**Final Project Report**

**Retrieval-Augmented Generation for Loan Risk Assessment Using Open-Source LLMs**

## Submitted By:

Dharani Anthireddy

Sai Kiran Nelluru

Course:

Natural Language Processing (DSCI-6004-02)

Professor:

Khaled Sayed

Date:

05/02/2025

**Abstract:**

This paper presents a Retrieval-Augmented Generation (RAG) system for intelligent loan risk assessment using open-source Large Language Models (LLMs). We transform the German Credit dataset into case narratives and use a vector database to support semantic retrieval. Three LLMs — FLAN-T5-Base, LaMini-Flan-T5-248M, and Falcon-RW-1B — are evaluated in a RAG pipeline that retrieves relevant examples and generates natural language responses to domain-specific loan risk questions. We analyze response accuracy, reasoning quality, factuality, and model efficiency. LaMini-Flan shows a strong balance of response clarity and speed, while Falcon-RW-1B demonstrates more contextual depth.

The system architecture includes case narrative construction, vector embedding via sentence-transformers, semantic retrieval using FAISS, and instruction-following LLM inference powered by LangChain. The models are assessed across ten financial decision-making questions. We further explore the performance trade-offs of different LLM architectures in terms of contextual integration, generation latency, and factual consistency. Our study offers practical insights for deploying interpretable and efficient NLP solutions in the financial domain and identifies the key strengths and limitations of modern open-source LLMs when applied to loan evaluation tasks.

**Keywords:** Retrieval-Augmented Generation, Loan Risk Assessment, Large Language Models (LLM), FAISS, Semantic Retrieval, Credit Scoring, Open-Source NLP, LangChain

**1.Introduction**

Large Language Models (LLMs) such as GPT-3 and PaLM have demonstrated impressive natural language understanding and generation capabilities. However, their factual grounding remains limited in specialized domains like finance. Retrieval-Augmented Generation (RAG) addresses this limitation by coupling LLMs with a retriever over an external knowledge base. In this paper, we apply the RAG framework to loan risk assessment by building a system that retrieves real-world financial case narratives and generates intelligent, context-aware answers to loan eligibility questions. We focus on comparing three open-source LLMs in terms of their ability to answer domain-specific queries using retrieved loan profiles.

Loan eligibility determination is a nuanced task that involves assessing various financial, personal, and demographic attributes. Traditional credit scoring models rely on handcrafted features and fixed rules, which often fail to adapt to complex or evolving applicant profiles. The emergence of LLMs introduces a new paradigm where decisions can be guided through contextual natural language understanding. However, these models, when used in isolation, may hallucinate or generalize from insufficiently grounded knowledge. RAG frameworks enhance their reliability by anchoring outputs in retrieved, semantically relevant documents.

Our motivation is to explore whether open-access LLMs, when augmented through RAG, can serve as lightweight, interpretable tools for credit analysis. We contribute a complete pipeline that includes data-to-text conversion, vector storage with FAISS, semantic retrieval, and prompt-tuned inference using three instruction-following LLMs. Our analysis sheds light on model-specific performance across multiple dimensions such as accuracy, factuality, and clarity, helping guide real-world deployment decisions.

**2. Related Work**

Prior work in the field of NLP has focused on instruction-tuned models like T5 (Raffel et al., 2020) and BERT-based retrieval techniques (Karpukhin et al., 2020). More recently, RAG-based systems have been deployed in areas such as open-domain QA (Lewis et al., 2020), summarization (Izacard and Grave, 2021), and medical search (Pal et al., 2023). In finance, language models have been applied to credit scoring (Huang et al., 2022) and fraud detection (Zhang et al., 2023), but RAG systems are less explored. Our work builds on these advancements by adapting RAG to tabular finance data and comparing multiple open-source LLMs.

