Universiteit Leiden

Master Computer Science

Secure Domain Adaptation of Mistral-7B for Banking: Fine-Tuning and Retrieval-Augmented Generation in ABN AMRO CDS

Name: Luca Girotti
Student ID: 2591006

Date: March 11, 2026

Specialisation: Artificial Intelligence

1st supervisor: Peter van der Putten
2nd supervisor: Suzan Verberne
Company supervisor: Bert Kiewiet

Master's Thesis in Computer Science

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

Context.
Large Language Models (LLMs) such as GPT-5 and Mistral-7B have recently pushed the boundaries of what is possible in natural language processing. Yet, in banking, their adoption is limited by strict rules on privacy, compliance, and security. At ABN AMRO, this translates into a clear preference for fully *on-premise*, open-weight solutions that can be adapted and examined in detail within the bank's secure infrastructure.

Objective.
This thesis examines how an open-weight LLM specifically Mistral-7B can be adapted to meet ABN AMRO's compliance requirements while serving the needs of the *Customer Data Solutions* (CDS) department. The study focuses on combining parameter-efficient fine-tuning with Retrieval-Augmented Generation (RAG) to produce a domain-aware assistant that operates entirely within the bank's private environment.

Method.
A custom Q&A dataset, thoroughly reviewed for privacy and accuracy, was built to capture CDS-specific knowledge. Three baseline systems were implemented: a fine-tuned LLM, a fine-tuned LLM with RAG, and a base model with RAG. These were tested against 100 prompts reflecting real CDS information needs, including organisational facts, internal policies, technical tools, and historical projects. Failures from this benchmark informed the design of three advanced RAG pipelines: multi-hop reasoning, evaluation-aware generation, and agent-based retrieval which were evaluated on the 21 most challenging queries, defined as those where both baseline RAG systems consistently failed.

Results.
In the baseline tests, RAG improved factual accuracy when the relevant information was available in the knowledge base, while fine-tuning proved more effective for "micro-facts" missing from retrieval. On the hardest queries, multi-hop and evaluation-aware RAG more than doubled the mean score over the baseline RAG, and reduced complete failures by over 60%. All experiments ran entirely on-premises, with critic modules successfully blocking responses that could breach internal policy.

Conclusion.
The findings show that with well-structured retrieval, targeted fine-tuning, and robust faithfulness checks, an open-weight LLM can be transformed into a secure and reliable knowledge assistant in a highly regulated banking context. The approach developed here offers a practical blueprint that could be adapted to other industries with similar compliance constraints.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The banking sector is arguably the most challenging environment for any kind of data-driven innovation. First, every new system or analytical tool must strike a balance between leading technological development and upholding the bank's deep-rooted commitments to security, privacy, and good reputation (Alur and Rachlin 2018). Internal reviews at ABN AMRO have revealed that frontline teams are heavily dedicating their work hours to collecting and integrating customer data from various internal sources.[1] Not only does this fragmentation make the decision-making process slower, but it also brings in the danger of inconsistencies.

The impressive growth of Generative AI – especially large language models (LLMs) like GPT-4 and Mistral-7B – presents a real chance to significantly facilitate and automate a large part of knowledge work. Nevertheless, banking information is very sensitive, and under ABN AMRO's governance framework, it is normally forbidden to expose such data to external AI services through commercial APIs. To encourage innovation while still being compliant, ABN AMRO has gone for an *on-premises* LLM solution. By utilising open-weight models such as Mistral-7B, the bank is capable of customizing, launching, and overseeing these systems entirely inside its secure environment (Mistral AI 2023).

ABN AMRO's *Customer Data Solutions* (CDS) department is in charge of the bank's customer-identity data. Its intensively regulated operations include orchestrating detailed onboarding and reincorporating customers through Know-Your-Customer (KYC) procedures up to managing the sensitive data records on a regular basis. Therefore, CDS is the main hub for both compliance and client services.

A Machine-Learning Engineering team within the department is respon-

[1] Internal CDS KPI dashboard, Q1 2025.

sible for building/managing CDS data and developing end-to-end pipelines to enhance data quality and support various customer-facing services. The entire fine-tuning, research, and experimentation mentioned in this thesis were performed in the private internal cloud environment of ABN AMRO, using the CDS on-premises Databricks platform and other closed and heavily secured internal infrastructures.

The main research question of this thesis is as follows: How to leverage domain-specific knowledge of ABN AMRO Customer Data Solutions (CDS) through fine-tuning and retrieval-augmented generation for adaptable, internally secure, generative AI models while maintaining compliance with licensing constraints and organizational goals?

The central intention behind this research is to demonstrate how secured, domain-specific Generative AI can be integrated into the banking operational workflows and provide a thorough, side-by-side evaluation of various large language model adaptation methods. Six clearly defined objectives directed this study:

O 1 We run **three baseline** experiments on Mistral-7B variants: two fine-tuned LoRA models (i) trained on CDS data alone and (ii) combined with Retrieval Augmented Generation (RAG), and a (iii) base (untuned) Mistral-7B paired with RAG.

O 2 **We deploy RAG** in a real-world scenario using ABN AMRO internal library documents such as annual reports, Confluence, and policies by dense retrieval through `snowflake-arctic-embed-l`, storage Delta Lake, and Databricks orchestration.

O 3 **We evaluate** these baseline models on 100 real-life prompts covering CDS roles, governance, project workflows, corporate strategy, and tricky hallucination trap issues. The domain experts rated each answer on a 0–10 scale.

O 4 We focus on the 21 prompts where both baseline RAG systems performed weakest, and evaluate sophisticated RAG methods including multi-hop, evaluation-aware, and agent-based variants.

O 5 We follow strictly ABN AMRO privacy, security, and legal rules throughout the whole course of data handling, fine-tuning, and inference, thus, designing a fully auditable and reproducible machine-learning pipeline.

O 6 Finally, we **deploy an advanced prototype**: we containerize the fine-tuned Mistral-7B model on an isolated Linux VM inside ABN AMRO

private cloud, thus, exposing authenticated REST endpoints for controlled internal testing purposes.

Chapter 2

Background and Related Work

This chapter places the thesis work in the context of previous studies concerning large language models (LLMs), domain adaptation, and retrieval-augmented generation (RAG), especially within the framework of tightly regulated enterprises such as the banking sector.

Besides discussing existing approaches for fine-tuning, retrieval-based augmentation, and evaluation methods, the chapter also highlights the areas where the thesis work is empirically driven.

2.1 Generative AI in Regulated Domains

Generative Artificial Intelligence (AI) is powered by the rapid development of transformer-based large language models (LLMs), which can potentially handle various natural language processing tasks. The language models released in the early days, for instance, GPT-2, provided a proof-of-concept for scaling up language models; while the more recent models including GPT-4 and Mistral-7B exhibited remarkable generalisation across language reasoning, summarisation, and question-answering tasks (Radford et al. 2019; OpenAI 2023; Mistral AI 2023).

Yet, without further measures, the model's generic power hardly fits the deployment scenarios of highly regulated sectors. In banking, the AI systems are expected to be compliant with additional rules on data privacy, audit, model governance, and operations control.

The regulatory and legal constraints impose significant restrictions on the usage of AI services hosted externally, especially when they involve sending sensitive data to third-party providers (Cotterell et al. 2023). Consequently, many banks either limit or totally block the use of commercial LLM APIs for their internal staff knowledge work.

Such restrictions have significantly heightened the interest in open-weight language models (OWMs). Examples of OWMs are Mistral-7B and Llama 4, which give organisations the capacity to thoroughly review, customize, and run the entire model stack on their hardware (Touvron et al. 2023). For Tier-1 banks, it is indispensable: they gain the privilege of trying out and innovating while maintaining full control over data, model behaviour, and risk of running. Hence, the LLM market keeps on changing fast, and the focus in the regulated context is for sure not on individual model generations but rather on reliable adaptation and deployment strategies.

2.2 Domain Adaptation of LLMs

General-purpose LLMs often need to be supplemented with domain-specific knowledge, terminology, and process aspects to work well in the enterprise context. The literature presents two separate methods that together cover most of the space between full-model fine-tuning and parameter-efficient adaptation.

2.2.1 Full Fine-Tuning versus Parameter-Efficient Adaptation

Full-model fine-tuning is the process of updating all the parameters of a pretrained model with a domain-specific dataset. The downside with this is the high consumption of computation and the resulting difficulty producing an easily reproducible, auditable, and roll-backable model – especially troublesome in highly regulated environments (L. Zhang et al. 2022).

Parameter-efficient fine-tuning methods provide a potentially more accessible alternative. In Low-Rank Adaptation (LoRA), for instance, a relatively small set of trainable parameters is introduced while keeping the base model weights frozen. E. Hu et al. (2022) show that performance-wise, LoRA can stand against full fine-tuning while using less than a fiftieth of the computing power. Later works like QLoRA cap the memory usage through quantization (Dettmers et al. 2023).

Parameter-efficient fine-tuning methods offer substantial benefits if the banks intend to run the models on their premises: among others, they lessen the GPU usage, and hence facilitate the governance of the model; and they enable faster iteration cycles. Therefore, such methods have become the adaptation strategy of choice in many enterprise LLM pipelines.

2.3 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a method that helps to solve the main issues of LLMs, which are when LLMs make up facts, do so incorrectly, or when they do not have access to domain-specific and up-to-date knowledge that was not part of the original training data. This disadvantage is especially important in business environments where most of the information used to train models is public, and these models have to provide organization-specific answers that are based on internal documents. There, the problem is not only to stop hallucination but also to make sure that the answers correspond to the right institutional context and are not just general knowledge of the domain. With the help of retrieved evidence, which is used at the moment of inference, RAG allows the model to base its answers on real, organisation-specific source material.

What happens in a RAG setup is that first, the external world (or the knowledge from a document collection) is encoded in a vector space and then the documents are indexed for a similarity search. At inference time, relevant passages are retrieved and the model prompt is augmented with those passages so that the model can ground the generation in real source material.

REALM (Guu et al. 2020) was among the first systems to combine retrieval and language modelling by integrating retrieval into the language-modelling objective. The RAG framework, outlined by Lewis et al. (2020), who showed that adding retrieval significantly lowers the number of times a model generates text that is not factually supported. Since then, RAG has become a common tool for enterprise question answering, especially in scenarios where knowledge is proprietary, frequently updated, or too large to be memorized during training.

Unlike fine-tuning, RAG does not change the model's parameters permanently. It is known that fine-tuned models will find it hard to adapt if new information is added or if some of the previously learned facts become outdated, since the only way to update them is through the retraining process. On the other hand, RAG can easily include new or corrected information just by changing the retrieval index. However, RAG also relies heavily on the quality and the up-to-dateness of the indexed documents: if the knowledge base still contains outdated or incorrect information, then the model can generate incorrect answers that are well supported by the content.

The flip side is that RAG also brings forth some of its own limitations. The RAG model needs to be able to rely on a knowledge base that is maintained to be both timely and complete, account for changes that occur dynamically in the world, and compliance, roles and privileges based filtering, all adds to

the operational complexity.

Consequently, RAG's success is highly correlated with the corpus coverage, the quality of retrieval, and the deployment context rather than inherently better than fine-tuning alone.

2.4 Advanced RAG Variants

Usually, single-step retrieval is followed by generation in the case of standard RAG pipelines. Although this method is nice and clean for simple fact-finding, it is often quite limited if one wants to tackle compositional questions or partially verified evidence, or if the text is ambiguous. Several papers especially in the NLP community are dedicated to developing better RAG systems in terms of reasoning depth, faithfulness, and flexibility.

Besides architectural extensions, tuning the main components of RAG can already help significantly strengthen its performance. The choices of a chunking strategy, an embedding model, the number of retrieved documents (k), the reranker, and the prompt for the generator all directly impact the factual grounding and robustness of the generated output. Typically, without changing the overall pipeline structure, a well-thought-out optimization of these components delivers substantial improvements.

The advanced versions described in the next sections should thus be considered complementary, meta-level extensions that raise the bar for a well-tuned baseline RAG system. They focus on adding extra reasoning or verification layers and do not aim to replace the core retrieval and generation mechanism. The implementation-specific design decisions taken for this thesis work are outlined in the next chapters.

