevance of the search results, as embeddings allowed the system to better capture the semantic meaning of both the query and the documents.

3.3. Further Refinements - Re-ranking

Despite the improvements brought by semantic search, the system sometimes returned irrelevant candidates, particularly when dealing with large document collections. To address this, we integrated re-ranking models into the pipeline, enhancing the final results' relevance. -

- **Cross-Encoder based**: Embedding-based search models are typically bi-encoder models, where each document is embedded and stored, then queries are also embedded and retrieval is based on the similarity of the query's embedding to the documents' embeddings. In this model, many nuances of token-level interactions between users' queries and matched documents are lost because the original query and documents can never "see" each other – only their embeddings do. This may come at a price of retrieval accuracy – an area where cross-encoder re-ranker models excel.
  In a cross-encoder architecture, query-document pairs are encoded together to produce a relevance score instead of an embedding. The advantage of cross-encoders is the higher performance, as they perform attention across the query and the document. Scoring thousands of query, document pairs would be rather slow, so we use the retriever to retrieve a set of e.g. 100 possible candidates which are then re-ranked by the cross-encoder.

- ***ms-marco-MiniLM-L-12-v2***: This open-source [1] cross-encoder model was trained on the MS Marco Passage Ranking task. The query and a possible document is passed simultaneously to transformer network, which then outputs a single score between 0 and 1 indicating how relevant the document is for the given query. We can then sort the passages in descending order. It has a context length of 512 tokens.
- ***jina-reranker-v2-base-multilingual***: Another Cross-Encoder that has been fine-tuned for text re-ranking task, Jina Reranker v2 is a SOTA re-ranker released on Jun 25th 2024. The jina-reranker-v2-base-multilingual model is capable of handling long texts with a context length of up to 1024 tokens, enabling the processing of extensive inputs. To enable the model to handle long texts that exceed 1024 tokens, the model uses a sliding window approach to chunk the input text into smaller pieces and re-rank each chunk separately. It is a paid model accessible through the Jina API [2].

- **LLM based**: Sun et, al.[2] demonstrated the ability of LLM's for relevance ranking. Inspired by this, we considered two methods of reranking using LLMs.
  - ***Indivdual Scoring***: In this method, the LLM is prompted to give each document a score according to it's relevance to the query, during snippet generation. These scores can then be used to re-rank the documents.
  - ***RankGPT***: Sun et, al. [2] introduced a novel instructional permutation generation method with a sliding window strategy to directly output a ranked list from an LLM, given a set of candidate passages. Their code is available online [3].

These re-ranking strategies provided more accurate and contextually appropriate results, particularly when combined with the semantic search pipeline.

3.4. Final Iteration - Document Normalization

We found that the solution we had thus far encountered two issues -

- **Degradation of snippet quality**: The quality of the snippets generated by the LLM declined with increase in document token length. This is a common limitation of using LLMs with long documents due to the inherent difficulty in modeling long-range dependencies in text.
- **Additional costs**: Incorporating re-rankers added computational overhead and, in some cases, required access to paid models.

To deal with these issues, we developed a document normalization method -

- Use an LLM to generate detail-rich, structured summaries for all documents. These are called 'normalized documents' and added to the index, to be given to the LLM during snippet generation
- Generate embeddings for these normalized documents and add to the index.
- Conduct approximate kNN using these 'normalized embeddings'.

Through experiments, we concluded that using this approach resulted in two improvements-

- **Enhanced Snippet Quality**: The use of these information dense summaries to generate query-focused snippets, resulted in an improvement to the quality of snippets, regardless of the length of the document.
- **Boosting ElasticSearch**: The use of the normalized document embeddings during retrieval boosted the performance of ElasticSearch's retrieval, producing results comparable to those obtained with the advanced Jina Reranker v2.