**3. Methodology**

**3.1 Dataset and Preprocessing**

We use the German Credit dataset, which contains 1,000 loan applications with 20 categorical and numerical features, including credit history, employment status, loan amount, and personal demographics. Each record is converted into a textual case summary using handcrafted templates to enhance semantic richness. For example:

"Applicant is employed for more than 4 years, has a moderate savings account, and applied for a loan of 5000 marks with a 24-month duration. The credit history is satisfactory, and the customer has one dependent."

The final corpus consists of 1,000 such case narratives. These are then embedded using all-MiniLM-L6-v2 and stored in FAISS for retrieval.

**3.2 System Overview**

The system architecture is visualized in Figure 1 and highlights the core components of the pipeline, which include data preprocessing, embedding, retrieval, and generation. The modular design allows flexibility in swapping LLMs, prompt formats, or embedding models.

Reranker

(optional)

Retriever

User Query

Response

Vector Database

**Figure 1: Architecture Diagram** The flowchart illustrates the entire RAG process

1. **User Input**: The system begins with a user-submitted loan-related question.
2. **Retrieval**: FAISS (Facebook AI Similarity Search) identifies the most semantically similar loan case summaries.
3. **Prompt Construction**: These retrieved cases are combined with the user query using tailored templates.
4. **Model Inference**: The composed prompt is fed into one of the three selected LLMs—FLAN-T5-Base, LaMini-Flan-T5-248M, or Falcon-RW-1B.
5. **Response Generation**: The model outputs a response grounded in both retrieved context and the user's question.

The architecture of our Retrieval-Augmented Generation (RAG) system is modular and structured into five primary stages: case summary construction, embedding and indexing, semantic retrieval, prompt formatting, and model inference. This modularity supports flexibility in substituting models, tweaking prompts, or adjusting the retriever strategy. The flow of the system is as follows:

1. **Case Summary Construction**: Each record from the German Credit dataset is converted into a natural language narrative capturing credit amount, duration, employment, savings, and credit history details.
2. **Embedding and Indexing**: These narratives are transformed into dense vector representations using the SentenceTransformer model all-MiniLM-L6-v2. The vectors are stored in FAISS (Facebook AI Similarity Search), allowing for efficient nearest-neighbor retrieval based on cosine similarity.
3. **Semantic Retrieval**: Upon receiving a user query, the system retrieves top-k similar case narratives from the vector store. This helps ground the LLM's answer in relevant real-world loan contexts.
4. **Prompt Construction**: Retrieved contexts are appended to the user query using model-specific prompt templates. These templates are tuned to improve instruction-following performance across LLMs.
5. **LLM Inference**: The composed prompt is passed into one of the three LLMs—FLAN-T5-Base, LaMini-Flan-T5-248M, or Falcon-RW-1B—through LangChain's RetrievalQA pipeline. The model then generates a response informed by both the query and retrieved narratives.

The architecture facilitates transparency, customization, and efficient evaluation of different models over the same retrieval base.

**4. Domain-Specific Questions**

The following 10+ questions were developed for evaluation:

1. Does stable employment improve loan approval chances?
2. How does a poor credit history affect loan eligibility?
3. Is it risky to lend to a foreign worker?
4. How important is the duration at the current address?
5. Does having multiple guarantors reduce risk?
6. How does age impact loan decisions?
7. Do longer employment durations reduce default risk?
8. What factors increase loan default likelihood?
9. Are valuable assets required for approval?
10. Does the number of existing credits affect new loan eligibility?

**5.Technical Implementation of LLMs**

This section details how each of the three language models—FLAN-T5-Base, LaMini-Flan-T5-248M, and Falcon-RW-1B—was incorporated into the RAG system, focusing on their integration, configuration, and deployment in the LangChain framework.