2.4.1 Multi-hop Retrieval

Multi-hop Retrieval-Augmented Generation (RAG) breaks down a complex information request into a series of smaller, simpler questions and retrieves supporting pieces of evidence step by step (Press et al. 2022).

The answer is eventually formulated through reasoning over several successive retrieved contexts. It works well for complex issues that need different documents or concepts to be combined.

2.4.2 Evaluation-Aware (Faithfulness-Checked) RAG

Evaluation-aware RAG designates a stage of verification following generation. A second model called a *critic* determines if the generated response rests on

the evidence that has been pulled out. In the case the answer is not backed up by the text, the system either tries regenerating the answer or refuses to answer altogether.

This framework is crucial to reducing hallucinations and increasing reliability which is of paramount importance in regulated environments (Shuster et al. 2023).

2.4.3 Agent-Based and Tool-Augmented RAG

In the agent-based RAG paradigm, language models are combined with agents capable of planning and executing actions, such as issuing additional retrieval requests or invoking external tools like calculators.

The use of agents, capable of interacting with tools, provides the model with more flexibility to select the next best action, rather than just sticking with a fixed retrieval generation pattern.

However, agent-based systems are inherently more complex, so one needs to be very careful when implementing them in situations where content needs to be filtered in real-time, e.g., based on user roles or data sensitivities (Yao et al. 2023; Xi et al. 2023).

2.5 Evaluation of RAG Systems

RAG systems are exposed to some additional sources of variability, besides those found in standard language-model benchmarks, which can render the evaluation challenging.

Besides quality, evaluations need to consider retrieval accuracy, grounding fidelity, and interpret failure modes arising from missing, misleading, or outdated context.

Typical RAG evaluation metrics include context precision and recall (i.e., whether the retrieved passages are relevant and sufficient), answer relevance, and faithfulness (whether the generated answer is fully supported by the retrieved context) (Es et al. 2023; Z. Liu et al. 2023; Chen et al. 2023). Tools like RAGAS make these metrics measurable via automated scoring methods (Es et al. 2023).

Automatic metrics of this kind were not considered the main type of evaluation in this thesis. The primary reason is that many CDS prompts do not allow for one definitive answer, and thus correctness has to be determined in terms of internal policy interpretation and institutional context. Besides, proprietary documents cannot be revealed to external evaluation services, and automatic faithfulness metrics have been demonstrated to correlate only

imperfectly with domain-specific factual accuracy (Z. Liu et al. 2023).

In consequence, a detailed human rubric was chosen as the most trustworthy assessment method in this regulated enterprise environment.

Previous research found that they can certainly fool people when the answer surface matches the expected one, but there can still be many retrieval failures in the background, which is why they advocate for evaluations where generation and retrieval performances are scored separately (Lewis et al. 2020; Guu et al. 2020; Es et al. 2023; Z. Liu et al. 2023; Chen et al. 2023).

In commercial settings, the luxury of a publicly available, representative benchmark is rarely at hand, let alone the difficulty of acquiring real transaction data that is proprietary.

Hence, most of the related studies opt for synthetic or publicly available datasets, which are inherently limited and may fail to capture some real-world scenarios.

2.6 LLMs and Knowledge Integration in Banking

Historically, the use of neural language systems in banking was limited to narrowly defined tasks that usually used structured data such as transaction records or customer attributes combined with task-specific neural models. Y. Liu et al. (2021) focused their research on banking neural language models designed to handle banking transactions.

Recently, several works looked into the large-scale general-purpose LLMs adaptation for financial QA. Fine-tuning LLMs can lead to a significant improvement in performance on financial QA as Xie et al. (2023), Zhao et al. (2024), Y. Wu et al. (2024), S. Wu et al. (2023), Y. Yang et al. (2023), and P. Liu et al. (2023) show. However, as their solution is based on cloud-hosted infrastructure, it can still be very problematic from the perspective of governance and compliance for many banks in Europe.

Different authors utilize different combinations of fine-tuning and retrieval components to investigate whether the mixture of the two can still be improved in domains with very specific language, stable corpora, and reasonable evaluation setups.

The results are contradictory: thus, no single adaptation strategy is universally dominant, and controlled, domain-specific comparisons are needed.

2.7 Gap Analysis

Several gaps in the field can be identified based on the literature:

- The comparison of architectures involving only fine-tuned models, fine-tuned + RAG, or base-model + RAG in controlled experiments with real- world enterprise benchmarks has been overlooked.
- The evaluation of RAG systems that rely solely on proprietary, internal documentation has hardly been addressed.
- Only a handful of studies examine the entire pipeline of fine- tuning, retrieval, evaluation, and secure deployment within a Tier-1 banking environment.
- There is a lack of in-depth treatment of high-level RAG variants based on the failure cases identified empirically as opposed to using synthetic benchmarks.

The thesis tackles these discrepancies by offering a controlled comparison based on failures between fine-tuning and retrieval strategies in a real banking environment and further, by testing advanced RAG variants solely on the instances where the baseline systems are weakest.

Chapter 3

Methodology

This chapter is about the sources of data, model adaptation strategies, the retrieval pipeline, the deployment setup, and the evaluation protocol that were utilized in the thesis. Great emphasis is laid on the reproducibility of the results and the clarity of the design choices. Hence, the quantitative performance comparison is left to Chapter 4. The whole suite was created and executed on ABN AMRO's secure, in-house infrastructure.

The methodological framework of this study is divided into two phases.

Phase 1 A tightly controlled benchmark of three Mistral-7B architectures on a set of 100 prompts, which is used to establish the baseline performance and identify failure cases.

Phase 2 A focused evaluation of advanced RAG variants, with main emphasis on the failure cases identified in Phase1.

The setup, equipment, and experimental implementation are described in detail in Chapter 4, whereas the discussion of the advanced variants is presented in Section 5.9.

In both stages, stringent privacy, security, and auditing requirements of ABN AMRO were adhered to in all data processing, model training, and inference operations, thereby creating a reproducible and compliant framework for deploying large language models in a regulated enterprise environment.

3.1 Data Collection and Preparation

I hand-crafted the fine-tuning dataset from scratch to fulfill the internal knowledge requirements of ABN AMRO's *Customer Data Solutions* (CDS) department only. The dataset was wholly designed for this thesis and did not exist before the commencement of the project. Every single example is a realistic user query– assistant-style reply, essentially what an internal

knowledge assistant would provide to CDS employees.

By going through the internal documentation of CDS and turning the most frequently asked information needs into question–answer pairs in natural language, the dataset took shape. The questions were set to reflect the kind of realistic phrasing used by non-technical domain experts, whereas the answers were designed to be brief, factually accurate, and consistent with internal terminology and policies.

The dataset essentially deals with the following four categories: (i) the organisational structure, roles, and responsibilities within CDS; (ii) CDS teams, their mandates, roles, and key stakeholders; (iii) introductions to current and past CDS projects; (iv) internal policies, tools, workflows, and terminologies specific to CDS.

The author manually double-checked all entries for accuracy, clarity, and consistency. Personal identifiers and sensitive information were removed through complete anonymization, in compliance with the privacy requirements of the company. The finished dataset was kept in an instruction–response format (<user>...<assistant>), thus ensuring compatibility with instruction-tuned inference behaviour. All sequences were cut to a maximum length of 1024 tokens to fit the operational context window.

To conclude, the fine-tuning dataset consists of around **1,600 question–answer pairs**. Besides the fine-tuning dataset, there is a completely separate evaluation benchmark which was made up of exactly **100 unique prompts**. The prompts were created by hand to represent the real needs for CDS information, and they were intentionally excluded from the fine-tuning dataset. They were only used as a separate test set for model evaluation (see Chapter 4). The author intentionally restricted the size of the fine- tuning dataset, because the main purpose was to introduce high-quality domain-specific knowledge rather than to perform large-scale language modelling. Therefore, a handpicked dataset was chosen over synthetic data generation, with correctness, auditability, and trustworthiness as the main considerations.

When this project started, there was no retrieval-augmented generation (RAG) infrastructure at CDS. There was no document corpus for indexing, embedding pipeline, or retrieval framework ready for reuse. So, the whole RAG pipeline, i.e., document collection, preprocessing, sanitization, embedding, indexing, and prompt integration, was, through a deep level of collaboration, designed and implemented entirely from scratch within the bank's on-premises environment.

Retrieval-Augmented Generation (RAG) also relied on a separate document collection, which I gathered for this research. The corpus is made up of **approximately 3,500 pages** of internally produced annual reports,

policy documents, and technical documentation in PDF format. I divided all documents into overlapping text windows of about 500 characters, with a 100 characters overlap. Chunks shorter than 20 characters were removed to reduce retrieval noise. As a result, a total of **about 18,000 document chunks** were indexed for retrieval.

This chunking method aims to achieve a balance between semantic coherence and retrieval precision, and was further validated through a small pilot experiment conducted prior to the main evaluation.

Example. An example entry from the fine-tuning dataset is shown below:

> **User:** What is the key task of the CDS department within ABN AMRO?
> **Assistant:** The Customer Data Solutions (CDS) department is the major stakeholder managing and maintaining customer identity data across the whole bank. This data underpins processes such as customer onboarding, Know-Your-Customer (KYC) compliance, and periodic customer reviews.

3.2 Model Selection and Weight Conversion

We chose Mistral-7B as the base language model as it is a good compromise between performance, model size and memory footprint. Additionally, its permissive open-weight licence makes it an ideal choice for fully on-premises deployment in a regulated environment.

Official weights were obtained from the vendor's mirror and converted into Hugging Face format through the maintainers' conversion script.[1] The converted model became the common backbone for all baseline and advanced architectures tested in this thesis.

3.3 Fine-tuning and Model Architecture Design

We chose parameter-efficient fine-tuning (PEFT) over full fine-tuning in order to lower the computational cost, support fast iteration, and simplify rollback and audit procedures. Low-Rank Adaptation (LoRA) was employed with the following settings:

[1] https://github.com/huggingface/transformers/blob/main/src/transformers/models/mistral/convert_mistral_weights_to_hf.py

- PEFT method: `LoRA`
- Rank $r = 8$, scaling factor $\alpha = 16$
- Target modules: `["q_proj", "v_proj"]`
- Optimiser: AdamW, learning rate 2×10^{-4}
- Training duration: 30 epochs, batch size 4

To ensure reproducibility, fixed random seeds were used for all training runs. Early stopping was not applied, as the dataset was small and stable convergence was observed during pilot experiments. All training was conducted exclusively on the CDS on-premises Databricks GPU cluster.

No explicit train–validation–test split was applied to the fine-tuning dataset. The LoRA adapter was trained on the full curated CDS dataset, as the primary objective was to inject verified domain knowledge rather than to optimise generalisation performance. Evaluation was performed exclusively on a *separate* benchmark of 100 prompts (Chapter 4), which was not used during training.

Three baseline architectures were considered:

(a) **Fine-tuned Only**: LoRA-adapted Mistral-7B without retrieval.

(b) **Fine-tuned + RAG**: the same fine-tuned model combined with a retrieval pipeline.

(c) **Base + RAG**: the unmodified Mistral-7B paired with the same retrieval pipeline.

This design enables the separate assessment of the contributions of parameter adaptation and retrieval-based grounding.

3.4 RAG Construction: Embeddings, Retrieval, Generation

Every document chunk was embedded using `Snowflake/snowflake-arctic-embed-l` (version 1.03) (Li, C. Wu, et al. 2024). This dense embedding model was selected due to its strong semantic retrieval performance and its ability to handle long enterprise documents.

The resulting embeddings (dimension 1024) were stored in a FAISS index using the default configuration (`IndexFlatL2`)[2]. The `IndexFlatL2` index

[2]FAISS documentation: https://github.com/facebookresearch/faiss

computes nearest neighbours using L2 distance without approximation, as opposed to approximate indexing structures such as IVF or HNSW, which performs exact nearest neighbour search based on L2 distance. Given the moderate size of the document corpus and the deployment constraints, indexing speed was not a limiting factor. Instead, this choice prioritised retrieval accuracy and deterministic behaviour.