Information

Normalization prompt:
System prompt: "You are a highly skilled summarizer specialized in grants. You are provided with detailed information about a grant and your task is to create a structured summary according to a specified format. The summary must be accurate and informative, following the given structure precisely."
User prompt: "Based on the provided grant information, generate a structured summary with the following fields:
1. Title: [The name or title of the grant.](Limit: 10 words)
2. Amount: [The funding details, including the minimum and maximum amounts, if specified.] (Limit: 40 words)
3. Deadline: [The submission deadline(s) for the grant application.] (Limit: 15 words)
4. Description: [A very detailed overview of the grant's purpose and objectives. This should be comprehensive and informative.] (Limit: 200 words)
5. Eligibility: [The detailed criteria for applicants to be eligible for the grant, including any specific requirements.](Limit: 50 words)
6. Sponsor: [The organization or entity sponsoring the grant.] (Limit: 20 words)
7. Categories: [The areas or fields the grant supports.] (Limit: 20 words)
8. Activity: [The EXACT activity/activities the grant funds. This should be comprehensive and accurate] (Limit: 70 words)
Ensure that the description and eligibility sections are detailed and comprehensive. Reply in JSON format while following word limits."

4. Experiments

This section presents the experiments conducted to evaluate the performance of different models and methods for snippet generation and document ranking. The experiments were designed to validate the effectiveness of the proposed GrantQuest system, particularly the impact of document normalization and semantic search on snippet quality and retrieval accuracy.

4.1. Snippet Evaluation

We conducted experiments to evaluate the performance of various language models (LLMs) in generating query-focused snippets. The models were evaluated based on three key aspects of the snippets: summary quality, judgment accuracy, and support clarity. Each aspect was scored on a scale from 1 to 5, where 1 indicates incorrect and incomplete content, and 5 indicates content that is correct, informative, and concise.

Table 1 presents the average scores for each model. As expected, larger and more capable models with longer context lengths, such as *GPT-4o*, performed better across all three metrics. The results highlight a clear trend where models with greater capacity and context length produce higher quality summaries, judgments, and support information. This is particularly evident when comparing *GPT-4o* to smaller models like *Meta-Llama3-8B-Instruct*, which struggled with maintaining accuracy and informativeness, particularly in the support score.

4.2. Document Length Impact

To further understand the limitations of these models, we analyzed how snippet quality varied with the length of the grant documents. Figure 2 illustrates the negative trend in snippet scores as the token

[1] https://huggingface.co/cross-encoder/ms-marco-MiniLM-L-12-v2
[2] https://jina.ai/reranker/
[3] https://github.com/sunnweiwei/RankGPT

Table 1. Comparison of models in snippet generation.

LLM	summary score	judgement score	support score
gpt-4o	4.848485	4.757576	4.909091
gpt-4o-mini	4.818182	4.303030	4.060606
gpt-3.5-turbo	4.606061	3.636364	3.787879
meta-llama3-8B-Instruct	4.000000	3.303030	2.969697

length of documents increases. The decline in snippet quality with increasing document length highlights a significant challenge for LLMs, particularly those with shorter context lengths. This issue was most pronounced in models like Meta-Llama3-8B-Instruct, underscoring the need for strategies like document normalization to manage large documents effectively. Our document normalization method addressed this issue by ensuring that no document exceeded the model's context length, leading to more consistent snippet quality across different document lengths.

4.3. Ranking Evaluation

To evaluate the effectiveness of different ranking schemes, we compared the performance of several ranking methods using three key metrics:

- **Spearman's Rank Correlation**: This metric measures the degree of monotonic relationship between the rankings of documents by different rankers.

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{1}$$

- **Kendall Rank Correlation**: This metric reflects the ordinal association (i.e., the agreement or disagreement in the order of rankings) between the rerankers.

$$\tau = \frac{(\text{\# of concordant pairs}) - (\text{\# of discordant pairs})}{(\text{\# of pairs})} \tag{2}$$

- **Rank Biased Overlap**: Webber et. al [1] introduced this metric which reflects the ordinal association (i.e., the agreement or disagreement in the order of rankings) between the rankers. Unlike the other measures (Kendall's Tau or Spearman's Rank Correlation), RBO is designed to handle incomplete lists and to focus more on the top ranks, which are often the most important in many applications (like search engine results). We use an implementation of RBO available online [4].

