Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Enhancing Retrieval-Augmented Generation Capabilities through the Design of a Quote-Extraction Agent

Aprimorando as Capacidades de Geração com Recuperação por Meio do Projeto de um Agente de Extração de Citações

Yuri Façanha Bezerra

Dissertation submitted as a partial requirement for the qualification for the Master's degree in Computer Science

Brasília
2026

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Enhancing Retrieval-Augmented Generation Capabilities through the Design of a Quote-Extraction Agent

Aprimorando as Capacidades de Geração com Recuperação por Meio do Projeto de um Agente de Extração de Citações

Yuri Façanha Bezerra

Dissertation presented as a partial requirement for completion of the Master's degree in Computer Science

Prof. Dr. Li Weigang (Advisor)
CIC/UnB

Prof. Dr. Wallace Anacleto Pinheiro
IME

Prof. Dr. Luís Paulo Faina Garcia
CIC/UnB

Prof. Dr. Pedro Garcia Freitas
CIC/UnB

Prof.ª Dra. Cláudia Nalon
Coordenador do Programa de Pós-graduação em Informática

Brasília, Março , 2026

Dedication

I dedicate this thesis to my parents, who always encouraged me to pursue my goals, and to my wife, who has been a constant source of support during this challenging journey.I dedicate this work first and foremost to God, who opened doors, empowered me, and brought me here according to His will. To my parents and grandparents, who at the beginning of my academic journey provided me with support that I know is not available to everyone. I acknowledge the privilege of never lacking the means necessary to develop myself to the fullest. To my wife Ivna, who is part of me and who suffers and rejoices with me at every stage of this journey. And to my son João Miguel, who was born exactly at the beginning of my master's degree and became a fundamental motivator for me to mature and pursue my goals with renewed determination.

Acknowledgements

First and foremost, I thank God, for from Him and through Him and to Him are all things.

To my wife Ivna, through whom I extend my gratitude to my entire family for understanding the importance of education and for never allowing me to compromise on the effort required for self-improvement.

To Professor Li Weigang, for his patience and persistence in ensuring that excellence was always achieved and surpassed, and to Professors Wallace Anacleto Pinheiro and Luís Paulo Faina Garcia for their valuable contributions to this work.

To my colleagues at TransLab, who provided excellent support and with whom I was able to share experiences, exchange ideas, and collaborate on meaningful work.

To the University of Brasília (UnB), for the opportunity to pursue a master's degree at one of the most important and renowned universities in the country.

To the Brazilian Army, the institution that has invested in my academic formation since my days at Colégio Militar de Fortaleza, through the Instituto Militar de Engenharia, and which allowed me to dedicate myself fully to this master's program and to the present work.

Resumo

Modelos de linguagem de grande porte (LLMs) alcançaram desempenho de referência em uma ampla gama de tarefas de processamento de linguagem natural, incluindo resposta a perguntas, sumarização e geração de diálogos. No entanto, essas capacidades acarretam custos computacionais elevados, latência de inferência significativa e consumo substancial de memória, fatores que limitam a implantação prática desses modelos em ambientes com restrições de recursos ou requisitos de tempo real. A Geração Aumentada por Recuperação (RAG) oferece uma direção promissora para mitigar parte dessas limitações, ao permitir que modelos acessem conhecimento externo em tempo de inferência em vez de codificar todo o conhecimento do mundo em seus próprios parâmetros.

Contudo, os pipelines RAG convencionais introduzem um problema próprio. O estágio de recuperação tipicamente retorna passagens completas com base em sinais de relevância superficiais, inundando a entrada do modelo com conteúdo redundante ou distrator. O modelo generativo precisa, então, simultaneamente filtrar esse ruído e raciocinar sobre a informação relevante, uma carga dupla que aumenta o risco de alucinações e reduz o embasamento factual. Esse problema é particularmente agudo para Modelos de Linguagem Compactos (SLMs), que possuem capacidade limitada para processar contextos longos e ruidosos.

As abordagens existentes apresentam limitações significativas. Frameworks como o RAFT treinam um único modelo para simultaneamente raciocinar sobre documentos recuperados e gerar respostas, exigindo ajuste fino de modelos grandes em dados específicos de domínio. Técnicas de engenharia de prompt podem melhorar o raciocínio sem modificar parâmetros, mas não resolvem o problema do ruído: o modelo ainda recebe o contexto completo e não filtrado. Além disso, a maioria dos trabalhos de otimização de RAG concentra-se em modelos grandes, embora SLMs em cenários sensíveis a latência ou com restrições de custo sejam precisamente os que mais se beneficiariam de entradas mais limpas.

Esta dissertação parte da hipótese de que uma arquitetura modular e desacoplada, na qual um extrator leve seleciona evidências relevantes e um gerador separado raciocina sobre elas, pode superar abordagens monolíticas, especialmente para modelos compactos.

Ao delegar a tarefa de filtragem a um componente especializado treinado por meio de destilação de conhecimento caixa-preta, o gerador é liberado para se concentrar exclusivamente no raciocínio sobre evidências concisas e de alta qualidade. Essa separação de responsabilidades também proporciona benefícios práticos: o extrator pode ser treinado uma única vez e reutilizado com diferentes geradores, sem necessidade de qualquer modificação no componente generativo, tornando o sistema adaptável e escalável para diversos cenários de implantação. Reconhece-se, contudo, que a generalização do extrator para domínios significativamente distintos daqueles vistos durante o treinamento permanece uma questão em aberto: a eficácia da filtragem pode degradar ao longo do tempo ou quando aplicada a corpora especializados não representados no conjunto de destilação.

A metodologia proposta se desenvolve em cinco estágios integrados. O primeiro estágio consiste na formalização da tarefa de extração de citações como um problema de aprendizado supervisionado via destilação professor-aluno. Dado um contexto textual C e uma pergunta Q, o objetivo é treinar um modelo capaz de identificar um subconjunto mínimo de sentenças $R \subset C$ que estejam semanticamente alinhadas com a resposta esperada A. Um aspecto crítico dessa formalização é a assimetria entre as entradas do professor e do aluno. Durante a construção do conjunto de dados, o modelo professor tem acesso ao triplo (Q, A, C), ou seja, conhece tanto a pergunta quanto a resposta esperada ao selecionar as citações de apoio. O modelo aluno, por sua vez, é treinado para produzir as mesmas seleções de citações recebendo apenas (Q, C), sem acesso à resposta A. Essa assimetria é deliberada: em tempo de inferência, a resposta é desconhecida, de modo que o aluno deve aprender a identificar evidências que sustentam a resposta a partir apenas da pergunta e do contexto. O processo de destilação, portanto, transfere a capacidade do professor de reconhecer passagens relevantes para um modelo que opera sob condições realistas de implantação, nas quais a resposta ainda não foi gerada.

O segundo estágio abrange a geração automatizada do conjunto de dados por meio de uma configuração híbrida de modelos professores composta por Gemini 2.5 Pro e GPT-5.1, orquestrados via LangChain. Para cada amostra, os modelos professores foram instruídos a extrair apenas o conjunto mínimo de citações do contexto que sustentasse diretamente a resposta de referência, utilizando um formato fixo de delimitação (`##begin_quote##` ,... `##end_quote##`). Essa convenção facilita a extração automática de citações e estabelece um padrão de formatação que o modelo aluno deve internalizar para uso em pipelines de produção. A escolha de delimitadores textuais em vez de JSON estruturado foi motivada por evidências recentes de que restrições de formato rígido degradam o raciocínio e introduzem sobrecarga operacional com taxas de falha não triviais. Para garantir a qualidade dos dados, o conjunto de teste completo e aproximadamente 2% do conjunto de treinamento foram manualmente revisados pelo autor.

O terceiro estágio consiste no ajuste fino dos modelos alunos compactos. Três modelos com capacidades distintas foram selecionados: Qwen 3:4B, Llama 3.2:3B e Llama 3.2:1B. Esses modelos representam pontos distintos ao longo do espectro desempenho-eficiência, desde uma configuração ultra-leve (1B) até um limite superior (4B) para avaliar a relação entre escala e qualidade da extração. Todos foram ajustados por meio de Adaptação de Baixo Posto (LoRA) com pesos base quantizados em 4 bits (INT4), enquanto apenas os adaptadores LoRA foram treinados em precisão FP16. A adaptação LoRA foi aplicada tanto às camadas de atenção quanto às camadas MLP da arquitetura transformer, seguindo evidências empíricas de que atualizar ambos os tipos de camadas produz desempenho superior.

