Development and Evaluation of an AI Assistant for Human-Machine Interaction: A Proof of Concept Using Retrieval-Augmented Generation and Few-Shot Learning for Industrial Automation

Hadi Ibrahim
Politecnico di Torino
s313385@studenti.polito.it

Naser Ibrahim
University of Vaasa
x7823417@student.uwasa.fi

***Abstract*—This report documents the research and development of a proof of concept for an AI assistant, commissioned by Prima Power, aimed at enhancing the interaction between operators and HMI (Human-Machine Interface) machines, specifically focusing on a laser machine termed "Laser Genius$^+$." The project was research-oriented, with the primary objective of exploring the potential of advanced language models to understand and respond to natural language queries, thereby laying the groundwork for future applications that could improve operational efficiency within industrial settings.**

The project involved several key phases: the extraction and preprocessing of text and image data from the Laser Genius$^+$ general and alarms manuals, followed by the creation of a custom dataset for training and evaluation purposes. The research focused on developing and testing different approaches to AI-driven machine interaction, including Retrieval-Augmented Generation (RAG) and few-shot prompt engineering. Initial experiments were conducted to evaluate the performance of these models using various metrics, providing insights into their effectiveness and limitations. The findings from this research will inform future developments, with the proof of concept serving as a foundational step towards the potential deployment of AI-assisted machine interfaces by Prima Power in their industrial operations.

I. Introduction

The rapid advancement of Artificial Intelligence (AI) has ushered in a new era of intelligent systems capable of performing complex tasks across various domains. In industrial environments, the integration of AI has the potential to significantly enhance operational efficiency by providing more intuitive interfaces for interacting with complex machinery. Specifically, AI-powered assistants that can interpret and respond to natural language queries present a promising avenue for simplifying human-machine interaction, reducing operational errors, and improving overall productivity.

This project, commissioned by Prima Power, focuses on developing a proof of concept for an AI assistant designed to aid operators in managing a sophisticated laser machine, referred to as *Laser Genius*$^+$. The core objective is to explore how advanced language models can be leveraged to create a system capable of understanding natural language queries, retrieving relevant information from machine manuals, and generating accurate and contextually appropriate responses. By doing so, this research aims to lay the groundwork for future AI-assisted human-machine interfaces that could be deployed across various industrial settings.

In recent years, large language models (LLMs) have become increasingly powerful, demonstrating an ability to generate human-like text and perform a wide range of tasks, from question answering to creative writing. However, the utility of these models in specialized domains, such as industrial automation, requires careful adaptation and enhancement. Two key approaches that have shown promise in this regard are Retrieval-Augmented Generation (RAG) and few-shot learning.

Retrieval-Augmented Generation (RAG) enhances the capabilities of LLMs by integrating them with retrieval mechanisms that allow the model to access and incorporate external knowledge, such as technical manuals, into its responses. This approach is particularly valuable in industrial settings where accuracy and context are critical. RAG not only ensures that the generated text is grounded in factual information but also allows the system to handle a wider range of queries by leveraging a large corpus of domain-specific documents.

Few-shot learning, on the other hand, enables LLMs to generalize from a limited number of examples, making it possible to train models effectively even when large datasets are not available. This technique is especially useful in scenarios where the AI assistant must adapt to new tasks or understand specific jargon without extensive retraining.

In this project, both RAG and few-shot learning were employed to explore their potential in enhancing human-machine interaction within the context of the *Laser Genius*$^+$ machine. By combining these approaches, the research aims to develop a robust proof of concept that can serve as the foundation for future AI-driven solutions in industrial automation.

The subsequent sections of this report will provide a detailed account of the research methodology, experimental setup, and results obtained from implementing these techniques. The related works section will delve into the existing literature on RAG, few-shot learning, and other relevant AI technologies, highlighting how they have informed and influenced the approaches taken in this project.

II. Related Works

The development of AI systems for complex industrial machinery, such as the Laser Genius$^+$ machine, is grounded in advancements in large language models (LLMs) and natural language processing (NLP) techniques. This project particularly leverages Retrieval-Augmented Generation (RAG) and prompt engineering, including few-shot learning. This section reviews key research influencing these methodologies.

A. Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) enhances the capabilities of large language models by integrating external information retrieval into the generative process. By accessing relevant documents from a knowledge base, RAG ensures that the model's responses are grounded in accurate, contextually appropriate information, making it particularly effective for knowledge-intensive tasks.