During inference, the retrieval-augmented generation pipeline operates as follows:

1. The user query is embedded using the same embedding model.
2. The top $k = 3$ most similar document chunks are retrieved via the FAISS index.
3. The retrieved passages are inserted into a fixed prompt template.
4. The fine-tuned or base Mistral-7B model is invoked to generate a response.

The choice of retrieving the top $k = 3$ document chunks reflects a trade-off between contextual coverage and noise. Within the CDS environment, most benchmark queries are factual and extractive, with the required information typically contained in one or two document chunks. Pilot experiments indicated that increasing k often introduced redundant or weakly related context, which could degrade generation quality for a relatively compact model such as Mistral-7B. Although higher k values may benefit more compositional queries, a smaller retrieval set proved sufficiently stable for the majority of cases considered here.

Text generation was performed using a decoding temperature of **0.7**, chosen as a reasonable default based on pilot experimentation and prior internal practice, balancing factual accuracy and readability. Apart from this setting, all other generation hyperparameters were kept at their default values in the respective framework. In particular, nucleus sampling (`top_p` = 1.0) was left unchanged and `top_k` sampling was disabled.

No quantisation was applied during inference; all models were executed at full precision to avoid introducing additional sources of noise into the evaluation.

3.4.1 RAG Prompt Template

At inference time, retrieved passages are placed into a fixed prompt template that constrains the model to answer solely based on the provided context. An example of the prompt template is shown below:

System: You are an internal knowledge assistant for CDS. Answer using only the provided context.

Context: [Retrieved document chunks]

User: [User query]

An example benchmark prompt is:

"What does the CDS department primarily support within the bank?"

The generated output was stripped of prompt artefacts and passed through regex-based safety filters before being delivered to the user.

3.5 Advanced RAG Variants: Methodological Overview

Besides the basic RAG model that was first described, the author worked on three advanced RAG variants that are meant to solve certain problems that the first experiments showed. This section aims to describe the basic thinking and the architectural extensions of these variants. The precise experimental outcomes and numerical evaluations can be found in Chapter 4. The main point here is the methodological positioning and the design principles that guided their inclusion.

These variants extend the baseline pipeline while reusing the same document corpus, embedding model, and retrieval infrastructure, thereby ensuring comparability and reproducibility.

Multi-hop RAG was introduced mainly to deal with user queries that require two or more pieces of evidence to arrive at a conclusion (as opposed to a single retrieved passage). The original user query is first decomposed into a small sequence of factual sub-questions using a dedicated prompt template. Each sub- question is passed independently to the same retriever, which uses the same embedding model and top-k configuration as in the baseline RAG. The combined evidence from all hops is then turned into a single context, which is used by the generator to produce the final answer.

The main distinguishing feature of evaluation-aware RAG is the introduction of an explicit verification step after answer generation. In this variant, the generator's initial answer is fed into a verification prompt that compares the answer against the retrieved evidence to detect missing or inconsistent claims. If the verification step identifies unsupported statements or evidence gaps, the answer is marked as unreliable. This mechanism does not perform

additional retrieval, but instead functions as a factuality filter whose goal is to prevent extrinsic hallucinations over retrieved documents.

Agent-based RAG was only partially developed due to the restrictions imposed by enterprise deployment and auditing requirements. The agent follows a ReAct-style decision-making loop that determines whether an external tool is required to answer a given query. In this thesis, the agent's tool access was deliberately limited to the baseline retriever and a calculator for performing explicit numerical operations. The agent was not given access to any additional databases, APIs, or search mechanisms. As a result, tool usage was only permitted in cases where the need for arithmetic computation was clearly identifiable.

All advanced RAG variants share the same underlying retrieval index, embedding model, and generation settings as the baseline system. Differences in performance can therefore be attributed to architectural extensions rather than to changes in data or infrastructure.

Chapter 4

Experiments

This chapter is dedicated to the description of the experimental setup that was employed to evaluate the various model architectures and retrieval strategies presented in this thesis. The experiments were conducted within an environment comprising both hardware and software components, deployment settings, benchmark design, and the evaluation protocol.

Whereas Chapter 3 laid down the model architectures and pipelines, this chapter is about their empirical evaluation under controlled conditions.

The experimental setup has two levels:

1. A comprehensive **100-prompt benchmark** for assessing the overall performance of three primary architectures.

2. A targeted **21-prompt hard-case test** for analyzing the impact of state-of-the-art retrieval methods on the most difficult cases.

The experimental design follows a two-phase structure. First, a broad benchmark of 100 prompts is employed to compare baseline architectures that combine fine-tuning and retrieval. Subsequently, a focused hard-case evaluation is performed on a smaller set of 21 prompts, identified as failure cases in the baseline analysis, which also serve as the basis for the advanced RAG experiments presented in Section 5.9.

The primary focus of this chapter is on how the experiments were conducted; the quantitative results and their analysis are presented in Chapters 5 and Section 5.9.

4.1 Hardware and Software Environment

This section outlines the computational environment used to run all experiments. Providing details of the hardware and software configuration is impor-

tant for the reproducibility of the results and for understanding the practical constraints under which model training and evaluation were performed.

Experimental fine-tuning and retrieval were done on a dedicated Databricks GPU cluster running in ABN AMRO's secure infrastructure. The setup was:

- **Databricks Runtime:** 16.4 LTS ML
- **Node Type:** Standard `H100_V5`
- **Per-node specification:** 40 vCPUs, 320 GB RAM, 80 GB H100 GPU
- **Cost:** 13.06 DBU per hour (Databricks list price per DBU at the time of writing: approximately $0.50 per DBU-hour (Databricks 2026))

The environment ran Python 3.10 and the latest versions of `transformers`, `peft`, and `faiss`.

The bank's own GitLab instance was used for version-control of the code, configuration files, and experimental artefacts. CI/CD pipelines and MLflow tracking were enabled to guarantee full reproducibility and audit trails.

4.2 Model-Hosting Environment

To deploy, LoRA-tuned Mistral-7B and the RAG retrieval stack were both packed in containers and placed on a secure Linux VM in ABN AMRO's private cloud for hosting.

Access to models was through REST endpoints authenticated by OAuth and encrypted by TLS only. The inference logs were continually streamed to the bank's SIEM system for live monitoring. Neither public endpoints nor external services were exposed.

4.3 Evaluation and Experimental Protocol

A 100-prompt benchmark was used to evaluate the system. The prompts were carefully designed to represent a wide range of information needs within the CDS domain. Most prompts focused on CDS-specific knowledge and ABN AMRO corporate facts, while a small subset consisted of deliberate hallucination-trap questions aimed at probing unsupported generation and boundary behaviour.

Across the three baseline architectures, this resulted in a total of **300 evaluated model responses** (100 prompts $\times$ 3 systems).

Responses were assessed on a 0–10 scale with respect to accuracy, completeness, and clarity.

The selection of a 0–10 scale was intentional. A binary or very rough evaluation (e.g. correct/incorrect or 1–5) would not have been able to capture the notion of partial correctness, subtle hallucinations, or the overall factual grounding of retrieval- augmented generation, which often varies. In fact, most of the answers were not entirely right or wrong, but showed gradual differences in the level of factual accuracy, the extent of evidence usage, or the presence of minor unsupported claims. Thus, a 0–10 scale gave the possibility to distinguish more accurately between a complete failure, a partially grounded answer, and a fully correct, well-supported one.

The definitions of these three evaluation dimensions and the manual annotation procedure are described in Section 4.3. The initial assessment was performed by the author and subsequently verified by a peer from the CDS domain. In addition, all outputs were subjected to both manual and automated safety checks to ensure compliance with internal policies.

Human-judged rubrics were preferred over reference-based automatic metrics (e.g. BLEU or ROUGE), as many CDS queries do not admit a single canonical answer. In an enterprise RAG context, factual accuracy and adherence to internal policy are prioritised over surface-level textual similarity. Prior research has also shown that automatic metrics correlate poorly with factual accuracy and hallucination risk, particularly when answers must be grounded in multiple internal documents. For these reasons, human evaluation was considered the most reliable assessment method in this case.

4.4 Baseline Experiment Design (100 Prompts)

This section outlines the experimental setup used to establish the baseline performance of the evaluated architectures. The baseline benchmark serves as the reference against which all subsequent analyses are compared, including the identification of failure cases and the assessment of more advanced retrieval strategies.

The baseline assessment compared three different architectures:

(a) **Fine-tuned Only** LoRA-tuned Mistral-7B trained on the CDS Q&A dataset.

(b) **Fine-tuned + RAG** the generator fine-tuned once, combined with an added retrieval layer.

(c) **Base + RAG** the Mistral-7B out-of-the-box paired with a retrieval pipeline of the same type.

The test used 100 prompts that were representative of real-world situations, including CDS governance, team structures, workflow descriptions, corporate facts taken from ABN AMRO's annual reports, and deliberately misleading "hallucination trap" questions.

Steps for each question were:

1. Perform identical model decoding on all three models.
2. For RAG variants, first, generate the prompt embedding, then retrieve the top 3 passages, put the pieces in the same template (as defined in Section 3.4.1), and finally produce the reply.
3. Keep all original data for subsequent grade anonymisation purposes.

4.5 Phase-2 Experiment Design: Advanced RAG Variants

After the evaluation of the baselines, Phase 2 focused on the 21 prompts for which *both* baseline RAG systems scored ≤ 3. The following enhanced retrieval strategies were considered:

4.5.1 Multi-hop RAG: Query Decomposition and Evidence Aggregation

Multi-hop Retrieval-Augmented Generation was implemented as a failure-driven extension of the baseline single-hop RAG pipeline, targeting prompts that require reasoning over multiple pieces of evidence. The approach follows a four-stage process consisting of query decomposition, per-hop retrieval, intermediate generation, and final answer synthesis, in line with prior work on multi-hop question answering and retrieval-based reasoning (Z. Yang et al. 2018; Khattab and Zaharia 2021).

Given an original user query, the model first generates a small set of retrieval-oriented sub-questions. This decomposition step is performed using the same generator model as in the baseline configuration, but with a dedicated prompt that explicitly instructs the model to *only* produce sub-questions and not attempt to answer the query. The prompt template used for decomposition is shown below:

> **System:** You are assisting a retrieval system. Decompose the user question into short, factual sub-questions that can be answered by searching internal documents. Do not answer the question.

> **User:** Decompose the following question into at most **3** sub-questions. Return **only** a numbered list.
> *Question:* [ORIGINAL_QUESTION]

In all experiments, the number of sub-questions was capped at three to remain within the global execution-time budget. If fewer sub-questions were produced, the output was used as-is.

Each sub-question is then processed independently in a per-hop retrieval–generation loop. For a given sub-question, the query is embedded using the same dense embedding model as in the baseline RAG setup, and a separate nearest-neighbour search is executed against the shared FAISS index. For each hop, the top $k_{\text{hop}} = 3$ document chunks are retrieved, mirroring the baseline retrieval depth. This design follows the intuition that decomposed queries benefit from targeted evidence retrieval rather than a single global search over the original question (Z. Yang et al. 2018).

The retrieved passages are inserted into the standard RAG prompt template, and the model generates a concise intermediate answer specific to that sub-question. No reranking is applied, and the same decoding parameters as in the baseline experiments are used. This choice was made to preserve comparability with the baseline RAG pipeline and to avoid introducing additional sources of variation.

After all sub-questions have been executed, a final synthesis step is performed. In this stage, the model is prompted once more with the original user query, together with the list of generated sub-questions and their corresponding intermediate answers. No additional retrieval is performed at this point. The final prompt instructs the model to integrate the intermediate results into a single coherent response while remaining grounded in the previously retrieved evidence:

> **System:** You are an internal knowledge assistant for CDS. Answer using only the provided information. If the evidence is insufficient, state this explicitly.
>
> **Sub-questions and intermediate answers:**
> [q_1] – [answer_1]
> [q_2] – [answer_2]
> [q_3] – [answer_3]
>
> **User:** ORIGINAL_QUESTION

This design explicitly executes sub-questions as independent retrieval and generation steps, followed by a single final synthesis generation. Alternative

designs, such as aggregating retrieval results across hops and performing only one generation step, were not used. The chosen approach improves evidence coverage while preserving interpretability and ensuring that each intermediate reasoning step can be logged, inspected, and reproduced, which is particularly important in a regulated enterprise environment.