O quarto estágio define um framework de avaliação híbrido que combina julgamento semântico via avaliação baseada em LLM com métricas lexicais e estruturais determinísticas. Essa abordagem multifacetada garante que as citações extraídas sejam avaliadas não apenas pela qualidade do conteúdo, mas também pela conformidade de formato e pela reprodutibilidade.

O mecanismo central de avaliação emprega o GPT-4.0 como juiz semântico automatizado independente, não utilizado no processo de destilação, evitando viés circular. O juiz opera por comparação pareada em ambas as direções: a precisão semântica mede qual fração das citações extraídas é sustentada pelo conjunto de referência, enquanto a revocação semântica mede qual fração das referências é capturada pela saída do modelo. O F1-score semântico combina ambas em uma medida balanceada, capturando equivalências semânticas que seriam ignoradas por matching exato de strings.

A pontuação BM25 fornece uma medida determinística e agnóstica a modelos do alinhamento lexical, operando puramente sobre estatísticas de tokens. A pontuação de formatação constitui uma métrica binária que verifica se a saída adere estritamente ao formato de delimitação prescrito, requisito essencial para implantação em produção, onde as citações precisam ser programaticamente parseáveis. A combinação dessas três dimensões garante que o framework capture qualidade de conteúdo, conformidade estrutural e fidelidade lexical simultaneamente.

Os resultados experimentais fornecem evidências empíricas consistentes de que a separação entre extração de citações e raciocínio é uma estratégia viável e eficaz para melhorar pipelines RAG. A avaliação foi conduzida sobre 600 amostras de teste reservadas, comparando os três modelos alunos antes e depois do ajuste fino com LoRA.

No que diz respeito à conformidade de formatação, o Qwen 3:4B alcança 100% tanto antes quanto depois do ajuste fino, demonstrando capacidades de seguimento de instruções que se transferem diretamente para extração estruturada. Em contraste, os modelos Llama requerem adaptação específica: o Llama 3.2:1B melhora de 3,51% para 92,65%,

enquanto o Llama 3.2:3B aumenta de 63,94% para 93,49%. Em sistemas de produção, a formatação correta é essencial para a integração confiável com ferramentas downstream.

A pontuação BM25 complementa a avaliação semântica com uma medida determinística de sobreposição lexical. O Qwen 3:4B com LoRA alcança a maior pontuação (81,58%), seguido pelo Llama 3.2:3B com LoRA (76,66%). O Qwen 3:4B original já demonstra forte alinhamento lexical (67,23%), superando ambos os modelos Llama originais. Os ganhos com LoRA variam: +41,19 pp para Llama 3.2:1B, +36,47 pp para Llama 3.2:3B e +14,35 pp para Qwen 3:4B, indicando retornos decrescentes para modelos mais recentes que já capturam parte do comportamento necessário durante o pré-treinamento. Observa-se também forte correlação entre conformidade de formatação e pontuações BM25: a saída estruturada reduz tokens espúrios e foca a atenção do modelo na evidência central.

Do ponto de vista semântico, os resultados revelam perfis de desempenho distintos entre as famílias de modelos. Para os modelos Llama, o LoRA produz melhorias consistentes: o Llama 3.2:1B apresenta ganhos modestos (F1: 23,52% → 32,08%), confirmando que modelos muito pequenos permanecem limitados em capacidade para extração multi-hop, enquanto o Llama 3.2:3B alcança melhorias substanciais (F1: 38,31% → 59,14%). O achado mais revelador diz respeito ao Qwen 3:4B, cujo modelo original já alcança 68,32% de revocação, 83,75% de precisão e 71,30% de F1, superando ambos os modelos Llama originais e até o Llama 3.2:3B ajustado. Após o LoRA, o Qwen 3:4B atinge o melhor desempenho geral: 83,84% de revocação, 75,90% de precisão e 75,99% de F1. Observa-se um trade-off em que a revocação aumenta (+15,52 pp) enquanto a precisão diminui (−7,85 pp). Esse trade-off é coerente com o objetivo do extrator, que não é depurar perfeitamente as citações, mas reduzir substancialmente o ruído do contexto recuperado. Trata-se de uma filtragem gradual: mesmo com citações marginais remanescentes, o contexto entregue ao gerador é significativamente mais enxuto que o original. Privilegiar revocação é, portanto, deliberado, já que omitir uma citação crucial pode induzir alucinação, enquanto incluir citações adicionais topicamente relacionadas representa apenas um incremento sobre um contexto já reduzido.

Para validar a utilidade prática da extração de citações, uma comparação controlada foi conduzida utilizando três modelos geradores de base em duas condições: recebendo apenas as citações curadas e recebendo o contexto completo. Os resultados demonstram que fornecer citações concisas em vez do contexto integral melhora consistentemente a acurácia das respostas em todas as configurações testadas. O Llama 3.2:1B quase triplica sua acurácia, subindo de 24,4% para 62,2% (+37,8 pp), evidenciando como o ruído contextual prejudica desproporcionalmente arquiteturas compactas. O Llama 3.2:3B melhora de 57,7% para 83,0% (+25,3 pp), enquanto o GPT-3.5 Turbo, um modelo consideravelmente maior e mais capaz, ainda registra um ganho expressivo, subindo de 75,8% para 88,5%

(+12,7 pp). Esses resultados confirmam que a filtragem de citações de fato auxilia consideravelmente a acurácia das respostas em modelos de diversos tamanhos, comprovando a importância do trabalho em questão.

Palavras-chave: geração aumentada por recuperação; extração de trechos relevantes; destilação de conhecimento; modelos de linguagem compactos; ajuste fino; avaliação semântica; ruído contextual

Abstract

This thesis proposes a lightweight quote-extraction pipeline that strengthens Retrieval-Augmented Generation (RAG) while remaining practical for resource-constrained deployments. A hybrid teacher setup composed of Gemini 2.5 Pro and GPT-5.1 first annotates a subset of the HotpotQA corpus with sentence-level evidence through a black-box distillation process, in which the teacher models generate structured training data without exposing their internal representations. These annotations are then used to fine-tune three compact student models (Qwen 3:4B, Llama 3.2:3B, and Llama 3.2:1B) via Low-Rank Adaptation (LoRA) with 4-bit quantized base weights during training to minimize GPU memory consumption. After training, the LoRA adapters are merged into full-precision checkpoints suitable for standard deployment. Remarkably, all three students require only a single epoch (under three minutes each on a single A100 GPU) to learn to isolate the excerpts most germane to a user's query.

The distilled extractor offers three core benefits. First, noise reduction: by supplying the generator with a short, evidence-focused context window, it lowers hallucination risk and speeds inference. Second, plug-and-play modularity: it integrates with a range of generators, from 1B-parameter SLMs to GPT-class models, without further fine-tuning of the generator itself. Third, transparency: every answer is accompanied by explicitly surfaced supporting quotes, facilitating auditability.

A hybrid evaluation framework (combining semantic precision, recall, and F1-score assessed by GPT-4.0 as an independent automated judge with deterministic BM25 lexical scoring and binary formatting compliance) reveals substantial improvements after fine-tuning. The best-performing student, Qwen 3:4B, achieves 83.84% semantic recall, 75.99% F1-score, and perfect formatting compliance. When extracted quotes replace full context as input to baseline generators, answer accuracy improves across all tested configurations: Llama 3.2:1B rises from 24.4% to 62.2%, Llama 3.2:3B from 57.7% to 83.0%, and GPT-3.5 Turbo from 75.8% to 88.5%, with smaller models exhibiting the largest gains. These findings demonstrate that a few minutes of targeted fine-tuning on a compact model can deliver disproportionate improvements to RAG pipelines, offering an efficient, interpretable, and easily deployable alternative to heavyweight end-to-end

approaches.