Gao et al. [1] provide a comprehensive survey of RAG, highlighting advanced techniques such as retrieval-guided generation and retrieval-enhanced generation, which ensure that retrieved documents actively guide the generative process, improving coherence and relevance. The original RAG model by Lewis et al. [2] demonstrated significant improvements in tasks like open-domain question answering by conditioning generated responses on retrieved passages. Additionally, Karpukhin et al. [3] and Izacard et al. [4] refined these methods with dense retrieval and effective document ranking, which are crucial for ensuring that only the most relevant information informs the generation process.

In this project, RAG was employed to navigate the extensive technical documentation of the Laser Genius$^+$ machine, enabling the AI assistant to deliver precise and informed responses to operator queries.

B. Prompt Engineering and Few-Shot Learning

Prompt engineering is a vital technique for fine-tuning LLMs without modifying their underlying parameters. This involves designing prompts that guide the model's behavior, allowing it to perform tasks it was not explicitly trained for. Sahoo et al. [5] provide a structured overview of various prompt engineering techniques, including few-shot prompting, which is particularly valuable in scenarios with limited annotated data.

Few-shot learning, as demonstrated by Brown et al. [6], allows models to generalize from a small number of examples, making it essential in industrial applications where large datasets are often unavailable. In this project, few-shot learning was crucial for enabling the AI assistant to adapt to the specific jargon and operational requirements of the Laser Genius$^+$ machine, ensuring accurate responses even with limited training data.

Recent innovations, such as automatic chain-of-thought (Auto-CoT) prompting [5] and retrieval-augmented CoT [7], further enhance the reasoning capabilities of LLMs, enabling them to handle complex queries more effectively. These techniques were incorporated into the AI assistant's design to improve its performance in managing the machine's operations.

C. Image Captioning

Other research has also informed the development of the AI assistant in this project. Bommasani et al. [8] discuss the capabilities and limitations of foundation models, emphasizing the importance of adapting these models to specific domains. This is particularly relevant for industrial applications, where general LLM capabilities must be tailored to meet domain-specific needs.

Moreover, Radford et al. [9] explore multimodal learning, which integrates textual and visual information. This was crucial for the preprocessing steps of this project, such as image captioning and document segmentation, allowing the AI assistant to provide more accurate and contextually rich responses.

This project leverages recent advancements in RAG and prompt engineering to develop an AI assistant capable of handling complex, domain-specific queries in industrial settings. By incorporating these cutting-edge techniques, the project demonstrates the potential for AI to significantly improve human-machine interaction in industrial environments.

III. Methodology

The methodology for developing an AI assistant capable of understanding and responding to natural language queries in the context of the Laser Genius$^+$ machine is structured into three main phases: Text and Image Preprocessing, Dataset Creation, and Development of AI Models. Each phase is crucial for the overall success of the project and is detailed below.

A. Text and Image Preprocessing

This phase involved preparing the raw data extracted from the Laser Genius$^+$ manual and alarm dataset for subsequent stages. The key tasks in this phase included data extraction, image captioning, text preprocessing, and document splitting.

1) Data Extraction

The first step in preprocessing was extracting text and images from the PDF manuals associated with the Laser Genius$^+$ machine. These manuals included both general operational guidelines and specific information about alarms and icons used in the software interface. The extracted data served as the foundational content for building the AI assistant.

2) Image Captioning

To make the images extracted from the manuals more useful, captions were generated using the GPT-4O Vision API. The image captioning process involved describing the content of each image in detail, which helped in providing context for the images when used later in the dataset and AI model training.

3) Text Preprocessing

The extracted text underwent several preprocessing steps to clean and normalize the data:

- **Removal of Special Characters:** All unnecessary special characters were removed from the text to avoid processing errors.
- **Whitespace Normalization:** Extra whitespaces were eliminated to ensure consistent formatting throughout the text.
- **Cleaning Irrelevant Words:** Words and phrases that were not relevant to the operation of the Laser Genius$^+$ machine were removed to focus the content on pertinent information.

4) Document Splitting

To ensure the text was manageable and relevant for AI model training, documents were split into smaller, semantically coherent chunks. This was achieved using LangChain's Semantic Chunker, which employed the OpenAI `text-embedding-3-large` model. This approach allowed for more meaningful segmentation of the text based on semantic content rather than arbitrary length, enhancing the relevance of each chunk for subsequent processing.

B. Dataset Creation

The second phase focused on creating a comprehensive QA dataset that would be used to train the AI models. Two different methods were employed to generate QA pairs, ensuring both breadth and depth in the dataset.

1) GPT-4O-Based QA Generation

The first method involved using GPT-4O to generate question-answer pairs from the semantically split text chunks. The model was prompted to create relevant, high-quality QA pairs, including a category for each question, a context summary, and lists of correct and incorrect answers. This method resulted in the creation of 400 QA pairs for the general manual and 400 for the alarms manual.