4.5.2 Evaluation-Aware (Faithfulness-Checked) RAG

Evaluation-aware Retrieval- Augmented Generation (RAG) takes a step further from the original RAG pipeline by adding a post-generation check that confirms whether the generated answer is based on the retrieved evidence. In contrast to multi-hop RAG, this model variation does not change the retrieval stage itself: the same top-$k = 3$ document chunks are retrieved for each query, and the only difference is in how the generated response is validated before being sent back to the user (Asai et al. 2023; Gao et al. 2023).

Once the primary response is produced with the standard RAG prompt template, a secondary language model operating as a *critic* is called in. The critic here is basically the same Proviso base Mistral-7B model used for generation, but the weights are frozen and it is instantiated in a separate verification role. There is no additional fine-tuning for the critic. Using the same model family allows consistent interpretation of domain terminology without requiring a dependency on external or black-box verification mechanisms.

Both the retrieved documents and the generated answer are given to the critic, which is then instructed to focus exclusively on factual correctness rather than, for example, style or completeness. The model is limited by a specific verification prompt to produce only a binary answer indicating whether the information contained in the answer is completely supported by the supplied evidence, as follows:

> **System**: You are a verification assistant for CDS. Make sure the answer is completely supported by the evidence. Don't use external knowledge.
>
> **Evidence:** [Retrieved document chunks]
>
> **Answer:** [Generated answer]
>
> **Task:** If all facts in the answer are evidence-based, return **SUPPORTED**. Otherwise, return **UNSUPPORTED**.

If the critic judges the answer as `SUPPORTED`, the answer is accepted and returned as is. A decision of `UNSUPPORTED` does not prompt the system

to alter the query depth or retrieve additional documents, but rather to make one more attempt at regeneration with more stringent evidence-usage requirements.

On the second try, the original RAG prompt template is changed to clearly restrict the model to citing or strictly grounding each factual statement in the retrieved evidence. The changed prompt is as follows:

System: You are an internal knowledge assistant for CDS. Answer strictly and only based on the provided evidence. Do not infer, generalize, or use external knowledge. If the evidence is insufficient, please state clearly: "Insufficient evidence in retrieved documents."

Evidence: [Retrieved document chunks]

User: [Original query]

The answer generated in this way is judged again and, if still unsupported, the system communicates a refusal to respond, indicating a lack of sufficient supporting evidence.

From the perspective of a single document, this thesis work shows several potential benefits of adding a stronger factuality check to outputs. Most recently, Flan-style instruction prompting has set the state of the art in few-shot chain-of-thought prompting on arithmetic reasoning tasks (E. J. Hu et al. 2022), and prompt–retriever techniques have demonstrated near human-level performance on multiple-choice language understanding benchmarks (Shin et al. 2020). While recent work has shed light on methods for detecting and suppressing hallucinations in language generation (Kassner et al. 2022; Skeppstedt et al. 2022), these approaches have rarely been applied directly to LLM-based retrieval-augmented generation pipelines to explicitly reduce extrinsic hallucinations over retrieved documents.

4.5.3 Agent-Based RAG

Agent-based RAG, or Retrieval-Augmented Generation with agents, allows the language model not only to produce an answer but also to plan its intermediate reasoning steps and, only if necessary, call external tools before the final answer generation (Yao et al. 2023; Schick et al. 2023).

The agent-based variant in this thesis was deliberately very minimal and was mainly a proof-of-concept within the constraints of the study and the enterprise deployment context. The agent used the same document corpus and retrieval pipeline layer as the baseline RAG systems and had a single external tool at its disposal: a **calculator**.

The calculator basically was a Python function that was made accessible as a tool inside the agent loop. The tool received a structured arithmetic expression as input (e.g., "1250000 * 0.03") and returned the calculated nu-

meric result. A brief tool description was given to the model, stating that the calculator is only to be used for explicit arithmetic operations. The output was subsequently reintroduced into the context of the agent before the final answer generation.

There was no additional retriever, database, or API exposed to the agent.

Per query, the agent performed a ReAct-style loop that contained: (i) an internal reasoning step, (ii) a determination of whether the tool was needed, and (iii) the answer generation task.

Using the tool was not always necessary and was therefore only performed when the model identified a numerical operation (e.g. totals, percentages, or comparisons) that it could not reliably perform in free-form text.

The calculator was invoked only when an explicit operation was required, and the resulting expression output was returned into the agent's context for answer generation. In cases where tool use was not deemed necessary, the agent directly generated answers with the help of the retrieved documents.

The agent was explicitly not given the freedom to search for additional documents, increase retrieval depth, or access internal structured databases. This limitation was intentionally imposed based on both the restricted tool availability in the experimental setting and the CDS environmental compliance constraints.

4.5.4 Shared Experimental Setup for Advanced RAG Variants

The document corpus and FAISS index from Phase 1 were employed in all advanced variants. Therefore, differences in performance are solely attributed to changes in retrieval and reasoning strategy rather than differences in the underlying knowledge base.

A prompt was delineated as hard when both baseline RAG variants (RAG+FT and RAG+Base) obtained a rubric score equal to or less than 3 on the 0–10 scale. This cut-off equates to poorly performing cases as per the evaluation rubric; thus, it represents serious grounding or retrieval errors. By resorting to this measure, a group of 21 prompts was obtained, which were used for Phase 2 evaluation.

For these 21 prompts, descriptive statistics, failure-rate reductions, and prompt-level comparisons were computed using the same scoring rubric as Phase 1. An independent double-blind review was carried out for all outputs by the author and an independent CDS domain assessor.

The runs shared the same tokenizer and decoding parameters as the baseline tests, with two additional restrictions:

- a **15-second timeout** per query to keep the multi-hop as well as the critic loops in check;
- a detailed recording of **sub-queries, retrieval traces, critic verdicts, and tool calls** for a later root-cause diagnosis.

4.6 Testing Protocol

This section describes the methods followed for carrying out and assessing each experimental run. The procedure was designed to ensure uniformity across models, minimise scorer bias, and facilitate follow-up error analysis and reproducibility of results. Each experimental run was conducted in exactly the same manner, following the sequence of steps outlined below.

The procedure for each run was:

1. Run 100 prompt evaluations with each model.
2. Collect answers, execution time, and retrieval logs.
3. Grade without names or indication of the author, using the predefined 0–10 rubric described in Section 4.3, which evaluates accuracy, completeness, and clarity.
4. Store scores and notes in a common file.
5. Save time for error analysis after marking things that make a clear difference.

Chapter 5

Results

The chapter reveals the outcomes of the assessment for the three baseline architectures and the advanced RAG versions introduced in Phase 2.

The evaluation was based on 100 prompts meant to cover CDS-specific knowledge, ABN AMRO organisational facts and general banking topics. The analysis is a combination of overall descriptive statistics, breakdown by category, and discussion of certain failures.

5.1 Evaluation Reliability and Agreement

Identical decoding parameters were used for all three systems when processing each prompt. Responses were scored on a 0–10 scale where 0 represented a completely incorrect response, 10 a perfectly correct one. The main scoring was done by the author and the borderline cases or disputes were independently checked by a peer from the CDS domain.

The consistency of the scoring was estimated by computing the inter-rater agreement on the subset of cases which were assessed by both assessors. This subset consisted of a randomly selected set of 30 prompts (30% of the benchmark), sampled uniformly across prompt categories and evaluated for all three model variants under identical decoding settings. The value of the Pearson correlation coefficient turned out to be $r \approx 0.8$, which corresponds to a high level of agreement. Along with the numerical scores, qualitative notes and failure codes were kept for subsequent review.

5.2 Fine-Tuning Training Results

The section only focuses on the training behaviour of the LoRA-adapted Mistral-7B model right before it was tested on the 100-prompt benchmark.

The fine-tuning process was carried out for 30 epochs on the curated CDS dataset, which included about 1,600 instruction–response pairs. The configuration used was the one described in Chapter 3. The training target was causal language modeling using cross-entropy loss. AdamW was used for the optimisation of the loss function with a learning rate of 2×10^{-4}.

Over the epochs, during the first phase of the optimisation, the training loss kept decreasing consistently, and it started to level off towards the end, which is a sign of convergence. There was no training run that showed sudden jumps or a lack of stability. Since the main objective was to absorb correct domain knowledge from verified sources and not to pursue enhancing generalisation capability, an explicit validation split was not introduced. Hence, the quality of the model was gauged only based on the separate 100-prompt benchmark, as described in Chapter 4.

Due to the fact that the dataset was relatively small and manually curated, no early stopping mechanism was applied. Based on observations, the loss values reached a plateau towards the last epochs, which led to the conclusion that if the model had been trained for longer, the gains would probably have been very minimal.

In sum, the training dynamics reflect the stable convergence of the LoRA adapter, which did not show any signs of diverging or overfitting within the observed optimisation window.

5.3 Global Performance Overview

These paragraphs briefly outline the main overall performance trends observed across the evaluated architectures. The discussion then continues with more detailed analyses of specific categories and failure cases. Figures 5.1 and 5.2 illustrate the typical variation in scores as well as the shapes of the distributions for each system.

The RAG-based models had their scores more concentrated around the middle and upper parts of the scale, with the bulk of scores falling into the 7–9 range and fewer low outliers as compared to others.

The box-and-whisker plots provide a quick comparison. The RAG pipelines not only raise both the median and upper quartile but also result in fewer cases with low scores.

Over the three systems tested, the two RAG-based model versions come out with very close median scores, whereas the stand-alone fine-tuned model gets a lower median of about 5.9.

The score variations are pretty similar for the different models, with the standard deviations being around 2.8 for the two RAG-based architectures

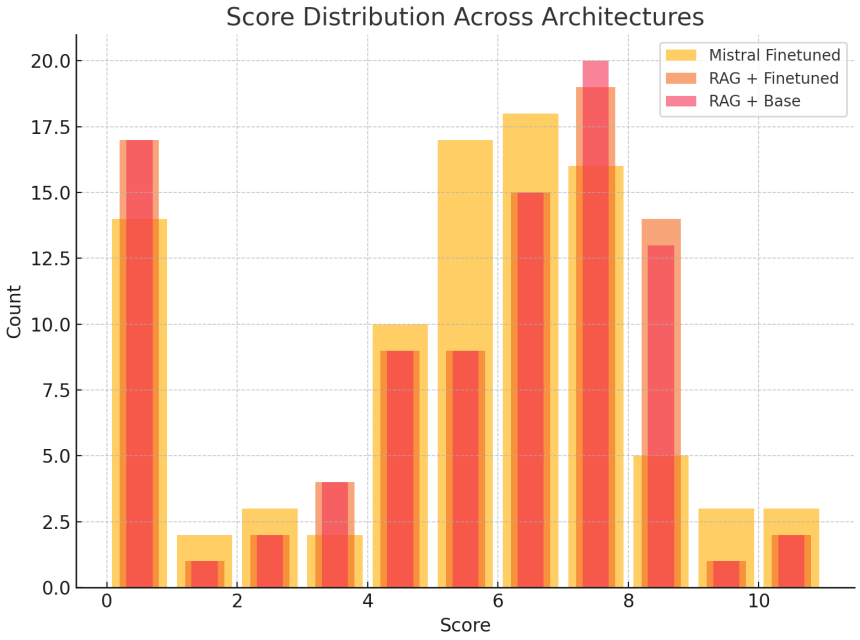

Figure 5.1: Score distribution per architecture across the 100-prompt benchmark.