Keywords: retrieval-augmented generation; quote extraction; knowledge distillation; small language models; low-rank adaptation; semantic evaluation; contextual noise

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Large Language Models (LLMs) have delivered state-of-the-art performance across a wide range of natural language processing tasks, including question answering, summarization, and dialogue generation [2]. However, these capabilities come at the cost of high computational demands, inference latency, and significant memory usage, which limit practical deployment in real-time or resource-constrained environments [3, 4]. Retrieval-Augmented Generation (RAG) offers a promising direction to mitigate some of these limitations by retrieving external knowledge at inference time rather than encoding all world knowledge within the model itself [5].

Nevertheless, standard RAG pipelines introduce a problem of their own. The retrieval stage typically returns full passages or documents based on shallow relevance signals such as keyword overlap or embedding similarity, often flooding the model's input with redundant, loosely relevant, or outright distracting content [6]. The generative model must then simultaneously filter this noise and reason over it to produce an answer, a dual burden that increases the risk of hallucinated responses and reduces factual grounding. This problem is particularly acute for Small Language Models (SLMs), which have limited capacity to process and reason over long, noisy contexts [7].

These challenges point to a clear need: rather than asking a single model to both filter and reason, RAG pipelines would benefit from a dedicated component that selects only the most relevant evidence *before* it reaches the generator. This thesis proposes such a component, a compact, inference-efficient quote extractor trained through knowledge distillation, and investigates whether interposing it between retrieval and generation can improve answer quality, reduce hallucinations, and make RAG pipelines more practical for deployment at scale.

1.1 Motivation

While the limitations of noisy context in RAG are well recognized, existing approaches to addressing them present significant trade-offs. Frameworks such as RAFT [1] attempt to improve RAG performance by training a single model to jointly reason over retrieved documents and generate answers. However, this joint approach requires fine-tuning large models on domain-specific data, making it computationally expensive and difficult to adapt across tasks or domains. Prompt engineering techniques such as Chain of Thought or instruction-based filtering can improve reasoning without modifying model parameters, but they do not fundamentally solve the noise problem: the model still receives the full, unfiltered context and must expend capacity distinguishing relevant from irrelevant information.

A further gap concerns model scale. Most existing RAG optimization work focuses on large models with tens or hundreds of billions of parameters. Yet there is growing practical interest in deploying SLMs, models in the range of roughly one to five billion parameters, in latency-sensitive, edge, or cost-constrained settings [7]. These compact models stand to benefit the most from cleaner input, precisely because they have the least capacity to spare on filtering noisy contexts. Despite this, few studies have systematically investigated how targeted evidence extraction affects downstream performance across models of varying scale.

This thesis is motivated by the hypothesis that a modular, decoupled architecture, in which a lightweight extractor selects relevant evidence and a separate generator reasons over it, can outperform monolithic approaches, particularly for compact models. By delegating the filtering task to a specialized component trained through black-box knowledge distillation, the generator is freed to focus exclusively on reasoning over concise, high-quality evidence. This separation of concerns also yields practical benefits, since the extractor can be trained once and reused across different generators without requiring any modification to the generative component, making the system adaptable and scalable across deployment scenarios. We acknowledge, however, that the extractor's effectiveness may degrade when applied to domains substantially different from those seen during training, an open question discussed in the Future Work section.

1.2 Objective

The objective of this thesis is to develop and evaluate a modular quote-extraction system that enhances Retrieval-Augmented Generation pipelines by training a compact language

model to identify semantically relevant excerpts from retrieved contexts before they are passed to a downstream generator.

1.2.1 Specific Objectives

The main goal is pursued through the following specific objectives:

- Formalize the quote extraction task as a supervised learning problem using a teacher–student distillation framework, in which a high-capacity teacher model generates structured quote-level supervision for training a compact student model.
- Construct an automated dataset generation pipeline in which the teacher model produces curated, delimited excerpts from large textual contexts, forming a gold-standard training corpus.
- Fine-tune multiple compact language models of varying parameter scale using parameter-efficient adaptation techniques, and evaluate how model capacity influences extraction quality.
- Design a hybrid evaluation framework that combines semantic judgment via LLM-based judges, deterministic lexical scoring, and structural compliance metrics to assess extraction performance across multiple dimensions.
- Conduct a controlled comparison between quote-only and full-context input conditions to quantify the downstream impact of quote extraction on answer accuracy across generators of different sizes.
- Demonstrate that the trained quote extractor integrates as a reusable preprocessing component into RAG pipelines without requiring fine-tuning or modification of the downstream generator, within the scope of the domains represented in the training data.

1.2.2 Contributions

This work contributes four main elements. First, a black-box knowledge distillation pipeline for training compact quote extraction models that operate independently of the teacher's architecture, making the approach compatible with proprietary or API-only teacher models.

Second, a systematic empirical analysis of quote extraction across multiple student models spanning the SLM parameter spectrum,providing evidence on the relationship between model scale, extraction quality, and formatting compliance.

Third, a multi-metric evaluation framework that combines semantic precision, recall, and F1-score assessed by an independent LLM judge with deterministic BM25 lexical scoring and binary formatting compliance, offering a more robust assessment than any single metric alone.

Finally, a modular pipeline architecture in which evidence extraction is decoupled from answer generation, enabling the extractor to be trained once and reused across diverse generators and domains without modification.

1.3 Organization of the Thesis

This thesis is organized into seven chapters and one appendix, structured as follows:

- **Chapter 1 – Introduction.** This opening chapter frames the research problem, motivates the need for more efficient context selection in Retrieval-Augmented Generation, and states the overarching and specific objectives that guide the work.
- **Chapter 2 – Theoretical Background.** This chapter lays the conceptual foundation, covering large and small language models, the transformer architecture, fine-tuning and parameter-efficient adaptation, knowledge distillation, retrieval-augmented generation, quantization, agent-based reasoning, and semantic and lexical evaluation metrics.
- **Chapter 3 – Related Work.** A survey of prior studies on RAG optimization, parameter-efficient fine-tuning, knowledge distillation, and semantic evaluation that inform and contextualize the proposed methodology.
- **Chapter 4 – Methodology.** The design of the proposed quote-extraction pipeline is presented, including the teacher-student distillation framework, the dataset generation process, the fine-tuning strategy, and the hybrid evaluation protocol.
- **Chapter 5 – Experimental Setup.** All practical aspects of the study are described, including the dataset, hardware resources, hyperparameters, model configurations, and baseline conditions used to benchmark the proposed approach.
- **Chapter 6 – Results and Discussion.** Quantitative metrics and qualitative examples are analyzed to assess quote extraction performance and the downstream impact of quote filtering on answer accuracy. Findings are interpreted in light of the research objectives.
- **Chapter 7 – Conclusion.** The main findings are summarized, broader implications are discussed, and directions for future work are outlined.

- **Appendix A – LLMQuoter Web Application.** A companion web application developed for interactive testing and demonstration of the fine-tuned models is described, including its architecture and relevance to the thesis.

Chapter 2

Background

2.1 Large Language Models (LLMs)

Large Language Models (LLMs) are advanced deep learning models trained on vast amounts of textual data to understand and generate human-like language. They are designed to perform various natural language processing tasks, such as translation, summarization, and question-answering, by predicting the next word in a sequence based on provided context [2].

LLMs leverage the **Transformer architecture**, enabling them to process and generate text efficiently. Introduced by Vaswani et al. [8], the Transformer marked a major turning point in natural language processing by moving away from recurrent and convolutional neural networks. Instead, it relies entirely on attention mechanisms to model relationships between words in a sequence, enabling greater parallelization, more stable training, and improved performance on long-range dependencies.

At its core, the Transformer combines self-attention mechanisms, positional encodings, and feedforward neural networks. This design has become the foundation of modern state-of-the-art language models, including BERT, GPT, and T5 [9].

Generative Artificial Intelligence (Generative AI) emerges as a direct consequence of the architectural advances introduced by the Transformer and subsequently scaled in large language models. While the Transformer provides the computational framework for modeling long-range dependencies through attention mechanisms, large-scale pre-training enables these models to internalize rich statistical representations of language. When trained on vast corpora using autoregressive or masked objectives, such models acquire the ability not only to understand text but also to generate novel content conditioned on contextual input.