2) RAGAS-Based QA Generation

The second method used the RAGAS package, which allowed for a more sophisticated QA generation process by introducing various types of questions such as simple, multi-context, and reasoning-based queries. This method used GPT-3.5-turbo-16k as the generator and GPT-4O as the critic. The dataset generated using this method included 100 QA pairs for the general manual and 100 for the alarms manual, providing additional depth to the training set.

C. Development of AI Models

The final phase involved developing and evaluating AI models using the generated datasets. This phase explored various approaches, including Retrieval-Augmented Generation (RAG), few-shot learning, and a combined RAG and few-shot learning approach where all utilize the llama3.1 70B from Groq Cloud.

1) Retrieval-Augmented Generation (RAG)

The RAG model was implemented to leverage the extensive documentation available for the Laser Genius$^+$ machine. This approach combined information retrieval with generative modeling to provide accurate and contextually relevant answers to operator queries. The following enhancements were applied:

- **Advanced Query Refinement:** Queries were refined using LLaMA 3.1 70B, improving the relevance of the retrieved documents.
- **Retriever Methods:** A variety of retrieval methods were utilized to ensure the accuracy and comprehensiveness of the retrieved information.
- **Embedding Models:** Diverse embedding models were experimented with to optimize the retrieval process.

The prompt used for the RAG model was:

```
Using the information contained in the
   context, give a comprehensive answer
   to the question. Respond only to the
   question asked; the response should be
    concise and relevant to the question.
    Given the following context: \{
   context\}, answer the question: \{
   question\}. If the answer cannot be
   deduced from the context, use your own
    knowledge.
```

2) Few-Shot Learning

Few-shot learning was utilized to allow the model to generalize effectively from a small set of examples. This approach was particularly advantageous in adapting the model to specific queries related to the Laser Genius$^+$ machine.

The prompt used for few-shot learning was:

```
You are a helpful assistant that answers
   mostly questions about Tulus Manual
   software from the company Prima Power.
    Answer the question below. But first,
    to have a better understanding, here
   you can find some examples: \{examples
   \} Answer the following question: \{
   Question\}:
```

3) Combined RAG and Few-Shot Learning

The combined approach leveraged both RAG and few-shot learning to create a hybrid model that could handle a broad range of queries effectively.

The prompt used for the combined RAG and few-shot learning model was:

```
Answer the following question about Tulus
   Manual software from Prima Power.
```

```
Here are some example questions and
    answers to help you provide a better
    response: {examples}
Using the following context, give a
    comprehensive answer to the question.
Respond only to the question asked, and be
     concise and relevant.
Given the following context: {context},\n
    answer the question: {question}
If the answer cannot be deduced from the
    context, use your own knowledge.""",

}
```

The methodology employed in this project outlines a systematic approach to developing an AI assistant tailored to the needs of operating the Laser Genius^{+} machine. By carefully structuring the phases—text and image preprocessing, QA dataset creation, and the development of AI models—this project has successfully implemented advanced natural language processing techniques to enhance human-machine interaction in industrial settings.

IV. Experimental Setup

A. Architecture

The architecture of the system is designed to leverage both a Retrieval-Augmented Generation (RAG) model and a few-shot learning model, each implemented to perform specific tasks with the use of advanced retrieval mechanisms and Large Language Models (LLMs). The combined model leverages the strengths of both RAG and few-shot methods to enhance overall performance, specifically in the context of answering complex queries based on provided contexts.

RAG Model:

- The RAG model is built to perform retrieval and generation tasks by utilizing multiple retrievers and sophisticated reranking techniques.
- The model uses the LLaMA 3.1 70B, a state-of-the-art LLM hosted on the Groq Cloud, for generating refined queries and answers.
- **Retrievers:**
  - **Naive Retriever:** A baseline retriever that returns documents based on semantic similarity. It uses text embeddings to find the closest documents to the input query, but may not always retrieve the most contextually relevant information, especially in complex or highly specific queries.
  - **Advanced Retrievers:**
    - `LLMChainFilter`: This retriever uses a chain of large language models (LLMs) to filter out irrelevant documents, ensuring that only the most contextually relevant documents are retrieved. It adds an extra layer of processing to refine the initial retrieval results.
    - `EmbeddingsFilter`: Works directly with document embeddings to rank and filter the retrieved documents based on their semantic similarity to the query embeddings. It enhances relevance, especially for large datasets and complex documents, by ensuring better alignment of embeddings between the query and the documents.
  - **Ensemble Retriever:** This method combines the strengths of both sparse and dense retrieval approaches by integrating BM25 and FAISS.
    - `BM25`: A traditional sparse retrieval method that ranks documents based on term frequency and inverse document frequency (TF-IDF), providing precise keyword matching.
    - `FAISS`: A dense retrieval technique using vector embeddings to retrieve documents based on their semantic similarity to the query, particularly useful for large-scale retrieval tasks.
    - `Combination`: By combining BM25 for exact term matching and FAISS for semantic retrieval, the Ensemble Retriever enhances the overall accuracy and relevance of retrieved documents.
- **Vector Stores:** FAISS and Chroma are used to store document embeddings and retrieve the most relevant chunks.
- **Embedding Models:**
  - `"BAAI/bge-small-en-v1.5"`
  - `"BAAI/bge-large-en-v1.5"`
- **Temperature Settings:** The model's temperature hyperparameter is adjusted in various experiments to fine-tune the balance between exploration and exploitation.

Few-Shot Learning:

- The few-shot learning model is designed to perform well with limited training examples by providing the model with specific prompts and examples that closely resemble the task at hand.
- This model also utilizes LLaMA 3.1 70B to generate high-quality responses based on a few examples.
- Different model temperatures were experimented with to optimize the response generation.
- Various values for k, representing the number of examples provided, were tested to assess their impact on model performance.

Combined RAG and Few-Shot:

- This model integrates both RAG and few-shot capabilities, allowing it to utilize the retrieval strengths of RAG while leveraging the prompt-based adaptation of the few-shot model.

B. Datasets

Two main datasets were created for this project: one focused on the general manual and the other on the alarms manual of the Laser Genius^{+} machine. These datasets consist of question-answer (QA) pairs generated using GPT-4O and RAGAS, as detailed in the methodology.

In total, **500 QA pairs** were generated for the general manual and **500 QA pairs** for the alarms manual, resulting in **1000 QA pairs** overall. The datasets were split into training, validation, and test sets for evaluation purposes (see Table I).

Dataset	QA Pairs
General Manual	500
Alarms Manual	500
Total	**1000**

TABLE I
SUMMARY OF QA PAIRS GENERATED FOR EACH MANUAL.

C. Implementation Details

The implementation of this project was done using state-of-the-art natural language processing tools and cloud-based infrastructure, which are outlined below:

1) **Programming Languages and Packages**
   - **LangChain**: Used for implementing Retrieval-Augmented Generation (RAG) and managing the retrieval and generation processes.
   - **Hugging Face Transformers**: Pre-trained models such as "BAAI/bge-small-en-v1.5" and "BAAI/bge-large-en-v1.5" were used for generating document embeddings.
   - **OpenAI API**: Leveraged for generating question-answer pairs and handling model interactions using GPT-4o and GPT-3.5-turbo-16k.
   - **FAISS and Chroma**: FAISS was employed as the vector store for managing and retrieving document embeddings efficiently.
   - **BM25**: Integrated as part of an ensemble retrieval system, combining sparse and dense retrieval methods for improved accuracy.
2) **Cloud Infrastructure and Models**
   The experiments were conducted using the Groq Cloud API, utilizing the LLaMA 3.1 70B model for computation-intensive tasks such as training and model evaluation. Groq Cloud provided the necessary infrastructure to handle the large-scale data processing and model training.
3) **Embedding Models**
   Various embedding models were used to optimize the retrieval process, including "BAAI/bge-small-en-v1.5" and "BAAI/bge-large-en-v1.5", which played a crucial role in generating high-quality document embeddings for the RAG models.
4) **Evaluation Metrics**
   Model performance was evaluated using a combination of traditional NLP metrics such as ROUGE and BERT Score, along with advanced metrics provided by the RAGAS package. These metrics ensured a thorough evaluation of both the retrieval and generation capabilities of the system.

D. Cloud Infrastructure and Models

All experiments were conducted using the Groq Cloud API, which provided access to the LLaMA 3.1 70B model. The use of Groq Cloud was instrumental in handling the computational demands of training and running the large-scale models involved in this project.

E. Evaluation Metrics

To assess the performance of the AI assistant, we employed a combination of retrieval and answer generation evaluation metrics. These metrics are divided into two main categories: *retrieval evaluation* and *answer generation evaluation*.