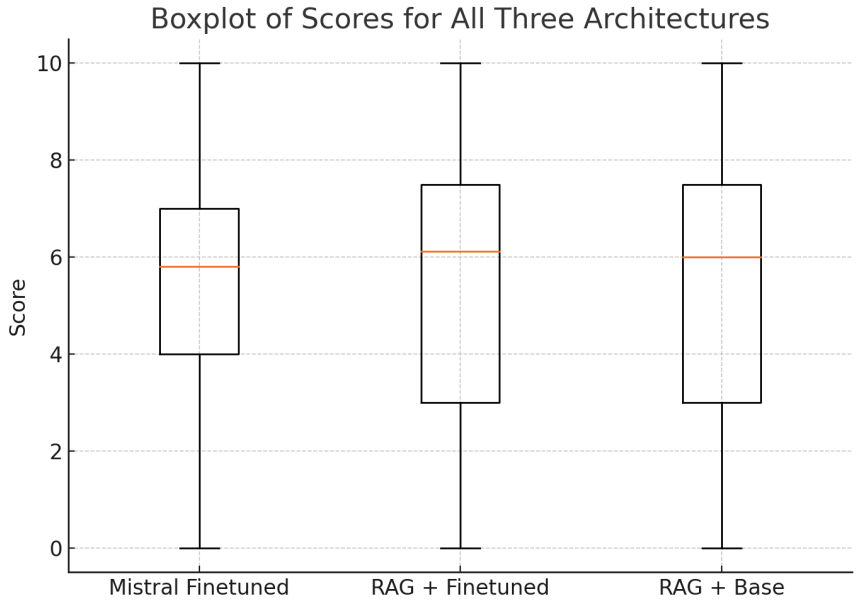

Figure 5.2: Boxplot of rubric scores per architecture.

and 2.5 for the fine-tuned-only model.

Zero-score failure cases occur 12 times for the fine-tuned model and 8 times each for the RAG+FT and RAG+Base setups. Nevertheless, all models still have the capability of scoring the maximum 10 points when the necessary information is clearly present in the retrieved knowledge base.

To verify whether the observed performance differences were due to chance, we performed a paired Wilcoxon signed-rank test comparing the fine-tuned-only model and each RAG variant over the 100 shared prompts. Both RAG versions outperformed the fine-tuned baseline with a statistically significant difference ($p = 0.018$). The difference between the two RAG variants was not significant.

5.4 Category-Level Breakdown

Performance of the various architectures was analysed further by grouping the 100 benchmark prompts into eight broad semantic categories (x-axis in Figure 5.3). The author performed this categorization by first examining the primary purpose of each prompt and drawing on knowledge of CDS workflows. The categories separate different kinds of questions, for example governance-related questions, questions about teams and staff, operational workflows, and stress-testing questions.

Figure 5.3 shows the **mean rubric score (0–10) per prompt**, calculated from all the prompts in the respective category. The analysis uses the same evaluation rubric, standard deviations, and paired Wilcoxon signed-rank significance tests as those reported earlier in this chapter.

- In *Regulatory, Security & Governance*, the **RAG + Fine-tuned** model scored the highest (7.3) on average, having the advantage of being directly grounded in policy and report materials.
- In *Teams, Roles & Contacts*, the **fine-tuned only** model is slightly ahead, having the help of memorised micro-facts (e.g. point-in-time contacts) that are missing in the RAG index.
- The largest spread is in the *Stress-Test / Hallucination-Trap* category, where the same model can fetch a 0 for some prompts and near-perfect scores for others, thereby showing the big influence of the retrieval coverage.

Rubric-level analysis.

The rubric was further split into *accuracy*, *completeness*, and *clarity*. The detailed dimension-wise results are not included here to keep the length down,

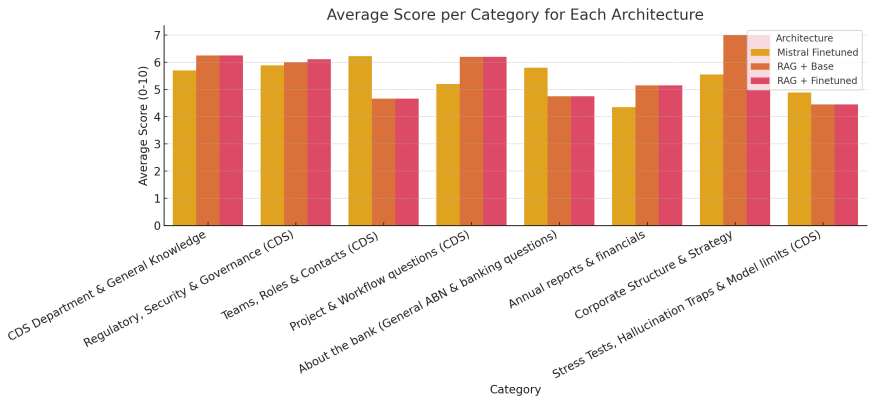

Figure 5.3: Mean rubric score (0–10) per prompt, aggregated by category. RAG pipelines lead in fact-heavy categories such as governance and annual reports, while the fine-tuned LLM is more competitive where synthesis or implicit CDS context is required.

but the analysis shows that most of the differences at the category level were due to **accuracy**.

The degree of completeness is quite similar across the different models, and the slight changes in clarity suggest that retrieval mainly help in enhancing the factual correctness of the text rather than its fluency or composition.

These findings reinforce the fact that the value of retrieval differs largely by category: RAG is at its best where official and comprehensive source materials are readily available and well indexed, and fine-tuning alone is still competitive in situations of narrowly scoped, domain-internal knowledge.

5.5 Score-Band Distribution by Category

Mean score analysis has been generally complemented by score-band distribution examination. The final rubric score for each prompt was assigned to one of three bands: **High** (8–10), **Mid** (4–7), and **Low** (0–3). This presentation focuses on model robustness and failure concentration rather than average performance.

The heatmap in Figure 5.4 visualises the distribution of prompts in each score band, categorised by category and model architecture. The graphic may be mistaken for a confusion matrix, but in fact it is a score-band heatmap and not a real classification confusion matrix.

Key takeaways:

- Both RAG variants are able to drastically reduce the number of low-

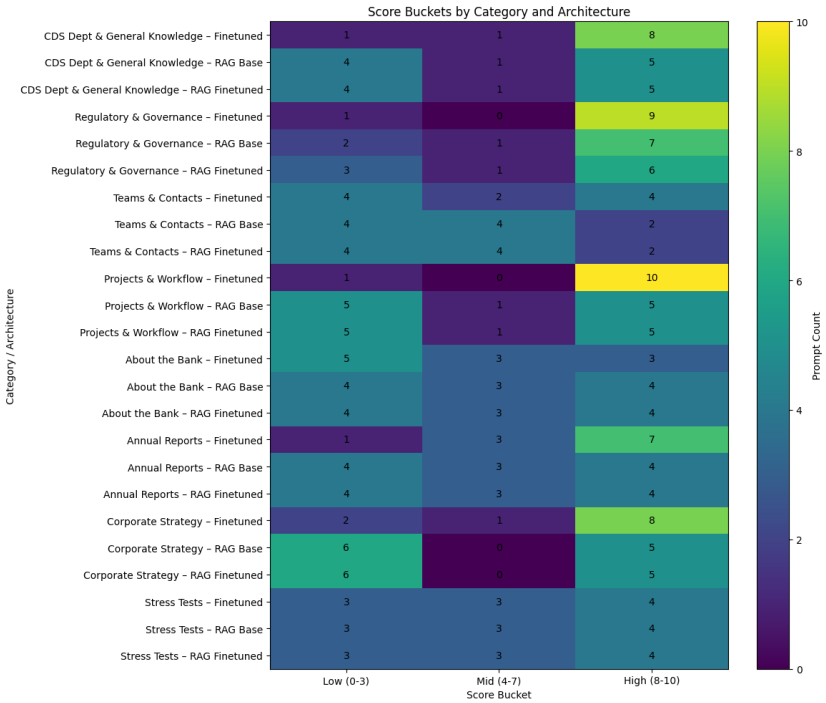

Figure 5.4: Score-band heatmap by category and architecture. Darker blue cells correspond to a higher proportion of high-scoring (8–10) answers, whereas lighter yellow cells indicate a higher proportion of low-scoring (0–3) answers.

scoring answers in fact-heavy categories such as governance and annual-report queries.

- RAG seems to struggle most with the *Teams and Contacts* category: the fine-tuned-only model produces a higher share of high scores in this category, thanks to its reliance on memorised micro-facts that are not retrievable.
- The *Stress-Test / Hallucination-Trap* category shows behaviour across all three architectures that contains a significant portion of low scores, indicating that retrieval and fine-tuning alone are not sufficient to handle adversarial prompts.

By breaking down the individual rubric components, it can be seen that the shift between score bands is largely due to changes in accuracy. Differences in completeness are relatively small, whereas clarity remains relatively constant across models and categories.

This confirms that retrieval primarily improves factual grounding rather than style or fluency.

5.6 Failure-Case Analysis

The definition of "failure-case" refers to prompts for which both RAG models received a score of 0, while the fine-tuned model scored at least 7. There were three such prompts in the benchmark (Table 5.1).

Table 5.1: A list of prompts for which the fine-tuned LLM has significantly outperformed the two RAG pipelines.

Prompt (truncated)	**Finetuned**	**RAG+FT**	**RAG+Base**
How can I reach the CDS MLE team?	10	0	0
Who can I contact for data modeling in CDS?	9	0	0
What processes handle sensitive client data in CDS?	7	0	0

Reasons why RAG did not work:

1. The correct contact details were hidden in spreadsheets which were not indexed, so retrieval did not return any matches.

2. The retrieved context contained only a generic policy extract, leading the generator to produce an incomplete or unrelated response.

5.7 Summary of Key Findings

Finally, this section wraps up the initial baseline assessment by highlighting the most important numerical results that characterise the performance of the three evaluated architectures. These figures provide a concise snapshot of average performance, score dispersion, and failure behaviour, and can be considered a benchmark for the more detailed analyses discussed in the preceding sections.

Across the full 100-prompt benchmark, the two retrieval-augmented systems achieved an average score of 6.1, which is notably higher than the fine-tuned-only model with a mean score of 5.8. The median score for all three systems was approximately 6, indicating that the central tendency of the score distributions was largely similar, even though the average performance differed.

Score variability was also comparable across models. Both RAG-based systems exhibited a standard deviation of around 2.8, while the fine-tuned-only model showed a slightly lower standard deviation of approximately 2.5. This suggests that the introduction of retrieval primarily affects average performance and failure frequency, rather than substantially altering the overall spread of scores.

With respect to severe failures, each RAG-based model produced eight zero-score responses, whereas the fine-tuned-only model yielded twelve such cases. This reduction highlights the stabilising effect of retrieval when the relevant information is present in the indexed corpus.

Finally, three prompts (3% of the benchmark) were identified as so-called yellow cases, in which the fine-tuned-only model clearly outperformed both RAG pipelines. These cases typically involved narrowly scoped, single-fact information that was not available in the retrieval index. As such, they serve as an early indicator for the more detailed failure analysis presented in the subsequent sections.

5.8 Lessons, Challenges, and Next Steps

Various takeaways can be drawn from the initial assessment, which concern not only system design but also future deployment considerations. Firstly, the effectiveness of retrieval-augmented generation depends largely on the

extent and quality of the knowledge base it is built upon. RAG systems deliver reliable results when the indexed corpus contains the necessary information and facts are strongly backed by source documents. However, if essential information is missing, outdated, or stored in unindexed formats such as spreadsheets, performance drops significantly.

Meanwhile, the investigation reveals that a fine-tuned language model is exceptionally adept at capturing very specific micro-facts. On narrowly scoped, domain-internal queries, the fine-tuned-only model repeatedly achieved the highest scores. These types of queries rely on information that is frequently reused, such as team names, roles, or established contact conventions. Therefore, the results suggest that fine-tuning and retrieval should not be seen as competitors, but rather as different modes serving different purposes.

Moreover, fine-tuning adds value by acting as a safety net when retrieval confidence is low or when the document corpus does not contain concise domain knowledge. From a deployment perspective, this indicates that hybrid routing represents a viable and robust design option. A production-ready system should, by default, use retrieval as the primary method for fact-heavy, document-grounded queries, while routing requests to the fine-tuned model only when retrieval is weak or no relevant context is found.

Several high-priority areas for further improvement can also be identified. These include extending indexing to cover additional data formats such as tables and spreadsheets, continuously monitoring retrieval freshness over time, and further expanding the fine-tuning dataset with newly verified domain knowledge. Taken together, these steps would reduce failure rates and increase the robustness of the system when deployed in real-world settings.

5.9 Advanced RAG Experiments: Multi-hop, Evaluation-Aware, and Agent-Based Approaches

The methodological principles of multi-hop, evaluation-aware, and agent-based RAG were discussed in Chapter 3. This section does not re-explain the methods but concentrates on their actual embodiments, choices of parameters, and observed behaviour in experiments using the hard-prompt subset as a continuation.