In this sense, **Generative AI** refers to systems capable of producing new artifacts—such as text, images, audio, and source code—by sampling from learned probability distribu-

tions over high-dimensional data spaces. Large language models such as GPT-4 exemplify this paradigm, as they leverage Transformer-based architectures to generate coherent, contextually appropriate, and often highly persuasive outputs across a wide range of domains [10]. Rather than retrieving memorized passages, these systems synthesize responses token by token, guided by learned patterns of syntax, semantics, and discourse structure. This generative capability, grounded in Transformer-based sequence modeling, underpins many of the practical applications and societal impacts discussed in the following sections.

In practice, generative AI has been rapidly adopted across multiple sectors. In education, these systems support learners through personalized explanations, writing assistance, and interactive tutoring, while also raising important questions about assessment integrity and responsible use [11, 12]. In healthcare, generative models assist with tasks such as drafting clinical reports, summarizing patient histories, and supporting decision-making, with recent work suggesting their potential to enhance medical simulation and research workflows [13]. Creative industries are also being reshaped, as AI-generated music, visual art, and literary content increasingly complement or transform traditional creative processes [14]. In business, organizations leverage generative AI for drafting emails, automating reports, and supporting marketing activities, leading to measurable productivity gains but also redefining white-collar job roles [15].

Despite these benefits, generative AI introduces substantial **societal risks**. Widespread automation may disrupt labor markets, requiring new policies and workforce adaptation strategies [15]. The ability of AI systems to generate highly realistic but false content amplifies the threat of misinformation, as documented in recent case studies of real-world harm [16]. Bias in training data can propagate through model outputs, reinforcing gender, racial, and cultural stereotypes unless actively mitigated [17]. More broadly, ethical concerns surrounding authorship, consent, data ownership, and accountability highlight the need for robust governance frameworks and transparent usage policies [18].

2.2 Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) is a paradigm that enhances large language models (LLMs) by coupling them with external information sources at inference time. Instead of relying solely on knowledge acquired during pre-training, RAG allows models to dynamically retrieve relevant evidence from external repositories before generating a response. In this thesis, the proposed methodology is applied **only to the generation stage of RAG, after the retrieval step has already occurred**—that is, assuming the context

has been retrieved, our contribution focuses on how this context is processed and filtered before being used by the generator [19].

The core motivation for RAG stems from two fundamental limitations of LLMs: their knowledge is static (frozen at training time), and they are prone to hallucinations when asked questions that require precise factual grounding. By retrieving external documents at runtime, RAG mitigates both issues, enabling models to provide answers that are not only fluent but also better aligned with real, verifiable information. Recent work has shown that grounding responses in retrieved evidence substantially improves factual accuracy and contextual relevance, including in multimodal settings that combine video and text [20].

Another key advantage of RAG is its adaptability. Instead of retraining a model whenever new information emerges, retrieval mechanisms can be updated independently, allowing the system to remain current while keeping the underlying model unchanged. Adaptive retrieval strategies such as Flare-Aug demonstrate how retrieval intensity can be dynamically adjusted to balance recency, relevance, and computational efficiency [21]. This modularity makes RAG particularly attractive for specialized domains, where curated corpora can be incrementally expanded without modifying the model itself [22, 23].

RAG systems can draw information from a wide variety of sources. Public web pages, scientific articles, and news repositories allow access to real-time information [24]. Vector databases enable semantic search over large document collections, retrieving passages based on meaning rather than keyword matching, which is now a foundational technique in modern RAG pipelines [25]. Structured data sources such as relational databases and knowledge graphs further improve grounding by providing precise, machine-readable facts that can be integrated with natural language generation [26, 27]. In enterprise settings, internal repositories—manuals, technical reports, and proprietary documentation—are commonly used to tailor responses to organizational needs [28].

At a high level, the standard RAG workflow follows a consistent pattern. A user submits a query, which is used to retrieve relevant documents from external sources. These documents are then combined with the original question to form an augmented input, which is finally passed to the LLM for generation. This process enables models to reason over external evidence rather than relying solely on internal knowledge.

Recent work has identified **noisy or partially relevant context as a central bottleneck** for Retrieval-Augmented Generation (RAG), since even strong models may **hallucinate or be distracted** when the retrieved passages are off-topic or contradictory [29]. To address this, several methods insert explicit denoising stages between retrieval and generation, including fine-grained filtering modules that locate and retain only answer-bearing sentences from long documents, as well as information-bottleneck style objectives

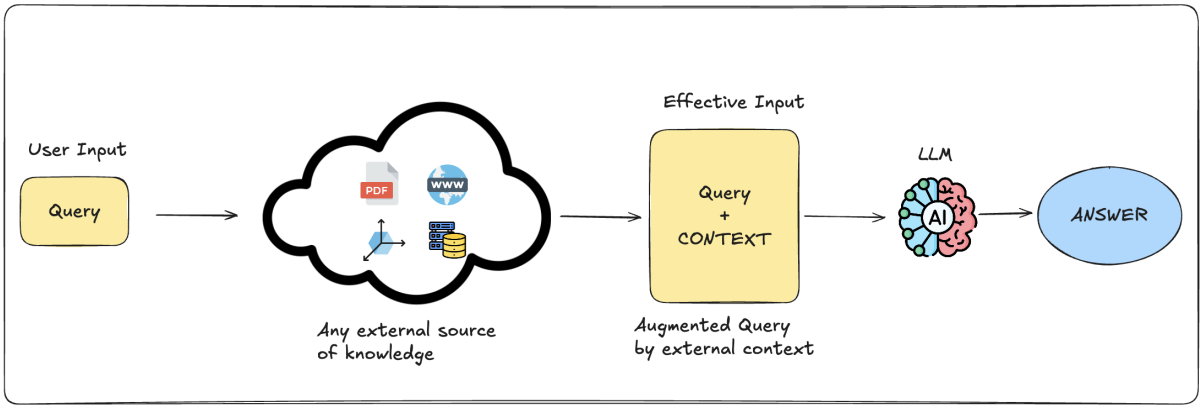

Figure 2.1: General RAG algorithm

that learn compressed intermediate representations which preserve answer-relevant information while discarding noise.[30, 31]

A complementary line of work focuses on reducing noise at the retrieval or control stage rather than only applying post-hoc filters. Examples include adaptive frameworks that decide when and how much to retrieve based on the information completeness of the query (thereby avoiding unnecessary, noisy retrieval) and domain-specific retrievers that are trained to filter out task-irrelevant signals such as spurious financial patterns before passing context to the generator.[32, 33, 29] Together, these approaches treat RAG as a multi-stage pipeline—retrieval, filtering, compression, and generation—explicitly optimized to minimize contextual noise while preserving enough evidence for accurate grounded answers.[31]

Overall, RAG represents a practical bridge between generative models and real-world information systems. By separating retrieval from generation, it allows each component to be optimized independently. In this work, we take this separation one step further by introducing a dedicated quote-extraction stage that operates **after retrieval but before generation**, aiming to reduce noise, improve grounding, and lower the cognitive burden on downstream models.

2.3 LLM Reasoning

Reasoning within Large Language Models (LLMs) refers to their ability to understand, integrate, and manipulate knowledge to solve tasks that require logic, inference, and decision-making. While LLMs have demonstrated significant progress in generating coherent and contextually relevant text, their capacity for deep logical reasoning and reliable inference remains an ongoing challenge in AI research [34, 35].

The development of reasoning capabilities in LLMs has followed two primary approaches: **prompt engineering**, which leverages carefully designed input formulations to elicit reasoning behavior without modifying model parameters, and **fine-tuning**, which adapts pre-trained models through continued training on reasoning-specific datasets. Both approaches aim to enhance the model's ability to perform multi-step logical inference, handle complex problem decomposition, and maintain coherence across extended reasoning chains. This section examines both strategies before discussing the broader challenges of multi-step logic in contemporary language models.

Prompt Engineering for Reasoning. Prompt engineering refers to the systematic design, formulation, and optimization of input prompts to elicit desired behaviors from large language models [36]. Rather than modifying the underlying model architecture or parameters, prompt engineering leverages the model's existing capabilities through strategic input design. This approach has proven particularly effective for reasoning tasks, where the structure and framing of prompts can substantially influence both the accuracy and reliability of model outputs.