1) Retrieval Evaluation

The following metrics were used to evaluate the quality of the document retrieval process:

- **Recall:** Measures the proportion of relevant documents that are successfully retrieved out of all relevant documents available. Recall is defined as:

$$\text{Recall} = \frac{\text{Number of relevant documents retrieved}}{\text{Total number of relevant documents}} \tag{1}$$

  A higher recall indicates that the retriever is effective at capturing the majority of relevant documents from the knowledge base.
- **Mean Reciprocal Rank (MRR):** This metric evaluates the ranking quality of the first relevant document retrieved. It computes the reciprocal of the rank at which the first relevant document is found and averages it over all queries. MRR is given by the following formula:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{2}$$

  where $|Q|$ is the total number of queries, and rank_i is the rank position of the first relevant document for query i. Higher values indicate better ranking performance.

These retrieval metrics focus on how effectively the system retrieves relevant documents from the knowledge base. Recall measures how many relevant documents are retrieved, while MRR emphasizes how quickly the first relevant document is found.

2) Answer Generation Evaluation

To evaluate the quality of the answers generated by the system, we used the following metrics:

- **ROUGE:** The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric measures the overlap of n-grams between the generated answer and the reference (ground truth) answer. We specifically used:
  - **ROUGE-1:** Measures the overlap of unigrams between the generated and reference answers.
  - **ROUGE-2:** Measures the overlap of bigrams between the generated and reference answers.

- **ROUGE-L:** Measures the longest common subsequence (LCS) between the generated and reference answers.

The ROUGE score can be defined as:

$$\text{ROUGE-N} = \frac{\sum_{\text{reference}\cap\text{generated}} \text{N-grams}}{\sum_{\text{reference}} \text{N-grams}} \tag{3}$$

- **Cosine Similarity:** We replaced the BERT score with cosine similarity to evaluate the semantic similarity between the generated and reference answers. Cosine similarity measures the cosine of the angle between two vectors (embeddings) in a multi-dimensional space. Using the embeddings generated by the `BAAI/bge-large-en-v1.5` model, cosine similarity is calculated as follows:

$$\text{Cosine Similarity} = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \tag{4}$$

where $\mathbf{A}$ and $\mathbf{B}$ are the embeddings of the generated and reference answers, respectively. A cosine similarity closer to 1 indicates greater semantic similarity between the generated and reference answers.

These answer generation metrics ensure that the responses generated by the AI assistant are both syntactically and semantically similar to the expected answers, with ROUGE focusing on surface-level overlap and cosine similarity providing a deeper, semantic comparison.

V. Results

This section presents the performance of the RAG and few-shot learning models across multiple experiments. The results are evaluated using various metrics such as Cosine Similarity, ROUGE scores (1, 2, L, LSUM), Recall, and Mean Reciprocal Rank (MRR). The arguments for each experiment are detailed before the tables.

A. Few-shot Experiments

The few-shot experiments demonstrated an incremental improvement with higher k-values. As seen in Table II, Experiment 4 with $k = 6$ achieved the best performance in the few-shot experiments, with a Cosine Similarity of 0.782, Recall of 0.708, and an MRR of 0.600 for the Manual dataset. The ROUGE scores were also higher compared to other few-shot experiments, with ROUGE-1 at 0.381 and ROUGE-L at 0.333. Similar trends were observed for the Alarm dataset with $k = 6$, which achieved a Recall of 0.501 and an MRR of 0.421. However, the combined approach (Experiment 10) provided comparable performance and demonstrated that integrating both methods can yield strong results.

B. RAG Experiments

- **Experiment 1-5:** Refine Query = 0.0, Embedding Model = BGE-Small, k_fewshot = NaN, Retriever method = False
- **Experiment 6:** Refine Query = 0.0, Embedding Model = BGE-Large, k_fewshot = NaN, Retriever method = NaN
- **Experiment 7:** Refine Query = 0.0, Embedding Model = BGE-Large, k_fewshot = NaN, Retriever method = embeddings_filter
- **Experiment 8:** Refine Query = 0.0, Embedding Model = BGE-Large, k_fewshot = NaN, Retriever method = llm_chain_filter
- **Experiment 9-10:** Refine Query = 0.0, Embedding Model = BGE-Large, k_fewshot = (Exp. 9: NaN, Exp. 10: 6.0), Retriever method = ensemble methods

The performance of the RAG experiments improved substantially in the later stages due to the combination of the **BGE-Large embedding model** and the **Ensemble Retriever** method. As shown in Table III, the shift from **BGE-Small** to **BGE-Large** boosted the retrieval quality.

