4DV504 - Selected Topics in Computer Science

S2 Work Plan

Student: Nicolas Mahn, nm222xc@student.lnu.se

1. Introduction

Research Questions:

- Can RAG-DB and LLMs be used to assess and compare the positions of different political parties based on publicly available data?
- Can such systems assist users/voters by providing clarity on the positions of different political parties and help them make informed voting decisions?

To answer these questions, a RAG-DB system must first be built. Since no Level 2 studies have yet explored the implications of RAG-DBs on the clarity of political parties, this analysis focuses on papers that discuss techniques for constructing RAG-DBs and achieving efficient prompt engineering instead.

The concept of RAG-DB was first introduced in a 2020 paper by Facebook (now Meta) (Lewis et al., 2020). In the paper Blended RAG by Sawarkar et al. (2024), the authors address a key challenge in RAG systems: the decline in response accuracy as the database size increases. Traditionally, RAG-DBs retrieve the most relevant documents by calculating the cosine similarity between the embeddings of all documents and the query (Dense Vector Index using KNN). However, the Blended RAG paper proposes two alternative retrieval methodologies to address these limitations.

The first approach utilizes TF-IDF (Term Frequency-Inverse Document Frequency) to prioritize documents where the most relevant terms are used most frequently (BM25 — Keyword-based Index). The second approach involves a Sparse Encoder, described as "an amalgam of semantic understanding and similarity-based retrieval" (Sawarkar et al., 2024, pg. 2, chap. 3). This Sparse Encoder combines the strengths of the previous two methodologies into a single framework (ELSER — Sparse Encoder Index) (Sawarkar et al., 2024, pg. 2, chap. 3).

The authors demonstrate that Sparse Encoders have significant advantages. Not only do they feature faster query times and require less storage space (Sawarkar et al., 2024, pg. 6, chap. 6.A), but they also achieve the highest retrieval accuracy across all tested benchmarks (Sawarkar et al., 2024, pg. 2f, chap. 4). Specifically, when using the Best Fields Query—where the retrieval process focuses on the most relevant fields of the documents (Sawarkar et al., 2024, pg. 2, chap. 3.A)—the Sparse Encoder consistently outperforms the Dense Encoder, which is the closest alternative. This superiority is evident across multiple tested benchmarks (Sawarkar et al., 2024, Fig. 2-6).

Another important segment of this project will be the extraction of relevant information from both the initial user query and from the manifestos of political parties. A paper by Polak and Morgan (2024) introduces a method called "ChatExtract," which aims to extract relevant data from research papers. The authors implement a series of prompts using multi-step prompt engineering to improve the accuracy of data extraction. This paper will be particularly useful for developing multi-step instructions in this project. Specifically, their flowcharts provide a useful template for our workflow (Polak and Morgan, 2024, Figures 1-3, especially Figure 2). The authors demonstrated very high performance in their approach, achieving 90.8% precision and 87.7% recall for extracting bulk modulus values (Polak and Morgan, 2024, pg. 9). Additionally, they effectively minimized hallucinations by incorporating redundancy through follow-up prompts (Polak and Morgan, 2024, pg. 3).

2. Objective and Scope

The project has two primary objectives:
Building the RAG-DB System: The main objective is to construct a RAG-DB containing the manifestos (and possibly additional information) of different political parties. This will involve creating a database for each major German political party (e.g., CDU/CSU, AfD, SPD, Die Grünen, FDP, BSW, and Linke). Given the limited time frame, an initial prototype may focus on just the three largest parties. The system will facilitate user queries by breaking down questions and querying the RAG-DBs of each party to provide comparative responses.

Evaluation of System Utility: The second objective is to assess how effectively the RAG-DB system helps users understand political party positions. This will be done via a survey of users who interact with the tool.