Recent research demonstrates that prompt characteristics—such as the inclusion of exemplars (few-shot prompting), explicit instructions, or stepwise reasoning cues—significantly impact model performance on complex, multi-step tasks [36, 37]. Structured prompting strategies can guide models to articulate intermediate reasoning steps, decompose problems into manageable components, and maintain logical consistency throughout the inference process. These techniques continue to evolve as researchers address challenges such as prompt sensitivity, context length constraints, and domain-specific adaptation [36, 38].

The most fundamental and widely adopted technique in this domain is Chain of Thought (CoT) prompting. Instead of requesting an immediate final answer, CoT explicitly instructs the model to articulate intermediate reasoning steps before reaching a conclusion [37]. This approach encourages the model to externalize its logical pathway, making the reasoning process both more reliable and more interpretable. The mechanism is straightforward: by prompting the model to "think step by step," CoT activates latent reasoning capabilities that remain dormant under standard zero-shot prompting conditions, where models are asked to provide direct answers without scaffolding or intermediate steps.

A canonical example illustrates this principle clearly. When asked "If John has 3 apples and gives away 1, how many are left?", a CoT-prompted model responds: "Let's think step by step: John starts with 3 apples. He gives away 1 apple. $3 - 1 = 2$ apples. Therefore, he has 2 apples left." By decomposing the problem into explicit sub-steps—identifying the initial state, applying the transformation, and computing the result—the model demon-

strates a more structured and verifiable reasoning process. Empirical studies have shown that CoT prompting yields substantial improvements on mathematical reasoning, logical inference, and multi-hop question answering tasks [37].

However, while Chain of Thought improves reasoning transparency and reliability, individual reasoning chains can still exhibit considerable variability, particularly when models use sampling-based generation with elevated temperature settings. The same prompt may produce different reasoning paths and, consequently, different final answers across multiple generations. To address this limitation, researchers developed Self-Consistency, a technique that generates multiple independent reasoning paths and aggregates their conclusions through majority voting [39].

The Self-Consistency procedure is conceptually elegant: given a reasoning task, the model is prompted k times to produce k distinct reasoning chains. Each chain may follow a different logical pathway or intermediate representation, but all converge toward a final answer. The most frequently occurring answer across all samples is selected as the final output, effectively leveraging the model's internal uncertainty to identify robust solutions. Formally, if $\{a_1, a_2, \ldots, a_k\}$ represent the final answers from k reasoning chains, the aggregated answer is $a^* = \arg\max_a \sum_{i=1}^{k} \mathbb{1}(a_i = a)$, where $\mathbb{1}(\cdot)$ is the indicator function. This approach has been shown to improve answer reliability and robustness across diverse reasoning benchmarks, particularly for ambiguous or multi-faceted problems where multiple valid reasoning approaches exist [39]. The primary limitation of Self-Consistency is that majority voting operates exclusively on final answers without evaluating the validity or logical soundness of the intermediate reasoning steps themselves. Consequently, the technique may still select incorrect answers if multiple reasoning chains independently converge on the same erroneous conclusion.

Building upon these foundations, Tree of Thoughts (ToT) introduces a more sophisticated approach that models reasoning as structured exploration over multiple trajectories at each decision point [40]. Rather than committing to a single linear reasoning path (as in standard CoT) or sampling multiple independent chains in parallel (as in Self-Consistency), ToT represents the reasoning process as a search problem over a tree of intermediate states. Each node in this tree represents a partial reasoning step, and edges represent possible continuations or alternative reasoning directions.

At each stage of reasoning, the model generates multiple candidate thoughts—intermediate conclusions or reasoning steps—forming branches in the tree. An evaluation mechanism, which may be learned, heuristic, or based on external feedback, assesses the promise of each branch. The search process then selectively explores high-value paths while pruning unpromising directions, effectively performing a form of lookahead reasoning. This structured exploration offers several advantages over simpler approaches: by evaluating

reasoning trajectories before committing to a final answer, ToT can recover from early missteps and avoid locally coherent but globally incorrect reasoning chains. It is particularly effective for tasks requiring complex planning, constraint satisfaction, or multi-hop inference, where early decisions significantly influence downstream outcomes [40]. However, the benefits come at a computational cost. Generating and evaluating multiple branches at each reasoning step requires substantially more inference than linear CoT or even Self-Consistency, making ToT best suited for high-stakes reasoning tasks where accuracy justifies the additional overhead.

Figure 2.2 illustrates the conceptual progression across these three techniques. Standard prompting requests direct answers without intermediate steps. Chain of Thought introduces linear step-by-step reasoning. Self-Consistency extends this by generating multiple independent reasoning chains and aggregating their outputs. Tree of Thoughts further generalizes the paradigm by exploring branching trajectories and evaluating partial reasoning states, enabling more systematic and globally optimized inference.

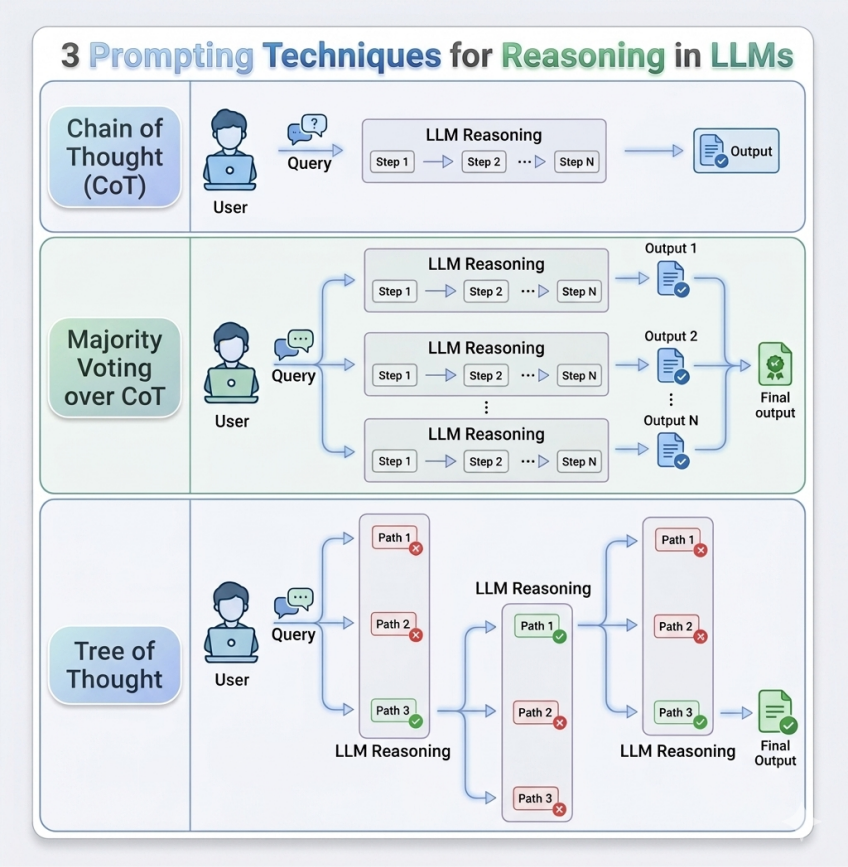

Figure 2.2: Comparison of prompt engineering techniques for enhancing reasoning capabilities in large language models. From left to right: standard input-output prompting, Chain of Thought prompting with intermediate steps, Self-Consistency aggregating multiple reasoning paths, and Tree of Thoughts exploring branching reasoning trajectories.

Fine-Tuning for Reasoning. While prompt engineering techniques operate within the constraints of a fixed pre-trained model, fine-tuning offers an alternative approach that modifies the model's internal parameters to enhance reasoning capabilities. Fine-tuning involves continued training of a pre-trained LLM on curated datasets that emphasize logical inference, multi-step problem solving, and structured reasoning patterns [41, 42].