In **Experiment 9**, where we used the **best-performing setup** (BGE-Large, Ensemble Retriever, and optimal temperature), we observed the highest Cosine Similarity (0.867) and Recall (0.879) for the Manual dataset. Importantly, no query refinement was applied in this experiment, as it had not performed better than the baseline in earlier tests (Experiments 1-5). This setup, with the best temperature, embedding model, and retriever, delivered the highest scores across most metrics.

Experiment 10 tested the combined approach, leveraging the best-performing RAG model (from Experiment 9) and the best few-shot parameters ($k = 6$). This combination aimed to capitalize on the retrieval strengths of the **Ensemble Retriever** with the guidance provided by the few-shot examples. The results show that this combined approach performed almost as well as the best RAG model alone, with only slight differences in Recall and MRR (see Table III).

VI. Discussion

The experiments conducted in this study aimed to evaluate the performance of different retrieval and generative methods, specifically Retrieval-Augmented Generation (RAG) and few-shot learning, as well as their combination. The results demonstrated that the **Ensemble Retriever** in the RAG model significantly improved retrieval quality, which in turn enhanced the accuracy and relevance of the generated responses. Additionally, the few-shot learning approach, while not as powerful as the RAG model on its own, added value when combined with retrieval, as seen in the combined few-shot and RAG experiment (Experiment 10).

A. Impact of the Ensemble Retriever on RAG Performance

The **Ensemble Retriever**, which integrates both sparse and dense retrieval approaches, played a pivotal role in improving the retrieval quality. By combining **BM25** for exact term matching and **FAISS** for dense, semantic-based retrieval, the ensemble method captured both the surface-level lexical matches (using BM25) and deeper semantic relationships (using FAISS) between the query and documents. This dual approach enhanced the system's ability to retrieve highly relevant documents that were not only lexically aligned with

Experiment	Dataset	Temperature	k	Cosine Similarity	ROUGE Scores			
					ROUGE 1	ROUGE 2	ROUGE L	ROUGE LSUM
Exp 1	Manual	0	0	0.773	0.278	0.160	0.237	0.241
	Alarm			0.700	0.189	0.105	0.165	0.169
Exp 2	Manual	0	2	0.767	0.334	0.194	0.287	0.287
	Alarm			0.689	0.230	0.121	0.198	0.200
Exp 3	Manual	0	4	0.781	0.378	0.224	0.328	0.329
	Alarm			0.697	0.260	0.140	0.226	0.228
Exp 4	Manual	0	6	**0.782**	**0.381**	**0.231**	**0.333**	**0.334**
	Alarm			**0.709**	**0.285**	**0.158**	**0.249**	**0.252**
Exp 5	Manual	0.4	2	0.764	0.325	0.187	0.277	0.278
	Alarm			0.690	0.222	0.115	0.191	0.193
Exp 6	Manual	0.4	4	0.778	0.368	0.218	0.317	0.320
	Alarm			0.699	0.257	0.138	0.222	0.226
Exp 7	Manual	0.4	6	0.781	0.379	0.152	0.327	0.328
	Alarm			0.705	0.278	0.152	0.239	0.241
Exp 8	Manual	0.8	2	0.757	0.309	0.171	0.258	0.262
	Alarm			0.683	0.211	0.106	0.179	0.182
Exp 9	Manual	0.8	4	0.763	0.329	0.186	0.278	0.280
	Alarm			0.693	0.235	0.122	0.202	0.205
Exp 10	Manual	0.8	6	0.764	0.342	0.196	0.292	0.293
	Alarm			0.698	0.250	0.130	0.217	0.217