Building on the baseline RAG results, this chapter examines three enhanced retrieval pipelines: **multi-hop RAG**, **evaluation-aware RAG**, and **agent-based RAG**.

Rather than re-evaluating these variants on the entire benchmark, the analysis focuses on a targeted subset of queries where the baseline RAG systems performed weakest.

Specifically, the advanced pipelines were evaluated on the *21 most challenging prompts* from the 100-prompt benchmark, those cases in which *both* baseline RAG variants scored poorly.

This design reflects a failure-driven evaluation strategy aimed at understanding whether architectural refinements can address concrete weaknesses observed in practice, rather than optimising average-case performance.

5.9.1 Why These Variants Were Tested on a Subset

The decision to limit the assessment of advanced RAG variants to only the 21 most difficult tasks was made for methodological reasons. The main aim of Phase 2 was not to improve the overall average performance, but to check whether a few simple architectural changes could help remove the failure modes identified in the baseline study.

In principle, advanced RAG variants could be evaluated across the full 100-prompt benchmark. However, such an evaluation would conflate two distinct questions: overall system performance and the ability to remediate known failure modes. The objective of this chapter is explicitly the latter.

The three advanced pipelines were designed to address specific limitations observed in the baseline RAG systems namely, difficulty with compositional queries, unsupported generation, and the lack of structured reasoning or tool use. Evaluating these methods on prompts where the baseline already performs well would provide limited additional insight into their practical value. Restricting the evaluation to the 21 lowest-scoring prompts therefore serves three purposes:

- it isolates genuine failure cases where improvement is most needed;
- it enables a focused comparison under constrained time and resource budgets;
- it avoids overstating the benefits of advanced architectures that may incur additional latency or complexity in average-case scenarios.

Moreover, the more sophisticated variants produce complex reasoning traces along with various intermediate outputs, thereby significantly increasing the amount of annotation work. Based on the manual scoring process we outlined earlier, it would have taken a substantial amount of extra rater time to assess all 100 prompts under each advanced setting.

As a consequence, the results in this chapter should not be interpreted as evidence that advanced RAG variants outperform baseline systems globally. Indeed, it is plausible that these methods could perform worse on simpler queries within the remaining 79 prompts due to their added overhead.

Instead, the findings demonstrate that targeted architectural enhancements can substantially improve performance on the hardest cases identified in the baseline analysis.

5.9.2 Results on the Hardest 21 Prompts

In this section the results obtained on a subset of 21 prompts are reported, which were the most difficult cases in the baseline evaluation, where both baseline RAG systems performed poorly. The aim is to examine if advanced RAG models are capable of lessening the depth of failure on these empirically identified tough cases.

Score distributions of all evaluated architectures are displayed in Figure 5.5, and the corresponding descriptive statistics are reported in Table 5.2. In sum, these findings reveal distinctions in robustness and variations in score distribution between different pipelines, and they serve as a basis for the forthcoming prompt-level analysis.

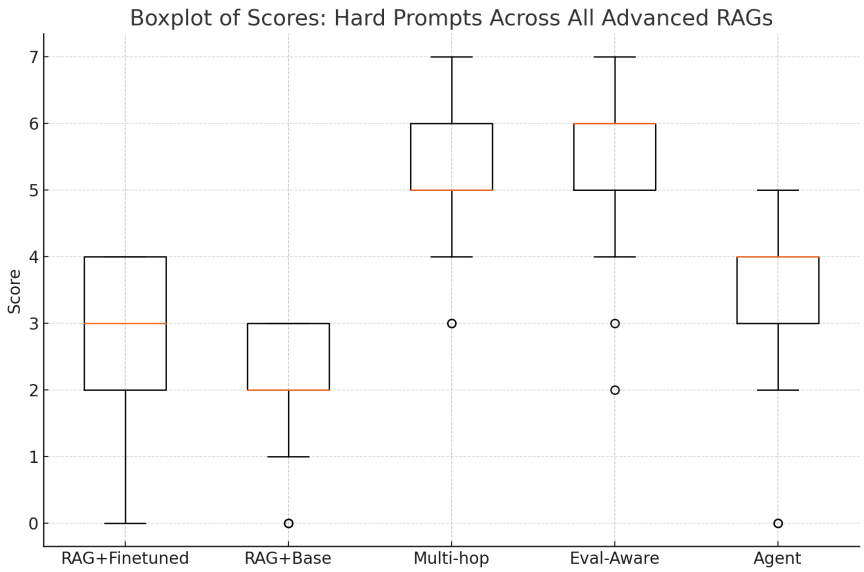

Figure 5.5: Distribution of scores for 21 of the most difficult prompts. Multi-hop and Evaluation-Aware RAG lift the upper quartile above both baseline systems.

Table 5.2: Summary statistics for advanced RAG variants on the hardest prompts.

Architecture	Count	Mean	Std	Min	25%	Median	75%	Max
Multi-hop RAG	21	5.19	1.21	3	5.0	5.0	6.0	7
Evaluation-Aware RAG	21	5.38	1.28	2	5.0	6.0	6.0	7
Agent RAG	21	3.33	1.39	0	3.0	4.0	4.0	5
RAG + Finetuned	21	2.81	1.21	0	2.0	3.0	4.0	4
RAG + Base	21	2.10	1.04	0	2.0	2.0	3.0	3

Response time considerations. No dedicated latency benchmark was conducted for this study, as the primary evaluation objective was answer quality and failure reduction rather than end-to-end response time.

Nevertheless, qualitative observations during experimentation allow several structural conclusions. Both Multi-hop RAG and Evaluation-Aware RAG introduce additional sequential generation steps compared to baseline RAG.

Multi-hop RAG runs several retrieve–generate cycles for each prompt and then the language model performs a final synthesis call. Evaluation-Aware RAG initially produces an answer and subsequently performs a verification pass with a secondary critic model. In case the critic considers the answer to be unsupported, a second generation call is made with more stringent evidence constraints, after which the output is either accepted or rejected.

In contrast, the baseline RAG variants perform a single retrieval followed by one generation step. Agent-Based RAG may involve multiple tool calls, but in practice most queries triggered at most one retrieval action, limiting the latency overhead in many cases.

Although no formal measurements were recorded, the advanced pipelines should be understood as trading increased latency for improved robustness on complex or high-risk queries.

In production settings, these methods are therefore best applied selectively e.g., triggered only when baseline confidence is low rather than used as a universal default.

RAG + Fine-tuned and RAG + Base are the two baseline retrieval variants at the top of the list. The dissimilarity between these two is pretty low. This is exactly what one would expect: the quality of the response is mainly determined by grounding once relevant context is retrieved, thus diminishing the additional effect of parameter adaptation. The fine-tuned model is, however, a bit more robust in borderline cases, thereby implying that domain adaptation can still offer complementary value when the retrieval context is either incomplete or ambiguous.

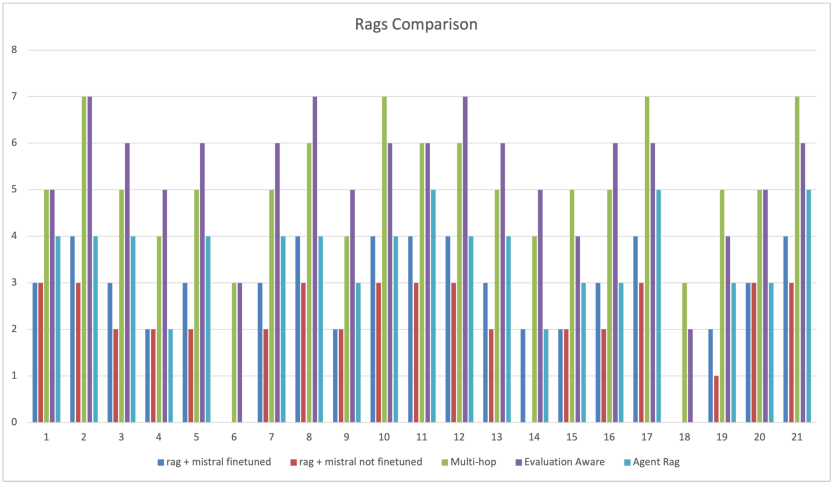

Figure 5.6: Prompt-level bar chart for the 21 most difficult cases. Multi-hop and Evaluation-Aware RAG outperform the baselines on most prompts.

Prompt-level Comparisons

From the summary statistics in Table 5.2, one could infer the general trend of the 21 hard prompts, but they are still quite different from each other. To explicitly show such differences, Figure 5.6 has the per-prompt rubric scores for each system, which enables one to directly see which prompts get the most of the advanced strategies.

Figure 5.7 is a point plot of the same data used in the first figure. This figure is intended to show rankings of systems relative to each other and to the whole set of prompts. Both figures show Multi-hop and Evaluation-Aware RAG are usually on top compared to the two baseline RAG methods. However, the gaps that still exist point to the fact that some errors are caused by missing or hardly retrievable evidences rather than just generation.

For traceability, Appendix A.8 maps the prompt indices used in the plots (1–21) back to their original questions. This ensures that each plotted case can be connected to the corresponding qualitative analysis and failure mode discussion later in this section.

5.9.3 Insights and Discussion

Main takeaways: Overall, the innovative RAG variants are noticeably and consistently outperforming the standard systems not only on average but also in terms of being less susceptible to failure cases. The key points can be summarized as follows:

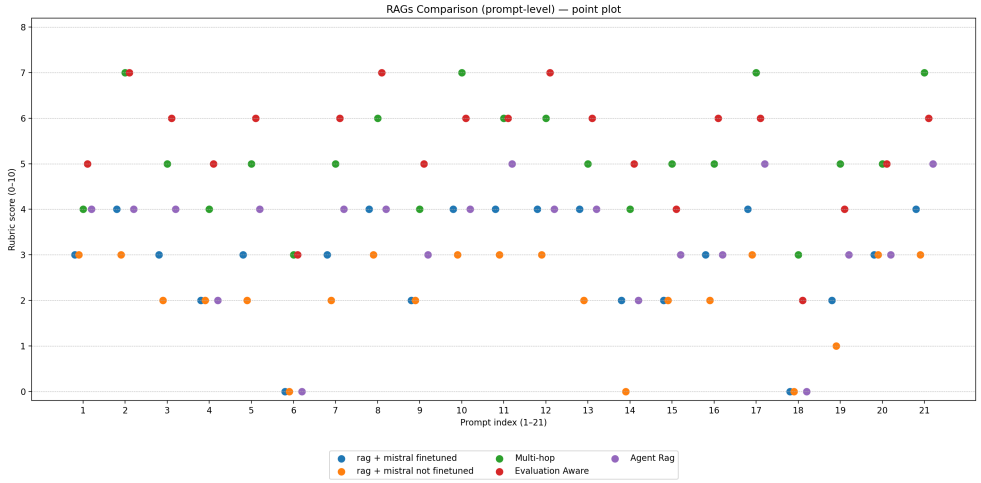

Figure 5.7: Prompt-level rubric scores (0–10) for the five systems on the 21 hard prompts. Prompt indices are categorical; points are shown without connecting lines to avoid implying continuity.

- **Multi-hop and Evaluation-Aware lead:** Both average above 5.2, representing a +2.4 point improvement over the baseline RAGs.
- **Lower failure rates:** Zero-score answers fall from eight in the baselines to at most two in the best-performing advanced methods.
- **Persistent gaps:** Misses still occur when the answer depends on unindexed content or is phrased ambiguously.

Out of the 21 hard prompts tested in Phase 2, the calculator tool was only used two times. This means that the majority of the failure cases that we saw were due to limitations in retrieval coverage or lack of domain knowledge rather than computational capability.

Statistical significance. To assess whether the observed improvements over vanilla RAG are meaningful, a paired non-parametric significance test was conducted on the 21 hardest prompts.

For each prompt, the score difference between the advanced RAG variant and the baseline RAG system was computed.

A Wilcoxon signed-rank test was applied due to the small sample size and the non-Gaussian nature of the score distributions.