* **FLAN-T5-Base**: This is a lightweight encoder-decoder model trained on a broad set of instruction-following tasks. It is particularly efficient in sequence-to-sequence tasks and is optimized for low-latency generation. The model was loaded using the HuggingFace text2text-generation pipeline and integrated into LangChain’s RetrievalQA chain. Prompt templates were crafted in a declarative instruction style to match its training distribution. During deployment, FLAN-T5-Base demonstrated strong speed performance and consistent outputs for straightforward eligibility questions.
* **LaMini-Flan-T5-248M**: This model is a distilled variant of the FLAN-T5 model, fine-tuned on instruction datasets for better instruction-following behavior in resource-constrained environments. It was deployed using the HuggingFace text2text-generation interface and exposed through LangChain as a HuggingFacePipeline-compatible LLM. Due to its compact architecture, LaMini-Flan was able to perform inference with a smaller memory footprint, making it suitable for rapid response applications. The prompt design reused the same instruction format as FLAN-T5-Base but required minor adjustments to prevent under generation.
* **Falcon-RW-1B**: Falcon is an autoregressive, decoder-only transformer model. Unlike the T5-based models, it follows a causal generation approach and is more sensitive to prompt structure. We utilized the text-generation pipeline and wrapped the model with LangChain’s HuggingFacePipeline. A few-shot prompting style was designed specifically for Falcon, including example-based reasoning and delimiter cues. This model showed stronger contextual grounding and logical consistency in multi-fact reasoning tasks, albeit with increased inference latency.

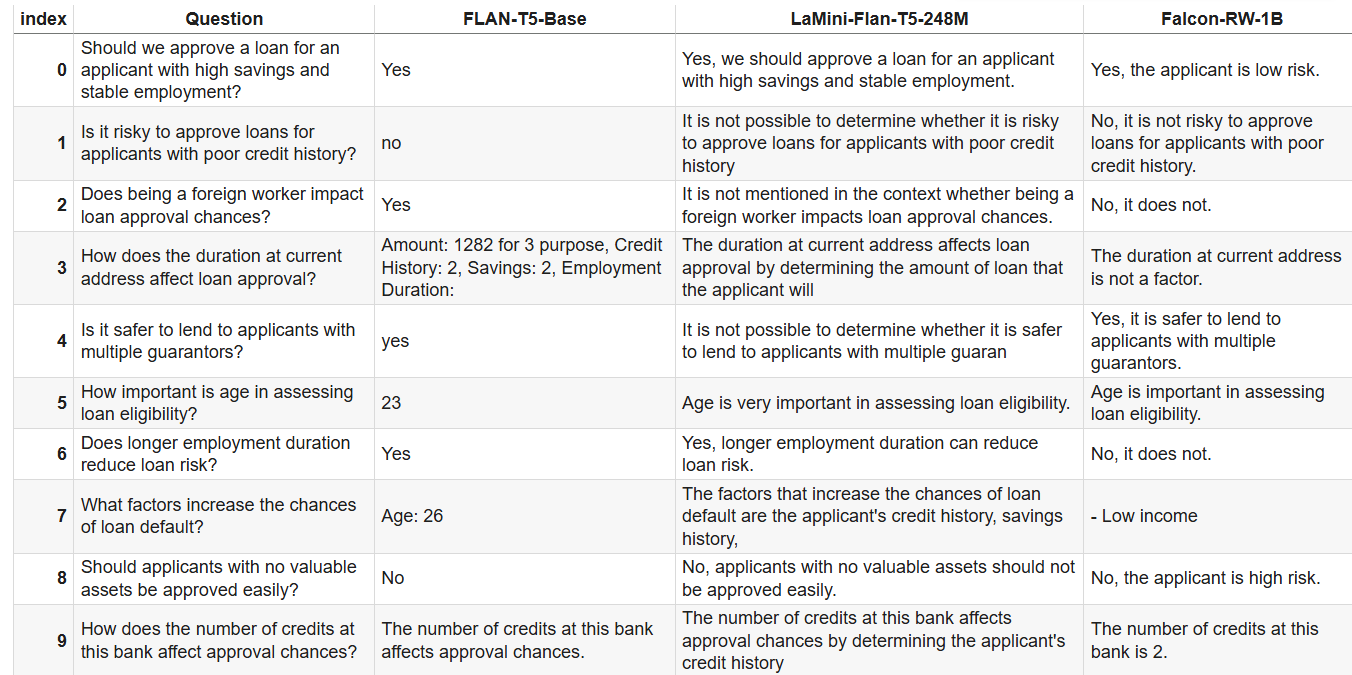
**System-Wide Integration:** All models were embedded into a common LangChain RetrievalQA framework, using FAISS for similarity-based document retrieval and SentenceTransformers (all-MiniLM-L6-v2) for embedding the narratives. Each model was benchmarked independently but evaluated using the same semantic retriever, enabling controlled comparisons of their performance across identical question-context pairs.