Scope:

- Project Architecture: The scope will include setting up the project architecture and ensuring a CI/CD pipeline is implemented to manage continuous integration and deployment effectively.
- RAG-DB System: This includes defining the architecture and scope of databases, collecting and structuring political party manifesto data, and developing the RAG-DB to ensure proper functionality.
- Multi-Step Prompt Engineering: Develop detailed flowcharts to handle queries effectively, implement these prompts, and iteratively test until a consistent response is achieved.
- Frontend Development: Create a simple, user-friendly interface that provides multilingual support (German/English) to make the tool accessible for users.
- Deployment: Deploy the project using a suitable cloud service, ensuring it is easily accessible for evaluation and user testing.
- Survey and User Testing: Design and conduct a survey to collect user feedback on the effectiveness of the tool, evaluate the gathered responses, and use this data to iterate and improve the project.
- Final Report: Compile a final report that summarizes findings, challenges, and evaluation outcomes based on both the technical work and user feedback.

3. Planned Activities and Research Methods

The planned activities are divided into several phases to build and evaluate the tool:

1. **Setup and Development**
   - RAG-DB System: Set up the cloud infrastructure, collect political party manifestos, and build the database. Focus on efficient data collection and database design using sparse encoder indexing.
   - Multi-Step Prompt Workflow: Design and implement multi-step prompts to extract information effectively.
   - Frontend Development: Create a simple user interface that supports German and English languages.
2. **Evaluation:** Develop and distribute a survey to evaluate the system's effectiveness in helping users understand political party positions.

4. Expected Outcomes

The expected outcomes for this project are two-fold:

1. **Technical Outcomes:** The successful development of a prototype RAG-DB system that can answer user questions about political parties' positions by providing detailed, consistent, and clear responses. Each major German political party will have a dedicated database, and the system will facilitate direct comparisons between them.
2. **User Insights:** Feedback on the tool's effectiveness from survey participants. The primary insight expected is whether the system helps users gain a clearer understanding of the political positions of different parties.
3. **Report of Findings:** A final report documenting the process, findings, and an evaluation of whether RAG-DB can effectively assist voters in making informed decisions.

5. Potential Threats and Risks

- **Complexity of Research Methods:** Developing and testing Sparse Encoders for RAG-DB is complex and may take longer than anticipated. To mitigate this, I will adopt a modular development approach, focusing first on a minimal viable product (MVP).
- **Data Availability:** Access to up-to-date and comprehensive manifestos may be challenging. As a mitigation strategy, I will initially work with a limited subset of documents or simply use older ones.

6. Milestones

In this project, a strict timeline for milestones will not be defined, as exact completion dates cannot be foreseen due to the iterative nature of development and potential unexpected challenges. Instead, Kanban methodology will be used for flexible task management, allowing for continuous progress tracking and adjustment as needed.

Milestones / Epics:

- Project Architecture
  - Build a System Architecture
  - Set up CI/CD pipeline to ensure the continuous integration of project updates and ease deployment.
- RAG-DB System
  - Define architecture and scope of databases.
  - Collect and structure data (party manifestos).
  - Develop the RAG-DB and ensure proper functioning.
- Multi-Step Prompt Engineering
  - Develop detailed flowcharts for query handling.
  - Test and iterate the multi-step prompt process until consistent responses are achieved.
  - Ensure multilingual support (German/English).
- Frontend Development
  - Implement basic user interface.
- Survey and Evaluation
  - Build survey for user evaluation.
  - Deploy survey and gather feedback.
  - Analyze and document findings.
- Final Report
  - Compile findings, challenges, and evaluation outcomes.
  - Finalize and submit the report.

References

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., and Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.-F., and Lin, H.-T., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Polak, M. P. and Morgan, D. (2024). Extracting accurate materials data from research papers with conversational language models and prompt engineering. *Nature Communications*, 15:1569.

Sawarkar, K., Mangal, A., and Solanki, S. R. (2024). Blended rag: Improving retrieval-augmented generation (rag) accuracy with semantic search and hybrid query-based retrievers. *arXiv preprint arXiv:2404.07220*.