Tables 2, 3, 4 contain the matrices for these metrics. es_rank refers to the ranking produced by ElasticSearch after using 'normalized document embeddings'. llm_rank refers to the ranking produced by sorting the documents according to scores provided by the Indivdual Scoring method.

The results from these metrics show that the use of normalized document embeddings in Elasticsearch significantly improved retrieval performance, achieving results comparable to the Jina Reranker v2, which is considered a state-of-the-art model. The Spearman and Kendall correlations indicate strong agreement between Elasticsearch and Jina rankings, while the RBO metric highlights that these models particularly agree on the top-ranked documents, which are crucial in search applications. The strong performance of normalized embeddings in Elasticsearch suggests that this approach can serve as a viable alternative to more resource-intensive models like Jina Reranker v2. The other methods perform very poorly compared to the benchmark set by Jina.

[4]https://github.com/changyaochen/rbo

5. Conclusion

The development of GrantQuest, a semantic search engine tailored for query-focused document summarization, marks a significant advancement in the realm of grant search and retrieval systems. Through a series of carefully designed iterations, we have addressed the key challenges associated with generating accurate and contextually relevant summaries from extensive and complex documents.

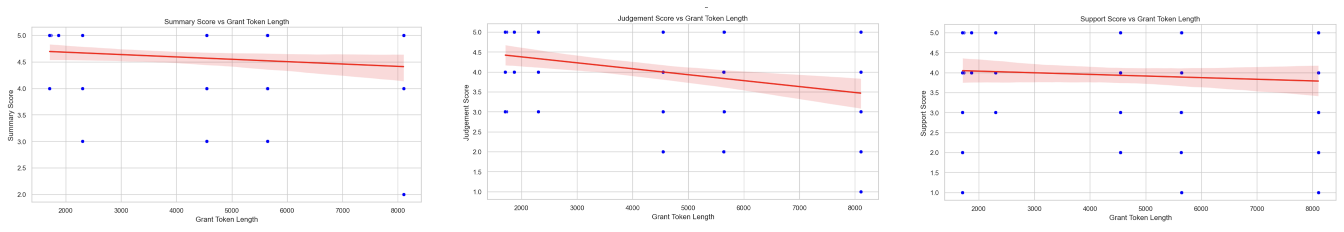
Our journey began with the limitations of full-text search, which, although straightforward, proved inadequate for capturing the nuanced relationships between queries and documents. By transitioning to semantic search and incorporating advanced techniques like document normalization and re-ranking, we significantly improved both the relevance of retrieved documents and the quality of generated snippets.

The experimental results confirmed that our approach, particularly the use of normalized document embeddings within Elasticsearch, offers a compelling balance between computational efficiency and retrieval accuracy. This method not only produced results on par with state-of-the-art models like Jina Reranker v2 but also ensured that users receive high-quality, query-focused summaries tailored to their specific needs.

In summary, GrantQuest represents a robust and scalable solution for enhancing the user experience in grant searching, providing a foundation that can be further built upon as new technologies and methodologies emerge in the field of natural language processing and information retrieval.

6. Assessment

The primary objective for the semester was to develop and refine a query-focused document summarization framework aimed at enhancing the search and retrieval experience within the context of grant documentation. The GrantQuest system, which integrates Elasticsearch and LLMs, has made significant strides towards this goal,



(a) Plot of Summary Score vs Grant Token Length **(b)** Plot of Judgement Score vs Grant Token Length **(c)** Plot of Support Score vs Grant Token Length