This approach is particularly effective for domain-specific reasoning tasks where generic pre-training may provide insufficient coverage. For instance, models fine-tuned on mathematical reasoning datasets demonstrate improved performance on arithmetic word problems, algebraic manipulation, and formal proof construction. Similarly, fine-tuning on scientific literature enhances reasoning over domain-specific concepts, nomenclature, and inferential patterns [41]. Task-specific fine-tuning allows models to internalize reasoning strategies that would otherwise require explicit prompting, effectively compressing multi-step reasoning patterns into the model's learned parameters. This internalization can reduce the need for elaborate prompts at inference time and improve consistency across diverse problem instances within the target domain [42].

However, fine-tuning for reasoning introduces several important trade-offs. First, it requires access to high-quality labeled reasoning data, which may be expensive or difficult to obtain for specialized domains. Second, the computational resources required for retraining large models can be substantial, particularly when adapting models with billions of parameters. Third, careful regularization is necessary to prevent overfitting to the fine-tuning dataset or catastrophic forgetting of general capabilities acquired during pre-training. Finally, fine-tuned models may exhibit reduced flexibility compared to prompt-based approaches, as their reasoning behavior becomes more tightly coupled to the specific characteristics and biases present in the fine-tuning data.

Despite these challenges, fine-tuning remains a powerful tool for enhancing reasoning performance when prompt engineering alone proves insufficient, particularly in applications where consistent, domain-specific reasoning is required at scale.

2.3.1 Multi-Step Logic and Current Limitations

Multi-step logic refers to the capacity of LLMs to decompose complex tasks into sequential reasoning stages, where each stage builds upon the conclusions of previous steps to arrive at a final solution. This capability is fundamental to advanced reasoning tasks such as mathematical proofs, multi-hop question answering, causal inference, and strategic decision-making [43].

The prompt engineering techniques discussed previously—particularly Chain of Thought and Tree of Thoughts—explicitly operationalize multi-step logic by instructing models to articulate and traverse intermediate reasoning states. Similarly, fine-tuning approaches

can embed multi-step reasoning patterns directly into model parameters, enabling more fluid execution of sequential logical operations. In both cases, the goal is to enable models to maintain coherent reasoning across multiple inferential steps, integrating information from previous stages while planning ahead toward problem resolution.

Despite these advances, multi-step reasoning in current LLMs reveals several persistent limitations. One significant challenge is **task interference**, where models experience performance degradation when required to handle multiple reasoning objectives simultaneously [34]. When an LLM is prompted to perform several distinct operations—such as summarizing a document, answering detailed factual questions, and generating an explanatory analysis—within a single context, accuracy and logical coherence often decline. This phenomenon suggests that even large models operate under cognitive resource constraints analogous to working memory limitations in human reasoning. The model's attention and processing capacity become divided across competing objectives, leading to confusion, incomplete reasoning chains, and diminished overall effectiveness.

Additionally, multi-step reasoning chains are susceptible to error propagation, where mistakes made in early reasoning stages cascade through subsequent steps, ultimately producing incorrect final conclusions. This fragility is particularly pronounced in long reasoning chains or when intermediate steps lack explicit verification mechanisms. While techniques such as Self-Consistency partially mitigate this issue by aggregating multiple reasoning attempts, they do not fundamentally address the challenge of validating intermediate logical steps. A single erroneous premise or flawed inference at an early stage can systematically corrupt all downstream reasoning, even when the model's later steps follow valid logical patterns.

Furthermore, LLMs often struggle with maintaining consistency across extended reasoning contexts, particularly when problems require tracking multiple interdependent state variables or reconciling potentially contradictory information [44]. The model may lose track of earlier constraints, introduce inconsistencies between reasoning steps, or fail to propagate critical information forward through the reasoning chain. These limitations become especially apparent in tasks requiring complex state management, such as planning under constraints, logical puzzle solving, or reasoning about dynamic systems[45].

One practical approach to mitigating these limitations involves decomposing complex workflows into multiple independent model calls rather than attempting to perform all reasoning steps within a single extended context [46, 47]. By splitting tasks across separate invocations, each focused on a specific sub-objective, systems can reduce the cognitive load on individual inference passes and minimize task interference effects [46]. For instance, rather than prompting a model to simultaneously retrieve information, perform analysis, and generate a formatted report, a multi-call architecture might first extract

relevant facts in one call, then analyze those facts in a second call, and finally synthesize the results in a third call [48, 49]. This modular approach offers several advantages: it isolates errors to specific stages rather than allowing them to propagate through an entire reasoning chain, enables independent validation of intermediate outputs, and allows different prompting strategies or even different models to be applied at each stage according to task requirements [48, 49]. However, this strategy introduces its own challenges, including increased latency from multiple round-trips, the need for explicit state management between calls, and potential loss of context or coherence across invocations. The trade-off between monolithic long-context reasoning and decomposed multi-call architectures remains an active area of research and engineering practice [46, 47].

These challenges underscore the need for continued research into more robust reasoning architectures, improved error detection mechanisms, and hybrid approaches that combine neural reasoning with symbolic verification. While contemporary LLMs demonstrate impressive reasoning capabilities through prompt engineering and fine-tuning, their performance on multi-step logic tasks remains bounded by architectural constraints, training paradigms, and the inherent difficulty of generalizable logical inference. Understanding these limitations is essential for designing systems that appropriately leverage LLM reasoning strengths while mitigating known failure modes.

2.4 Small Language Models (SLMs)

In this work, we adopt a parameter-centric view of *Small Language Models* (SLMs). Lu et al. [7] scope their survey to transformer-based language models with roughly 10^8 to 5×10^9 parameters (about 100M–5B), explicitly treating this interval as the "small" regime in contrast to models with tens or hundreds of billions of parameters.[7] Belcak et al. similarly focus on models in the 1B–8B range when discussing SLMs that can "still pack a punch" in agentic workloads, emphasizing that these models are substantially smaller than frontier LLMs while retaining strong capabilities [50]. Concrete model families are designed to fall into this band: MiniCPM introduces 1.2B and 2.4B parameter variants as SLMs, and SmolLM2 presents a 1.7B parameter model explicitly branded as a small language model.[35, 51] Motivated by these works, we use the term *small language model* to refer to models with on the order of 10^8–10^9 parameters, i.e., roughly 100M to 5–8B parameters.[7, 50]

Small Language Models (SLMs) are neural language models typically ranging from tens of millions to a few billion parameters, designed to deliver effective language capabilities under tight computational, memory, and latency constraints [7, 52]. In contrast to very

large language models (LLMs), SLMs intentionally trade some raw expressive power for efficiency, controllability, and practical deployability in real-world systems [52, 53].

A central motivation for SLMs is local or on-premise deployment, where models can run on organization-controlled hardware rather than relying on remote cloud APIs [7, 54]. This reduces exposure to service outages, rate limits, and data leakage risks, while facilitating tighter integration with internal databases, APIs, and enterprise workflows [53, 55]. Because SLMs require substantially less memory and compute, they can often be hosted on a single GPU, consumer-grade workstations, or even edge devices and microcontrollers [7, 56].

From a governance and privacy perspective, local hosting keeps prompts, logs, and model outputs within organizational boundaries, mitigating the risk of sharing sensitive information with third-party providers [55, 53]. Open-source SLMs further enable direct inspection of model weights, logging behavior, and retention policies, supporting auditing, compliance, and alignment with internal governance standards [52, 53]. Moreover, fine-tuning can be performed entirely on private datasets—such as internal documents, tickets, or source code—without transferring proprietary data outside institutional infrastructure [57, 58].

In terms of efficiency, SLMs generally exhibit lower inference latency, reduced energy consumption, and higher throughput per unit of hardware compared to larger models [7, 56]. Empirical studies indicate that well-optimized SLMs can achieve competitive performance per watt, making them attractive for resource-constrained devices and cost-sensitive large-scale deployments [56, 54]. Economically, their reduced compute requirements translate into lower per-token and per-request costs, enabling sustained high-volume usage in applications such as customer support, analytics, and coding assistance [59, 53].

SLMs are particularly well suited for specialization through parameter-efficient fine-tuning techniques such as LoRA or QLoRA, adapters, and targeted data augmentation [57, 7]. For structured or tool-centric tasks—such as text-to-SQL, API calling, or domain-specific question answering—carefully tuned SLMs can approach or even surpass larger general-purpose models on targeted benchmarks while remaining far cheaper to deploy [60, 59, 55]. This also supports incremental learning, where models can be periodically refined using newly collected interaction data rather than retrained from scratch [61, 58].