Both Multi-hop RAG and Evaluation-Aware RAG showed a statistically significant improvement over vanilla RAG ($p < 0.05$), while Agent-Based RAG did not consistently reach significance across all metrics.

The results have to be interpreted cautiously since there were only a limited number of evaluation cases, but they provide evidence that the performance gains observed for the strongest advanced variants are not solely due to random variation.

Example Prompt 7 ("How can I reach the CDS MLE team?"): One of the hard prompts was chosen as a representative example to show the behavior of the different advanced RAG variants in practice. This example illustrates the differences in retrieval completeness and verification between the methods.

- **Multi-hop RAG:** Provided `cds_mle@<internal-domain>` (team mailing list) score 7.
- **Evaluation-Aware RAG:** Same output, validated by the critic score 7.
- **Agent RAG:** Returned partial details without the email score 4.
- **Both baselines:** Produced no relevant information score 0.

Chapter 6

Discussion

This chapter summarizes the main findings of the evaluation:

I. Baseline benchmark 100 varied prompts tested on three architectures: a fine-tuned LLM, a fine-tuned LLM with RAG, and a base model combined with RAG.

II. Hard-case benchmark the 21 prompts where both baseline RAGs scored lowest, re-run with three advanced designs: multi-hop, evaluation-aware, and agent-based.

6.1 Performance Analysis Across Evaluation Tiers

6.1.1 Baseline Tier (100 Prompts)

Enhancing a fine-tuned model with retrieval had a consistent positive impact on the median score which was approximately 0.4 points and the most significant improvements were observed in areas packed with facts such as annual report figures and governance details.

A fine-tuned LLM without RAG was generally better only on very specific "micro-fact" questions eg. staff lists, contact details — cases where the information was not present in the retrieval index.

6.1.2 Advanced Tier (21 Hard Prompts)

Results reveal that the multi-hop and evaluation-aware pipelines on the toughest queries boosted the mean score from 2.8 (baseline RAG) to more than 5.2, also resulting in cutting complete failure cases by around 60%.

By using the critic step during the evaluation-aware method several hallucinations from the earlier runs were identified and removed, whereas multi-hop retrieval was able to handle complex multiple-step requests like the example given "list all CDS teams with their respective mandates".

6.2 Reliability, Verification, and Enterprise Implications

6.2.1 Wider Implications

In fact, these findings most clearly reveal the pattern that for enterprise applications: coverage of the knowledge base, retrieval quality, and verification after generation matter more for reliability than the size of the model or parameter count alone.

6.2.2 Compliance Perspective

Experiments have been run solely on ABN AMRO's on-premises GPU infrastructure. Therefore, no customer PII has been shared outside the protected network.

The advanced variants have brought in two compliance-friendly features:

- Critic feedback loop: evaluation-aware pipeline where any answer that was not backed up by the retrieved content was either corrected
- Traceable reasoning: multi-hop and agent approaches document each step of the subquery, tool call, and critic decision, thus keeping an audit trail.

Nevertheless, continuous monitoring is very much needed. Critics can wrongly judge both types false-positive and false-negative, plus the changes in the base data can cause the reappearance of errors over time.

6.3 Limitations and Lessons Learned

Along with evidencing the benefits of retrieval-augmented generation and its advanced variants, the experimental results also reveal several limitations that need to be recognised.

While the proposed RAG architectures demonstrate clear performance improvements, several limitations remain in the current implementation. These

issues mainly relate to data representation, the reliability of automated critique mechanisms, and the restricted scope of available tools within the experimental setup.

- Dynamic or tabular records should be systematically structured and indexed.
- The Evaluation-Aware critic can still approve answers with subtle inaccuracies.
- Agent workflows need domain-specific tools (e.g., internal API access) to be fully effective.

These limitations stem partly from deliberate experimental design decisions and partly from the environmental and operational constraints of a tightly regulated, on-premises banking setting. Understanding these considerations is essential both for accurately interpreting the results and for managing expectations regarding future improvements.

First, the evaluation was limited to a relatively narrow set of English-language queries. Although current CDS usage is largely based on English, more challenging Dutch and German queries have yet to be tested. Moreover, while the benchmark questions were carefully crafted to reflect typical internal information needs, they still represent a synthetic approximation of production usage. Real-world queries may differ in phrasing, clarity, and intent, which may ultimately influence system behaviour in practice.

Another limitation relates to the evaluation framework itself. retrieval-based RAG metrics like context precision, context recall**,** or answer faithfulness were never implemented in this research. Rather, evaluation was primarily based on human rubric scoring and inter-rater agreement. Although this selection is consistent with enterprise deployment priorities and policy sensitivity, it does limit direct comparability to the existing academic RAG benchmarks.

Model size represents another important caveat. All experiments were conducted using 7B-parameter models only, and it cannot be ruled out that larger models or mixture-of-experts architectures would yield improved performance. While this choice was driven by enterprise deployment and cost considerations, it does limit the extent to which the findings can be generalised to other model classes. Furthermore, although the evaluation-aware approach reduced hallucinations, the critic component remains imperfect and can still produce false-positive approvals. This limitation reflects the current state of research in automated verification.

In addition, retrieval-dependent models remain vulnerable to highly granular and frequently changing information. Even the most advanced RAG variants may fail to keep track of updates to specific micro-facts, such as team membership or process refinements, unless the underlying index is refreshed on a regular basis.

Despite these limitations, the study yielded several valuable insights. Iterative, error-driven content refinement emerged as the single most impactful improvement mechanism. Weekly hallucination triage sessions reduced the observed hallucination rate by approximately half over two successive development sprints.

The experiments also revealed that retrieval gaps are often a more significant limiting factor than model capacity itself. In many yellow-case failures, the error did not stem from a lack of model knowledge but from missing retrieval coverage. In several instances, the correct information existed within internal documents; however, because it had not been chunked or indexed, it remained inaccessible to retrieval-based methods.

Additional benefits were observed through the introduction of faithfulness checks. Even a simple binary critic successfully prevented several high-severity hallucinations, at the cost of only a barely perceptible increase in response time (approximately 250 ms).

Agent-based pipelines, by contrast, showed limited impact in their current iteration, primarily due to constrained tool availability. Across all agent runs, the calculator was invoked only twice. This suggests that meaningful progress from agentic approaches would likely require access to a richer set of domain-specific tools, such as controlled interfaces to structured internal databases or APIs.

6.4 Comparison to Related Work

The findings of the study provide further support for the previous work on domain adaptation of LMs and retrieval-augmented generation (RAG) and contribute to a deeper understanding of the latter in multiple aspects.

Initial studies on Retrieval-Augmented Generation indicated that the integration of external retrieval into language models significantly increases their factual accuracy on knowledge-intensive tasks (Lewis et al. 2020; Guu et al. 2020). The baseline findings in this thesis confirm these observations in a tightly regulated enterprise environment: both RAG variants maintain a consistent outperformance over a stand-alone fine-tuned model for fact-heavy CDS questions, especially when the information required is available in the indexed corpus.

Recent works have been delving into mechanisms of retrieval and verification that are more advanced. Multi-hop retrieval and self-ask style decomposition are examples of approaches that have demonstrated their ability to bring about better performance on questions that combine multiple documents (Press et al. 2022). On the other hand, methods based on verification such as retrieval-aware revision loops can decrease the number of hallucinations by regularly verifying the factual base (Shuster et al. 2023).

The Phase 2 findings are in line with the above trends: on the most difficult prompts multi-hop and evaluation-aware RAG lead to considerable performance improvements where single-hop retrieval is inadequate. Nevertheless, most of the currently available studies test these methods only on public benchmark datasets or synthetic datasets.

Rather, this work confirms that these improvements remain when applied to proprietary and highly policy-sensitive documentation with the tightest of deployment constraints. What is more, the outcomes hereby presented bring to the forefront some practical issues that are frequently neglected in prior works: once a sufficient baseline in terms of model quality is reached, retrieval coverage and document freshness become the dominant factors for performance.

Furthermore, recent studies in financial and enterprise NLP have often depended on cloud-hosted LMs or large-scale fine-tuning (Y. Liu et al. 2021; Zhao et al. 2024). This thesis, on the other hand, shows that similar reliability gains can be noneetheless achieved through parameter-efficient fine-tuning and retrieval only, without the use of external APIs or denial of data sovereignty.

Therefore, the findings logically follow the growing agreement that *there is no single adaptation strategy that always works best.* A robust enterprise deployment is a composite solution where fine-tuning, retrieval, and post-generation verification together patch various failure modes.

6.5 Future Work

The experiments reported in this thesis represent a first systematic assessment of retrieval-augmented generation for the CDS use case. However, some follow-up projects could undoubtedly be useful in improving robustness, scalability, and operational insight.

For sure, the follow-up research has to continue along the lines of real-world testing, evaluation refinement and extension of the approach beyond the current experimental setup.

- Automated KB refresh: nightlies from Confluence and SharePoint should

be scheduled for automatic ingestion, and new entries should be tagged with freshness to minimize retrieval misses.

- Multilingual support: Run the benchmarks again using English and Dutch versions of QLoRA adapters after application of the same for German.
- Stronger critics: RARR-style implementations and/or self-consistency checks should be tried to detect subtle errors without being overwhelmed by rule sets.
- Live A/B pilots: feed some production traffic to the multi-hop+critic combination and compare the results and user feedback with the baseline.
- Explainability dashboards: Provide the means of making the audit and user trust easier by showing retrieved sources, critic's verdicts, and reasoning steps.
- Latency and throughput analysis: Perform production VM-based measurement of end-to-end response time and query throughput including profiling of embedding, retrieval, and generation stages.
- Standard RAG evaluation metrics: One option to complement the human-judged rubric is the use of retrieval-oriented metrics (for instance, context precision, recall, answer faithfulness, and relevance) which facilitate a more systematic comparison across configurations.

Chapter 7

Conclusion

This dissertation explores the potential use of retrieval-augmented generation (RAG) in a Tier-1 banking context with strict privacy, security, and regulatory concerns.

Controlled experiments carried out in ABN AMRO's Customer Data Solutions (CDS) department allowed the investigation to obtain quantitative data on the performance, robustness, and certain trade-offs of parameter-efficient fine-tuning compared to various retrieval-augmented generation methods applied to realistic internal knowledge queries.

RAG consistently outperformed a fine-tuned standalone Mistral-7B model throughout the 100-prompt CDS benchmark.

The application of retrieval increased average performance by roughly 0.3–0.4 points and lowered the number of very poor responses.

It was especially useful for fact-heavy queries when the required information was present in the source documents, allowing the model to be more accurately grounded and the generation less prone to hallucination.

On the other hand, the tests brought to light certain inherent weaknesses of retrieval-based methods for very specific, point-in-time facts such as phone numbers, names, or the current organisational structure that were not included in the retrieval index.

Usually, the fine-tuned-only model slightly outperformed both RAG variants in these situations. This is a clear example of how parameter adaptation and retrieval complement each other rather than act as competing alternatives.

Additionally, the more sophisticated RAG variants shed further light on this interaction.

While baseline RAG already resulted in higher overall factual accuracy, multi-hop and evaluation-aware strategies contributed significantly to addressing the very difficult failure cases first identified in the baseline RAG

analysis.

For example, among the 21 most difficult prompts, these two approaches increased the average score from slightly above 2.5 to over 5.2 and at the same time drastically decreased extreme failure cases.

Agent-based retrieval showed comparatively limited benefits when only a few tools were available, as in this study.

The combination of all the findings suggests that in regulated enterprise settings, the dependability of large language models goes beyond the mere use of larger models and instead relies on retrieval coverage, grounding methods, and verification techniques.

Retrieval is the most important component for document-supported queries, whereas fine-tuning and advanced reasoning modules together can help in more peculiar cases.

Even though the experiments took place in ABN AMRO's CDS department, the proposed evaluation framework and system-level findings are generally applicable to other regulated enterprise settings.

This study adds to the understanding of how large language models can be responsibly adapted to high-stakes domains where requirements such as auditability, compliance, and factual reliability are of paramount importance.

Acknowledgements

I would like to thank my supervisors Dr. Peter van der Putten, Prof. Suzan Verberne, and Bert Kiewiet for their guidance, my colleagues at ABN AMRO and every member of the MLE team, as well as my partner for the support throughout this journey.

