Introduction

This paper examines the feasibility of automating knowledge acquisition from unstructured text through document-based question answering (DocQA). Information is embedded in formats such as news articles, research papers, and emails, leading to time-consuming and inefficient manual efforts (reading and checking documents to retrieve relevant knowledge). In this paper, I investigated whether it is possible to develop a system that can read documents and accurately respond to user questions in natural language.

Part 1: Literature Survey

The current state of the art in DocQA focuses on overcoming limitations in input length and ensuring that answers remain accurate and reliable by combining retrieval and generation techniques. Consider the following key models:

- BERT is transformer-based model for extractive QA but are limited to 512 token input size.
- T5 is a text-to-text transformer model for both extractive and generative QA but also limited by input length.
- Retrieval-Augmented Generation (RAG) combines a dense retriever (FAISS) with a generative model (BART). The retriever selects relevant passages based on the input question, and the generator synthesize the final answer based on the retrieved content.
- LangChain is a high-level model that offers composable components for document loading, text chunking, vector database integration (FAISS, Chroma), and connecting to large language models (GPT-4 via OpenAI).

As shown above, the field of DocQA has progressed from early transformer models effective for short inputs to more scalable retrieval-augmented systems. While models like BERT and T5 introduced core techniques for extractive QA, they are constrained by input length limitations. RAG address this limitation by separating retrieval and generation, allowing the model to access relevant context from long documents. Building on this model, LangChain further developed the system by integrating document parsing, dense retrieval, and large language models into end-to-end pipelines. These developments allow to build DocQA systems that can reason over long documents and answer user questions with contexts. However, challenges remain in accurately retrieving semantically aligned passages and evaluating the reliability of generative responses.

Part 2: Solution Design & Implementation

Based on Part 1, I identified Retrieval-Augmented Generation (RAG) as the best available solution for building a practical DocQA system. To implement this, I developed a simple prototype using Python, the LangChain framework, and the OpenAI GPT-4 API.

The system follows a six-step RAG pipeline. First, the system begins by extracting text from a PDF file. The loader converts each page of the document into a text object suitable for processing. Then, to preserve the context while handling length constraint, the document is split into overlapping 1000 character chunks with 200-character overlap. Next, each chunk is embedded into a dense vector and the resulting vectors are stored in a FAISS index for similarity-based retrieval. After embedding and indexing, the system combines the retriever with GPT-4 to create a chain capable of answering user question based on the most relevant context. Finally, when a user submits a question, the system retrieves relevant chnks from the index and generates an answer based on the document content using GPT-4.

Part 3: Implementation

The prototype demonstrates that an RAG pipeline can effectively answer questions from unstructured long-form documents. For example, when asked “Who is eligible for the NBA draft?”, the system correctly summarized the eligibility criteria outlined in the NBA Draft Tip Sheet. This reveals that it can extract structured information from dense text. However, limitations were observable during testing. When rephrasing the same question to “What are the requirements to enter the draft?”, the system returned a vague or partial response, indicating that the embedding-based retriever sometimes fails to retrieve the most relevant chunk when key terms or phrasing differ from the original text. In terms of performance, each query required a few seconds to process, which is manageable for small-scale use but may pose latency issues when scaled to large document collections. Overall, while the system is effective in controlled settings, its practical use would require improvements in retrieving system, answer traceability, and scalability.

Part 4: Proposed Improvements

To overcome the observed limitations and enhance system performance, several approaches can be considered. First, retrieval precision can be improved by integrating a neural re-ranking mechanism. This will allow the system to re-order the top-k retrieved chunks based on contextual relevance before they are passed to the language model. To improve interpretability and user trust, the system should return source information, enabling traceability. These enhancements together would transform the current prototype into a more accurate and interpretable DocQA system.