Finally, SLMs offer significant research value due to their smaller scale and often open training ecosystems [52, 7]. They serve as practical testbeds for studying robustness, interpretability, safety, and evaluation, while enabling controlled experimentation with pruning, quantization, and vocabulary reduction [62, 54]. As such, SLMs bridge applied

engineering needs with fundamental research on efficient, trustworthy, and scalable language technologies [52, 55].

2.5 Fine Tuning

Large Language Models (LLMs) are deep neural networks composed of billions of parameters trained to process and generate natural language. Built upon the transformer architecture [8], these models leverage self-attention mechanisms to capture dependencies across textual sequences, learning by iteratively adjusting their weights to minimize a loss function over massive corpora of text data.

The development of an LLM typically begins with a stage known as **pre-training**, in which the model is exposed to vast amounts of unlabelled data, such as books, web pages, and articles, and trained using self-supervised objectives like masked language modeling (MLM) or autoregressive next-token prediction [63]. Through this process, the model acquires a broad understanding of language structure, syntax, semantics, and general world knowledge. However, because pre-training is fundamentally task-agnostic, the resulting model does not inherently specialize in any particular downstream application such as summarization, translation, or question answering. This limitation motivates the need for an additional stage of training: **fine-tuning**.

Fine-tuning is the process of continuing the training of a pre-trained model on a smaller, task-specific dataset using supervised learning with labeled data tailored to the target application. By further adjusting the model's weights, fine-tuning enables it to specialize in solving problems that require task-specific reasoning, vocabulary, or output formats, capabilities that are often insufficiently captured during pre-training. This process enhances performance, generalization, and reliability on downstream tasks [64, 65].

The decision to fine-tune a model is typically driven by several interrelated factors. One of the most common motivations is **domain adaptation**: when a model needs to operate within a specialized field such as biomedicine, law, or finance, fine-tuning on domain-specific corpora allows it to adjust its vocabulary, reasoning patterns, and factual grounding to that domain. BioBERT, for instance, was fine-tuned on PubMed abstracts to significantly improve performance on biomedical tasks [66]. Closely related is the need for **task specialization**, where models must perform well on structured tasks such as classification, question answering, named entity recognition, or summarization. Models like T5 and BART were pre-trained on general text and subsequently fine-tuned on benchmarks such as SQuAD and CNN/DailyMail, achieving state-of-the-art results on these specific tasks [64, 1].

Beyond domain and task considerations, fine-tuning also plays a crucial role in addressing **low-resource scenarios**. When the target use case involves languages or data formats that are underrepresented in the original training corpus, fine-tuning can compensate for pre-training biases and enable effective transfer to settings where zero-shot or few-shot generalization would otherwise be weak [67]. Furthermore, organizations frequently fine-tune LLMs to achieve **custom behavioral alignment**, such as matching brand tone, reducing hallucinations, enforcing safety constraints, or complying with local regulations. Techniques like instruction tuning and reinforcement learning with human feedback (RLHF) are prominent forms of supervised fine-tuning designed to improve response quality and reliability in these contexts [68]. Finally, fine-tuning is especially relevant for **efficient deployment of smaller models** on edge devices, where parameter-efficient methods such as Low-Rank Adaptation (LoRA) and adapter-based fine-tuning allow compact models to retain task accuracy while minimizing inference cost [69, 70].

In summary, fine-tuning is a cornerstone of modern NLP pipelines, enabling the practical deployment of large models in real-world scenarios where task accuracy, domain relevance, and efficiency are critical. However, the standard approach to fine-tuning, which involves updating all model parameters, introduces significant practical challenges, particularly as models grow to billions of parameters. This motivates the development of more efficient adaptation strategies, among which Low-Rank Adaptation (LoRA) has emerged as one of the most widely adopted.

2.5.1 Low-Rank Adaptation (LoRA)

Low-Rank Adaptation (LoRA) is a parameter-efficient fine-tuning method designed to adapt large pre-trained neural networks to downstream tasks with minimal additional parameters [69]. To understand the motivation behind LoRA, it is important to first consider the limitations of the conventional approach.

Full fine-tuning involves updating all the parameters of a pre-trained neural network on a target dataset for a specific downstream task. In this approach, the entire model, often containing hundreds of millions or even billions of parameters, is retrained, allowing the network to adapt its internal representations comprehensively to the new data [71, 72, 73]. While this method can yield strong performance, it is computationally expensive and memory-intensive. For large-scale language models, maintaining and deploying separate copies of fully fine-tuned models for each downstream task quickly becomes impractical, particularly in resource-constrained environments or when frequent adaptation to new tasks is required [71, 72].

LoRA, introduced by Hu et al. (2021), addresses these limitations through an elegant insight: instead of updating the full set of parameters during fine-tuning, it freezes the

original pre-trained weights and injects additional trainable matrices of significantly lower rank into certain weight update paths. Specifically, for a given weight matrix $W_0 \in \mathbb{R}^{d \times k}$, LoRA modifies the update process as follows:

- **Standard update:**

$$W = W_0 + \Delta W$$

  In conventional fine-tuning, ΔW is a full matrix with the same dimensions as W_0.

- **LoRA update (low-rank decomposition):**

$$\Delta W = BA$$

  where:

$$B \in \mathbb{R}^{d \times r}$$
$$A \in \mathbb{R}^{r \times k}$$
$$r \ll \min(d, k)$$

By constraining the update ΔW to be the product of two much smaller matrices (B and A), LoRA dramatically reduces the number of trainable parameters needed for adaptation, while still allowing the model to learn effective task-specific adjustments. Figure 2.3 illustrates this mechanism: the input X is processed simultaneously by the frozen pre-trained weights W and the lightweight trainable update ΔW, and their results are combined to form the final output. This dual-path structure enables the model to learn new tasks efficiently without altering the original weights.

By optimizing only these low-rank matrices while keeping the pre-trained model weights fixed, LoRA achieves notable advantages in practice. First, it introduces only a small number of additional parameters per task, making multi-task and on-device adaptation feasible. Second, the memory and compute overhead are dramatically reduced compared to standard fine-tuning approaches. Third, and perhaps most remarkably, experiments have demonstrated that LoRA can match or even outperform traditional full fine-tuning across various downstream tasks, despite operating with a substantially smaller parameter and computational footprint [69].

These properties have led to LoRA's rapid adoption in both academic research and industry, establishing it as a foundational technique for scalable, parameter-efficient model adaptation. It is particularly valuable in settings where deploying full model copies for every downstream task is infeasible due to hardware constraints or resource limitations,

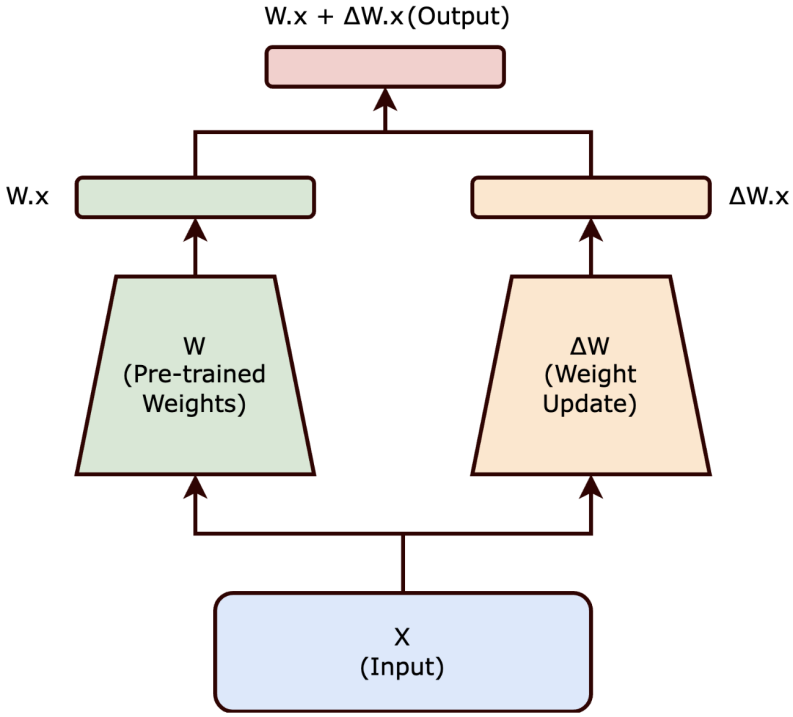

Figure 2.3: High level representation of LoRA usage

making the fine-tuning of large pre-trained models practical and accessible across a wide range of applications.