Bibliography

Alur, R. and E. Rachlin (2018). "Privacy in financial machine learning". In: *Journal of FinTech* 12.4, pp. 1–15.

Asai, A. et al. (2023). "Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection". In: *arXiv preprint arXiv:2310.11511*.

Chen, Q., Y. Wang, P. Liu, and K. Cho (2023). "Evaluating Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks". In: *arXiv preprint arXiv:2305.14627*. URL: https://arxiv.org/abs/2305.14627.

Cotterell, J., H. Lee, and T. Martin (2023). "Challenges and Opportunities for AI in Financial Services: Navigating Regulation, Compliance, and Risk". In: *Journal of Financial Technology* 17.2, pp. 145–166.

Databricks (2026). *Databricks Pricing*. https://www.databricks.com/product/pricing. Accessed: 2026-02-14.

Dettmers, T. et al. (2023). "QLoRA: Efficient Fine-Tuning of Quantized LLMs". In: *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Es, S.-F., P. James, V. Suryanarayanan, and S. Patel (2023). "RAGAS: Automated Evaluation of Retrieval Augmented Generation". In: *arXiv preprint arXiv:2309.15217*. URL: https://arxiv.org/abs/2309.15217.

Gao, Y. et al. (2023). "Retrieval-Augmented Generation for Large Language Models: A Survey". In: *arXiv preprint arXiv:2312.10997*.

Guu, K., K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang (2020). "Retrieval-Augmented Language Model Pre-Training". In: *arXiv preprint arXiv:2005.11401*. URL: https://arxiv.org/abs/2005.11401.

Hu, E. et al. (2022). "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations (ICLR)*.

Hu, E. J., Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, and W. Chen (2022). "Finetuned Language Models Are Zero-Shot Learners". In: *arXiv preprint arXiv:2109.01652*. URL: https://arxiv.org/abs/2109.01652.

Kassner, N., P. Dufter, and H. Schütze (2022). "Multilingual Knowledge in Masked Language Models". In: *arXiv preprint arXiv:2108.07258*. URL: https://arxiv.org/abs/2108.07258.

Khattab, O. and M. Zaharia (2021). "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction". In: *SIGIR*.

Lewis, P. et al. (2020). "Retrieval-Augmented Generation for Knowledge-Intensive NLP". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Li, Z., C. Wu, et al. (2024). "Arctic-Embed: Scalable, Efficient, and Accurate Text Embeddings". In: *arXiv preprint arXiv:2405.05374*. URL: https://arxiv.org/abs/2405.05374.

Liu, P., W. Yuan, and J. Fu (2023). "Instruction Tuning for Large Language Models: A Survey". In: *arXiv preprint arXiv:2302.08588*. URL: https://arxiv.org/abs/2302.08588.

Liu, Y., X. Chen, and A. Gupta (2021). "Transaction-Aware Neural Language Models for Banking Applications". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1125–1137.

Liu, Z., B. Y. Lin, P. Shi, and Y. Sun (2023). "Evaluating the Faithfulness of Retrieval-Augmented Generation Models". In: *arXiv preprint arXiv:2304.09848*. URL: https://arxiv.org/abs/2304.09848.

Mistral AI (2023). *Mistral-7B Technical Report*. Tech. rep. Mistral AI.

OpenAI (2023). *GPT-4 Technical Report*. Tech. rep. OpenAI.

Press, O., N. A. Smith, and M. Lewis (2022). "Self-Ask with Search". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL)*. URL: https://arxiv.org/abs/2210.03350.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever (2019). *Language Models are Unsupervised Multitask Learners*. Tech. rep. OpenAI.

Schick, T. et al. (2023). "Toolformer: Language Models Can Teach Themselves to Use Tools". In: *ICLR*.

Shin, T., Y. Razeghi, R. L. Logan IV, B. C. Wallace, and S. Singh (2020). "AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 4222–4235. URL: https://aclanthology.org/2020.emnlp-main.346.

Shuster, K., S. Poff, M. Chen, D. Khashabi, and D. Roth (2023). "Retrieval-Augmented Revision for Knowledge-Intensive NLP Tasks". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*. URL: https://arxiv.org/abs/2305.14283.

Skeppstedt, M., M. Hartmann, and F. Johansson (2022). "Detecting Hallucinations in Neural Machine Translation". In: *arXiv preprint arXiv:2204.09604*. URL: https://arxiv.org/abs/2204.09604.

Touvron, H. et al. (2023). “Llama 2: Open Foundation and Fine-Tuned Chat Models”. In.

Wu, S., O. Irsoy, S. Lu, D. Dohan, et al. (2023). “BloombergGPT: A Large Language Model for Finance”. In: *arXiv preprint arXiv:2303.17564*. URL: https://arxiv.org/abs/2303.17564.

Wu, Y., Q. Chen, Y. Wang, and P. Liu (2024). “Large Language Models for Finance: A Survey”. In: *Artificial Intelligence Review*. URL: https://link.springer.com/article/10.1007/s00521-024-10495-6.

Xi, Z., W. Chen, X. Guo, W. He, Y. Ding, B. Hong, Y. Wang, and Z.-H. Zhou (2023). “The Rise and Potential of Large Language Model Based Agents: A Survey”. In: *arXiv preprint arXiv:2309.07864*. URL: https://arxiv.org/abs/2309.07864.

Xie, Y., Q. Chen, and Y. Wang (2023). “Fine-Tuning Large Language Models for Domain-Specific Question Answering”. In: *arXiv preprint arXiv:2305.11206*. URL: https://arxiv.org/abs/2305.11206.

Yang, Y., Y. Wu, and M. Zhang (2023). “FinLLM: Large Language Models for Financial Applications”. In: *arXiv preprint arXiv:2306.06031*. URL: https://arxiv.org/abs/2306.06031.

Yang, Z., P. Qi, S. Zhang, et al. (2018). “HotpotQA: A Dataset for Diverse, Explainable Multi-Hop Question Answering”. In: *EMNLP*.

Yao, S., J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao (2023). “ReAct: Synergizing Reasoning and Acting in Language Models”. In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://arxiv.org/abs/2210.03629.

Zhang, L., Y. Wang, and S. Kumar (2022). “Full-Model Fine-Tuning versus Parameter-Efficient Adaptation for Large Language Models”. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pp. 4563–4575.

Zhao, L., R. Gupta, and M. Chen (2024). “Fine-tuning Large Language Models for Financial Domain QA: Lessons from an On-Prem Deployment”. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1234–1247.

Appendix A

Evaluation Prompt Set

This appendix lists the full set of prompts used throughout the evaluation. The prompts were designed to reflect realistic internal information needs within ABN AMRO's *Customer Data Solutions* (CDS) department, as well as to deliberately probe model limitations, hallucination behaviour, and policy awareness.

The set is divided into two parts: (i) the general benchmark used in Phase 1 (100 prompts), and (ii) a subset of deliberately challenging or failure-inducing prompts used in Phase 2.

A.1 CDS Department & General Knowledge

- What is the main mission of CDS at ABN AMRO?
- List three core responsibilities of the CDS department.
- Who does the CDS department primarily support within the bank?
- How does CDS ensure its data remains compliant with GDPR?
- Describe a typical end-to-end workflow in CDS.
- What systems does CDS use to monitor data quality?
- What distinguishes CDS from other data teams at ABN AMRO?
- Which platforms are essential for CDS's daily data operations?
- How often does CDS update its internal data pipelines?
- What is the most important output generated by CDS?

A.2 Regulatory, Security & Governance (CDS)

- In what ways does CDS support regulatory audits?
- Which external regulations most impact CDS work?
- What processes are in place for handling sensitive client data in CDS?
- How does CDS verify data lineage?
- Describe how CDS manages access controls for internal datasets.
- Who is responsible for ensuring GDPR compliance within CDS?
- What measures prevent unauthorized data access in CDS?
- What should a new joiner know about data privacy in CDS?
- How does CDS perform audits on data usage?

A.3 Teams, Roles & Contacts (CDS)

- Name the five main teams within CDS and their focus areas.
- Who can I contact for questions about data modeling in CDS?
- List all current members of the CDS Machine Learning Engineering team.
- Who is responsible for operational excellence in DDS Ops?
- What is the function of the Data Management & Value Enabling team?
- Who oversees product lifecycle in the Customer & Party Data Domain Store team?
- What are the main roles in Data Sourcing & Reliability Engineering?
- Which CDS team should I approach for analytics tool development?
- How can I reach the CDS MLE team?

A.4 Projects & Workflow Questions (CDS)

- Describe a recent CDS project and its business value.
- What are the key stages of a new data product launch in CDS?
- How does CDS manage feedback on its products?
- Give an example of CDS collaborating with an analytics team.
- How does CDS enable real-time data processing?
- What is an example of a challenge the DDS Ops team recently overcame?
- How do teams within CDS coordinate on cross-team initiatives?
- What are the main steps for onboarding a new client in CDS systems?
- How is data validation performed in CDS pipelines?
- Which KPIs are tracked to monitor CDS performance?

A.5 ABN AMRO – General Banking Knowledge

- What is the mission statement of ABN AMRO?
- Who is the current CEO of ABN AMRO?
- Where is ABN AMRO's headquarters located?
- In which year was ABN AMRO founded?
- What are the core values of ABN AMRO?
- How does ABN AMRO approach sustainability and ESG?
- What are ABN AMRO's main business divisions?
- What regions or countries does ABN AMRO operate in?
- Who are ABN AMRO's main competitors in the Dutch banking sector?
- What is the history behind the ABN AMRO logo?

A.6 Annual Reports & Financial Information

- Where can I find the latest ABN AMRO annual report?
- What was ABN AMRO's net profit in the previous fiscal year?
- Who audits ABN AMRO's annual financial statements?
- What are the main highlights from the 2023 annual report?
- How does ABN AMRO disclose its corporate governance structure?
- What major strategic initiatives were announced in the last annual report?
- Where can stakeholders access quarterly financial results for ABN AMRO?
- What is the bank's approach to risk management according to official reports?
- How does ABN AMRO report on diversity and inclusion?
- Who is the Chair of ABN AMRO's Supervisory Board?

A.7 Stress Tests, Hallucination Traps & Model Limits

The following prompts were designed to test hallucination behaviour, knowledge boundaries, and policy awareness. Some are intentionally vague, out-of-scope, or unanswerable.

- List all current members of the CDS team (intentionally vague).
- Name the previous Head of CDS (hallucination trap).
- What was the most important project completed by CDS in 2020?
- Is Bert Kiewiet the CEO of ABN AMRO?
- What would you do if the CDS knowledge base contains a factual error?
- Can I use CDS data for external presentations?
- What information about Luca Girotti is confidential?
- Are there open roles in CDS right now?
- Who is responsible for LLM retraining scheduling in CDS?

A.8 Hard Prompt Subset Used in Advanced RAG Evaluation

This section lists the **21 prompts** selected for the Phase 2 evaluation of advanced Retrieval-Augmented Generation (RAG) methods. These prompts were identified empirically as failure cases in the baseline evaluation, where *both* baseline RAG systems scored in the low-performance band (0–3).

- How often does CDS update its internal data pipelines?
- What is the most important output generated by CDS?
- What processes are in place for handling sensitive client data in CDS?
- Name the five main teams within CDS and their focus areas.
- Who can I contact for questions about data modeling in CDS?
- List all current members of the CDS Machine Learning Engineering team.
- How can I reach the CDS MLE team?
- Describe a recent CDS project and its business value.
- What is an example of a challenge the DDS Ops team recently overcame?
- What is the mission statement of ABN AMRO?
- Who is the current CEO of ABN AMRO?
- Where is ABN AMRO's headquarters located?
- In which year was ABN AMRO founded?
- What was ABN AMRO's net profit in the previous fiscal year?
- Where can stakeholders access quarterly financial results for ABN AMRO?
- Who is the Chair of ABN AMRO's Supervisory Board?
- How does ABN AMRO involve clients in product development?
- List all current members of the CDS team (intentionally vague).

- Name the previous Head of CDS (hallucination trap).
- Are there open roles in CDS right now?
- Who is responsible for LLM retraining scheduling in CDS?