TABLE II
FEW-SHOT RUNS RESULTS FOR MANUAL AND ALARM DATASETS

Experiment	Dataset	Temperature	Top-k	Cosine Similarity	Recall	MRR	ROUGE Scores			
							ROUGE 1	ROUGE 2	ROUGE L	ROUGE LSUM
Exp 1	Manual	0.0	5	0.834	0.766	0.598	0.503	0.372	0.457	0.457
	Alarm			0.735	0.497	0.419	0.404	0.273	0.364	0.365
Exp 2	Manual	0.4	5	0.846	0.766	0.598	0.512	0.377	0.462	0.463
	Alarm			0.735	0.497	0.419	0.399	0.269	0.357	0.359
Exp 3	Manual	0.8	5	0.842	0.766	0.598	0.493	0.355	0.440	0.444
	Alarm			0.740	0.497	0.419	0.397	0.267	0.354	0.355
Exp 4	Manual	0.0	3	0.844	0.692	0.582	0.510	0.372	0.461	0.462
	Alarm			0.728	0.457	0.410	0.396	0.266	0.354	0.354
Exp 5	Manual	0.0	7	0.858	0.794	0.602	0.514	0.379	0.465	0.465
	Alarm			0.738	0.513	0.421	0.406	0.275	0.365	0.365
Exp 6	Manual	0.0	5	0.857	0.789	0.632	0.524	0.391	0.475	0.476
	Alarm			0.738	0.607	0.501	0.358	0.231	0.316	0.318
Exp 7	Manual	0.0	5	0.762	0.222	0.222	0.335	0.201	0.285	0.291
	Alarm			0.696	0.285	0.253	0.227	0.129	0.196	0.199
Exp 8	Manual	0.0	5	0.865	0.754	**0.661**	**0.543**	0.407	0.494	**0.497**
	Alarm			0.750	0.588	0.568	0.326	0.222	0.289	0.294
Exp 9	Manual	0.0	5	**0.867**	**0.879**	0.660	0.540	**0.412**	**0.495**	0.496
	Alarm			**0.790**	**0.898**	**0.753**	0.404	**0.281**	0.356	**0.362**
Exp 10	Manual	0.0	5	0.867	0.879	0.660	0.540	0.413	0.493	0.495
	Alarm			0.788	0.891	0.731	**0.406**	0.281	**0.357**	0.364

TABLE III
RAG RUNS RESULTS FOR MANUAL AND ALARM DATASETS

the query but also contextually appropriate based on their semantic content.

In addition to using the Ensemble Retriever, the **BGE-Large embedding model** also contributed significantly to the performance improvement. As observed in **Experiment 9**, the use of this embedding model enhanced retrieval quality by generating more accurate vector representations for both the query and documents, leading to better semantic matching. Moreover, **optimal temperature settings** were applied to fine-tune the balance between exploration and exploitation in the generation process.

Query refinement, tested in earlier experiments, did not perform better than the baseline. As a result, no refinement was used in the best-performing RAG experiments. This decision to rely solely on the raw query, combined with the powerful retrieval methods and large embeddings, contributed to the superior performance of the RAG model.

Furthermore, model performance was monitored through **LangSmith**, a tool designed for tracking the quality of responses generated by the AI assistant. LangSmith plays a key role in identifying areas for further improvement and will facilitate future work, where **Reinforcement Learning from Human Feedback (RLHF)** can be incorporated. By leveraging professional feedback, RLHF will enable continuous fine-tuning and adaptation of the model to industrial needs.

B. Few-shot Learning: Strengths and Limitations

Few-shot learning experiments showed a gradual improvement in performance as the number of examples (k) increased. This approach enabled the model to generalize from a small set of task-specific examples, which is valuable in scenarios where annotated training data is scarce. As seen in Table II, higher k values, such as in **Experiment 4 (k = 6)**, led to better Cosine Similarity, Recall, and ROUGE scores. This suggests that providing more examples helped the model adapt better to the task and generate more accurate responses.

However, few-shot learning on its own did not outperform the RAG model, primarily due to the fact that it lacks a robust retrieval mechanism. The few-shot model relies on the examples provided within the prompt, and while it can generalize from these examples, it does not have access to the broader knowledge base like the RAG model does. Therefore, its performance is limited when it comes to handling queries that require specific information retrieval from a large dataset.

C. Combined Few-shot and RAG Model

Experiment 10, which combined the strengths of both few-shot learning and RAG, performed nearly as well as the RAG model with the Ensemble Retriever (**Experiment 9**). This combined approach leveraged the retrieval strengths of RAG while benefiting from the generalization capabilities of few-shot learning. The few-shot component provided the model with examples that helped guide the generation process, ensuring that the responses adhered more closely to the style and structure of the examples.

The results from Experiment 10 show that while the combined approach did not surpass the RAG model with the Ensemble Retriever, it offered comparable performance. The few-shot examples provided additional guidance, which may have been particularly useful in cases where the retrieved documents were not perfectly aligned with the query. In such scenarios, the few-shot learning component helped fill the gaps by allowing the model to generalize from the examples and provide a more coherent response.

Again, **LangSmith** was used to monitor the system's performance during the combined approach, ensuring that future improvements, such as incorporating RLHF, can be implemented based on real-world usage and professional feedback.

D. Streamlit Deployment

A key practical aspect of this work is its deployment via **Streamlit**, which provides an accessible interface for operators in an industrial setting. By deploying the application on Streamlit, the AI assistant can deliver real-time responses to user queries in an intuitive manner. The deployment on Streamlit ensures that the solution is scalable and user-friendly, offering seamless interaction for professionals who manage the Laser Genius+ machine.