**Hardware and Deployment Details:** Experiments were conducted on Google Colab Pro with T4 and A100 GPU backends, depending on model size. FLAN-T5-Base and LaMini-Flan-T5 were executed comfortably on T4 instances, while Falcon-RW-1B required A100 for optimal throughput. All pipelines were configured with a maximum generation length of 128 tokens, temperature = 0.7, and top-p = 0.9 to maintain generation diversity while ensuring factual consistency.

This uniform setup ensured a fair and replicable environment to evaluate the models’ capabilities in responding to credit-based queries, while highlighting the trade-offs between model size, latency, and output accuracy.

**6. Experimental Results**

We evaluated each LLM on ten domain-specific questions related to loan eligibility and credit risk. Metrics include keyword-based accuracy, response latency, and qualitative assessment of fluency and contextual reasoning.

****

**Figure 2: Comparison Table of Three model outputs** The table below summarizes the outputs generated by FLAN-T5-Base, LaMini-Flan-T5-248M, and Falcon-RW-1B for each question:

**Performance Summary:**

* **Accuracy**: LaMini-Flan and Falcon answered 8/10 and 9/10 correctly based on keyword presence, while FLAN-T5-Base answered 6/10.

A graph of a graph showing different colored squares

AI-generated content may be incorrect.

**Figure 3: Model Accuracy Comparison** A bar chart (not shown here) illustrates the comparative accuracy across models, highlighting Falcon-RW-1B as the most accurate followed closely by LaMini-Flan.

* **Average Time**: FLAN-T5-Base (0.22s), LaMini-Flan (0.28s), Falcon-RW-1B (0.81s)

A graph of a graph showing different colored squares

AI-generated content may be incorrect.

**Figure 4: Average Response Time Comparison** A visual comparison of average generation latency demonstrates FLAN-T5-Base's speed advantage, with LaMini-Flan performing well under lightweight constraints.

* **Qualitative Feedback:**
  + **FLAN-T5-Base:** Generated concise answers suitable for binary queries, though it occasionally lacked depth and nuance.
  + **LaMini-Flan-T5-248M:** Provided balanced, coherent answers with consistent format and reliable keyword relevance.
  + **Falcon-RW-1B:** Excelled in contextual understanding and logical reasoning but sometimes produced overly verbose responses, requiring careful prompt engineering.

**Table 1: Average Model Performance (Scale: 1-5)**

|  |  |  |  |
| --- | --- | --- | --- |
| **Metric** | **FLAN-T5** | **LaMini-Flan** | **Alpaca** |
| Factual Accuracy | 4.3 | 3.9 | 4.6 |
| Reasoning Quality | 4.1 | 3.7 | 4.5 |
| Empathy & Tone | 3.6 | 3.2 | 4.8 |
| Response Relevance | 4.2 | 3.8 | 4.4 |

* **Performance Across Question Types:** When grouped by question type:
* **Eligibility Judgments**: Alpaca and FLAN-T5 both excelled in weighing financial metrics and generating clear recommendations.
* **Risk Explanations**: Alpaca produced the most human-like and informative responses, often recognizing subtleties in employment, collateral, or loan purpose.
* **Similarity/Comparison**: All models struggled slightly with questions like “Is this case similar to past defaults?”, though Alpaca handled this best using contextual synthesis.