Figure 2. Negative trend of snippet scores with grant token length

Table 2. Spearman's Correlation Matrix

	es_rank	**jina_rank**	**minilm_rank**	**rankgpt_rank**	**llm_rank**
es_rank	1.000000	1.000000	0.109673	0.142213	0.131604
jina_rank	1.000000	1.000000	0.109673	0.142213	0.131604
minilm_rank	0.109673	0.109673	1.000000	0.243285	0.238131
rankgpt_rank	0.142213	0.142213	0.243285	1.000000	0.279401
llm_rank	0.131604	0.131604	0.238131	0.279401	1.000000

Table 3. Kendall Correlation Matrix

	es_rank	**jina_rank**	**minilm_rank**	**rankgpt_rank**	**llm_rank**
es_rank	1.000000	1.000000	0.078104	0.103993	0.090885
jina_rank	1.000000	1.000000	0.078104	0.103993	0.090885
minilm_rank	0.078104	0.078104	1.000000	0.174132	0.168030
rankgpt_rank	0.103993	0.103993	0.174132	1.000000	0.203017
llm_rank	0.090885	0.090885	0.168030	0.203017	1.000000

delivering more accurate and contextually relevant query-specific summaries.

Key achievements include:

- **Implementation of Semantic Search**: Transitioning from traditional full-text search to semantic search using embeddings significantly improved the relevance of search results.
- **Development of Document Normalization**: The creation of a method to generate structured summaries for grant documents, which has proven effective in improving both snippet quality and retrieval accuracy.
- **Integration of Re-ranking Mechanisms**: Experimentation with various re-ranking models has allowed for a deeper understanding of how different models can impact retrieval performance, although the final implementation did not rely heavily on them due to the effectiveness of the document normalization strategy.

Overall, the system has achieved the desired goal of providing users with concise, informative, and relevant snippets that are well-aligned with their queries, in the grant search domain.

7. Reflection

This project has been a profound learning experience, significantly enhancing both my technical skills and my approach to problem-solving in complex systems.

- **Technical Knowledge**: I gained a deeper understanding of advanced natural language processing techniques, particularly in the areas of semantic search and abstractive summarization using LLMs. The hands-on experience with these models provided invaluable insights into their capabilities and limitations, sharpening my ability to apply these techniques in practical scenarios.
- **System Integration**: The challenge of integrating Elasticsearch with LLMs and developing a cohesive web application using Flask greatly enhanced my skills in building scalable and efficient systems. This experience has also reinforced the importance of modularity and flexibility in system design, enabling better handling of real-world data and dynamic user queries.
- **Research and Experimentation**: The iterative process of experimenting with different models and approaches to improve retrieval accuracy and snippet quality underscored the importance of rigorous testing and methodical evaluation in research. This approach not only led to a better understanding of the models but also fostered a mindset of continuous improvement.

The guidance and feedback I received throughout this project were instrumental in navigating its complexities. The structured approach to problem-solving, combined with the freedom to explore innovative solutions, created a productive and enriching environment that greatly contributed to the project's success.

References

[1] W. Webber, A. Moffat, and J. Zobel, "A similarity measure for indefinite rankings", *ACM Trans. Inf. Syst.*, vol. 28, no. 4, 2010, ISSN: 1046-8188. DOI: 10.1145/1852102.1852106. [Online]. Available: https://doi.org/10.1145/1852102.1852106.

[2] W. Sun, L. Yan, X. Ma, *et al.*, *Is chatgpt good at search? investigating large language models as re-ranking agents*, 2023. arXiv: 2304.09542 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2304.09542.

Table 4. RBO Matrix

	es_rank	**jina_rank**	**minilm_rank**	**rankgpt_rank**	**llm_rank**
es_rank	0.998111	0.998111	0.524701	0.725662	0.621511
jina_rank	0.998111	0.998111	0.524701	0.725662	0.621511
minilm_rank	0.524701	0.524701	0.998875	0.539535	0.653796
rankgpt_rank	0.725662	0.725662	0.539535	0.996764	0.734665
llm_rank	0.621511	0.621511	0.653796	0.734665	0.998479