Streamlit's integration supports both the RAG-based retrieval system and the few-shot learning capabilities, making the assistant highly accessible while keeping the user interface simple. This deployment framework was chosen to ensure that the assistant could be effectively utilized in real-time, operational contexts with minimal training required for end users.

E. Overall Analysis

The results of this study highlight the critical role of retrieval in the RAG framework, particularly when combined with a strong retrieval method like the **Ensemble Retriever**. The ensemble method's ability to integrate both sparse and dense retrieval approaches proved highly effective in capturing both lexical and semantic matches, leading to superior retrieval and generation performance. This dual approach enhanced the overall relevance of the generated responses, making the RAG model with the Ensemble Retriever the best-performing model across both the Manual and Alarm datasets.

Few-shot learning, while valuable in certain contexts, was not sufficient on its own to outperform the RAG model. However, when combined with RAG, it provided additional flexibility and guidance to the model, particularly in cases where retrieval alone might not have been sufficient. This combined approach suggests that integrating few-shot learning into retrieval-based models can be an effective strategy for further improving performance, especially when retrieval does not provide perfect matches.

Future work could explore fine-tuning the **LLaMA 3.1 70B** model on a larger, more diverse dataset, which would likely improve its ability to generate contextually relevant responses across a wider range of queries. Fine-tuning the LLaMA model would enable it to better adapt to the domain-specific language and retrieval needs of industrial automation or other specialized areas. Combining this fine-tuned model with the best-performing RAG setup (BGE-Large and Ensemble Retriever) and few-shot learning configuration (using $k = 6$) would most likely yield even better results, given that the model would have both robust retrieval mechanisms and a more domain-tailored generation capability. This combination, along with continuous monitoring through LangSmith and future enhancements using RLHF, could enhance both the accuracy and relevance of generated answers in complex and diverse scenarios.

References

[1] Gao, Yunfan, Xiong, Yun, Gao, Xinyu, Jia, Kangxiang, Pan, Jinliu, Bi, Yuxi, Dai, Yi, Sun, Jiawei, Wang, Meng, & Wang, Haofen. ”Retrieval-Augmented Generation for Large Language Models: A Survey.” *arXiv preprint*, arXiv:2306.05284, 2023.

[2] Lewis, Patrick, Perez, Ethan, Piktus, Aleksandra, Petroni, Fabio, Karpukhin, Vladimir, Goyal, Naman, Küttler, Heinrich, Lewis, Mike, Yih, Wen-tau, Rocktäschel, Tim, et al. ”Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.” *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459–9474, 2020.

[3] Karpukhin, Vladimir, Oguz, Barlas, Min, Sewon, Lewis, Patrick, Wu, Ledell, Edunov, Sergey, Chen, Danqi, & Yih, Wen-tau. ”Dense Passage Retrieval for Open-Domain Question Answering.” *arXiv preprint*, arXiv:2004.04906, 2020.

[4] Izacard, Gautier, & Grave, Edouard. ”Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering.” *arXiv preprint*, arXiv:2007.01282, 2021.

[5] Sahoo, Pranab, Singh, Ayush Kumar, Saha, Sriparna, Jain, Vinija, Mondal, Samrat, & Chadha, Aman. ”A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications.” *arXiv preprint*, arXiv:2402.07927, 2024.

[6] Brown, Tom B., Mann, Benjamin, Ryder, Nick, Subbiah, Melanie, Kaplan, Jared, Dhariwal, Prafulla, Neelakantan, Arvind, Shyam, Pranav, Sastry, Girish, Askell, Amanda, et al. ”Language Models are Few-Shot Learners.” *arXiv preprint*, arXiv:2005.14165, 2020.

[7] Zhang, Zhuosheng, Zhang, Aston, Li, Mu, & Smola, Alex. ”Automatic Chain of Thought Prompting in Large Language Models.” *arXiv preprint*, arXiv:2210.03493, 2022.

[8] Bommasani, Rishi, Hudson, Drew A., Adeli, Ehsan, Altman, Russ, Arora, Simran, von Arx, Sydney, Bernstein, Michael S., Bohg, Jeannette, Bosselut, Antoine, Brunskill, Emma, et al. ”Opportunities and Risks of Foundation Models.” *arXiv preprint*, arXiv:2108.07258, 2021.

[9] Radford, Alec, Kim, Jong Wook, Hallacy, Chris, Ramesh, Aditya, Goh, Gabriel, Agarwal, Sandhini, Sastry, Girish, Askell, Amanda, Mishkin, Pamela, & Clark, Jack. ”Learning Transferable Visual Models From Natural Language Supervision.” *arXiv preprint*, arXiv:2103.00020, 2021.