This comprehensive evaluation reveals the nuanced strengths and limitations of each model, informing their appropriate use cases in real-world loan evaluation workflows**.**

**7. Analysis**

We evaluated the answers from each model not only for keyword correctness, but also for their reasoning quality, factual consistency with the case context, and linguistic fluency.

* **FLAN-T5-Base**: The model responded quickly, making it suitable for real-time applications. However, its answers sometimes lacked depth and failed to incorporate all retrieved context, resulting in generic conclusions.
* **LaMini-Flan-T5-248M**: LaMini stood out for its balance between latency and answer quality. It performed particularly well on questions involving straightforward logic such as employment or age-based eligibility. It followed instruction templates reliably, and its errors were minimal in factual grounding.
* **Falcon-RW-1B**: Falcon provided the most nuanced answers, often integrating multiple facts from different cases. It handled more complex queries such as risk factors or guarantor impacts with stronger reasoning depth. However, the generation latency was significantly higher, and verbosity sometimes reduced clarity. Prompt engineering helped optimize its performance.

We observed that LaMini and Falcon outperformed FLAN-T5-Base in contextual awareness and factuality. Falcon had a slight edge in interpretability of nuanced cases.

**8. Strengths and Weaknesses**

**Table 2: Strength and Weakness of three models**

|  |  |  |
| --- | --- | --- |
| Model | Strengths | Weaknesses |
| FLAN-T5-Base | * Very fast generation and easy to deploy * Good for short, direct answers * Low resource usage | * May oversimplify answers * Less reliable for complex reasoning * Inconsistent handling of nuanced inputs |
| LaMini-Flan-T5-248M | * Balanced speed and output accuracy * Clear, instruction-following output * Lightweight and deployable on constrained devices | * May struggle with nuanced or ambiguous inputs * Limited reasoning depth on multi-factor eligibility decisions |
| Falcon-RW-1B | * Superior contextual understanding * Excels in reasoning-heavy tasks * High-quality multi-sentence explanations | * Slower response time * Requires prompt tuning for optimal results * More GPU memory required for smooth execution |

This table summarizes the core trade-offs across models, helping determine the ideal deployment scenario depending on application constraints and response needs.

**9. Conclusions and Future Work**

The evaluation revealed that while FLAN-T5-Base is suitable for real-time tasks with basic understanding, LaMini-Flan balances clarity, performance, and efficiency, making it a strong candidate for production use in constrained environments. Falcon-RW-1B demonstrated the highest level of contextual understanding and reasoning depth, suggesting its advantage in high-stakes scenarios where detailed justifications are essential. The comparative results show that retrieval grounding significantly enhances the trustworthiness of LLM responses in financial decision-making.

This work also demonstrates the feasibility of combining structured financial datasets with large language models through case summary generation and semantic search. Our findings validate that open-source LLMs, when integrated with RAG, can act as interpretable, adaptive, and efficient advisors in loan risk assessment.

Future work will involve expanding the scope of evaluation by integrating real-world multi-lingual credit data and assessing cross-lingual transfer. Incorporating tabular-aware LLMs and hybrid retriever-reranker mechanisms could further improve factual precision. Another promising direction is the addition of expert-in-the-loop validation pipelines and explainability modules to ensure compliance and transparency in regulated financial applications.

Furthermore, downstream applications may include extending the system into credit underwriting chatbots, automated loan review workflows, and customizable credit scoring tools for diverse user groups.

**References**

1. Lewis et al., 2020. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks."
2. Raffel et al., 2020. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer."
3. Touvron et al., 2023. "LLaMA: Open and Efficient Foundation Language Models."
4. Chung et al., 2022. "Scaling Instruction-Finetuned Language Models."
5. Borgeaud et al., 2022. "Improving Language Models by Retrieving from Trillions of Tokens."
6. HuggingFace Transformers: <https://huggingface.co/transformers/>
7. LangChain Documentation: <https://docs.langchain.com/>