2.6 Knowledge Distillation in LLMs

As large language models continue to grow in scale, deploying them in resource-constrained or latency-sensitive environments becomes increasingly impractical. Knowledge distillation (KD) has emerged as a pivotal technique for addressing this challenge, enabling the transfer of knowledge from a high-capacity *teacher* model to a more compact and efficient *student* model. Through this process, smaller models can acquire the ability to perform complex tasks (such as reasoning, summarization, and domain-specific question answering) while significantly reducing computational and memory requirements [74].

The concept of knowledge distillation was originally formalized by Hinton et al. [75], who proposed that a student model could learn not only from hard labels (i.e., ground-truth annotations) but also from the *soft probability distributions* produced by a teacher model. The key insight is that these soft outputs encode richer information than one-hot labels: they capture inter-class similarities and the teacher's learned uncertainty, providing the student with a more nuanced training signal. In the original formulation, a temperature parameter T is applied to the softmax function to soften the teacher's output

distribution, and the student is trained using a combined loss that balances task-specific cross-entropy with a divergence term that encourages the student's predictions to match the teacher's soft targets.

While this foundational framework has proven effective, it assumes that the student has direct access to the teacher's internal representations during training—an assumption that does not always hold in practice, particularly when the teacher is a proprietary model accessible only through an API. This distinction has given rise to two broad families of knowledge distillation approaches.

The first, commonly referred to as **white-box distillation**, operates under the assumption that the student can observe the teacher's internals. This includes access to soft logits, intermediate layer activations, or attention distributions. Techniques such as temperature scaling and intermediate layer matching fall within this category and have been shown to mitigate hallucinations and improve factual accuracy in student models [76]. Methods like Dual-Space Knowledge Distillation (DSKD) further extend this paradigm by unifying the output spaces of teacher and student models to facilitate more effective knowledge transfer [77]. However, white-box approaches face practical limitations: they require architectural compatibility between teacher and student, they are computationally expensive during training since both models must be loaded simultaneously, and they are inapplicable when the teacher model's weights are not publicly available [78].

The second family, known as **black-box distillation**, relaxes these requirements entirely. In this paradigm, the teacher model is treated as a data generator rather than a live training signal. The teacher produces outputs (synthetic question-answer pairs, reasoning chains, or annotated examples, etc.)that are compiled into a new dataset, and the student is subsequently fine-tuned on this teacher-generated data using standard supervised learning. Because the student never accesses the teacher's internal states, this approach is agnostic to the teacher's architecture and is compatible with proprietary or API-only models [79].

Black-box distillation has been used as an important paradigm for modern LLM compression. Notable examples include models such as Alpaca and Vicuna, which were trained on datasets generated by larger models like GPT-4, achieving competitive performance at a fraction of the computational cost [80, 81]. Rationale-based distillation methods further enrich this approach by having the teacher generate not only final answers but also intermediate reasoning steps, enabling the student to internalize the teacher's problem-solving process rather than merely imitating its outputs [82]. Task-specific strategies, such as decomposing complex reasoning into problem decomposition and solution phases, have also been shown to enhance the generalization and inference efficiency of compact student models [83]. Generative Context Distillation (GCD) offers another refinement,

enabling student models to internalize prompt-based instructions during training so that they no longer depend on explicit prompts at inference time, thereby reducing latency and improving deployment efficiency [42].

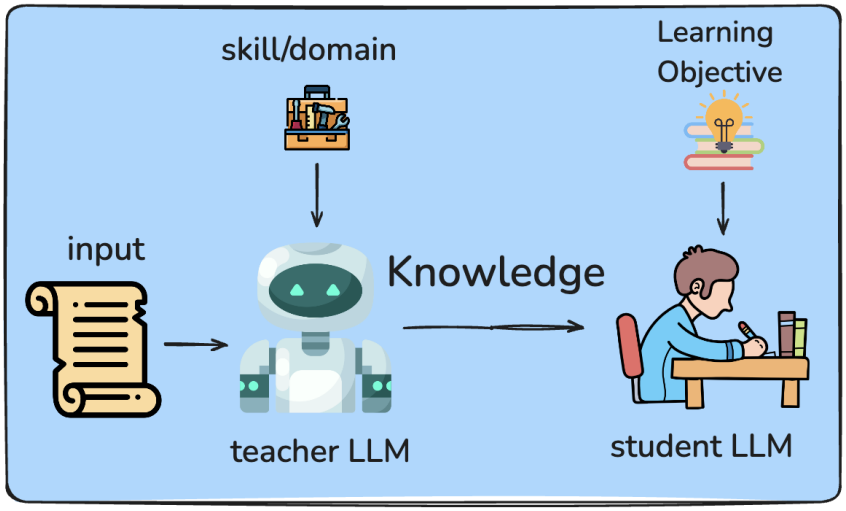

Figure 2.4: Overview of the data distillation process(Black Box approach): a teacher model generates a task-specific dataset that is subsequently used to fine-tune a smaller student model.

Despite its practical appeal, black-box distillation is not without challenges. The quality of the student model is fundamentally bounded by the quality of the teacher-generated data, making the data curation process critical. Capacity gaps between teacher and student can lead to situations where the student cannot fully absorb the complexity of the teacher's outputs, and semantic divergence between their respective embedding spaces may further hinder effective transfer. Collaborative embedding distillation and importance-aware ranking distillation have been proposed as strategies to bridge these gaps, allowing smaller models to align more closely with their larger counterparts [84].

2.7 Quantization in Large Language Models

Quantization is a technique to reduce the numerical precision of model parameters and activations, typically from 16- or 32-bit floating point to lower bit-width formats such as 8-bit or 4-bit integers. This compression reduces memory usage and speeds up inference, enabling large language models (LLMs) to run efficiently on hardware with limited resources [85, 86].

The main benefits of quantization include:

- **Reduced memory footprint**, allowing larger models or longer sequences on smaller GPUs.

- **Faster inference**, due to simpler and faster arithmetic operations.
- **Lower energy consumption**, important for edge devices and large-scale deployments.

Quantization can be applied after training (post-training quantization) or during training (quantization-aware training), with the latter generally achieving better accuracy [87]. However, aggressive quantization (e.g., 4-bit) may lead to some accuracy loss, requiring careful tuning and optimized hardware support [85].

Recent studies have shown that quantization, especially aggressive forms like 4-bit quantization, can lead to potential loss of accuracy and affect model confidence and calibration [88]. For example, 4-bit post-training quantization often introduces challenges due to activation distribution variance and outliers, which can degrade performance if not properly addressed [89].

However, recent research also demonstrates that well-designed quantization methods can maintain near-original performance while significantly reducing model size and computational cost. Techniques such as floating-point 4-bit quantization with optimized parameter search [89], and advanced compression schemes like QRazor [90], have achieved performance close to full precision models on large-scale benchmarks. Additionally, studies confirm that 4-bit quantized LLMs generally retain comparable perplexity and downstream task performance to their full-precision counterparts, with degradation becoming significant only below 4 bits [85].

Simplified illustration Quantization reduces the precision of model weights from high-precision floating-point numbers (e.g., 32-bit) to lower-bit integers (e.g., 4-bit), saving memory and speeding up inference. Consider a weight $w = 0.2345$ in a model layer where weights range from -1.0 to $+1.0$.

The quantization process for weights involves several steps. First, the **scale factor** S maps the floating-point range to the integer levels. For example, given a 4-bit quantization ($b = 4$), we have $2^b = 16$ discrete levels. The scale factor is calculated as

$$S = \frac{W_{\max} - W_{\min}}{2^b - 1} = \frac{1 - (-1)}{15} = \frac{2}{15} \approx 0.1333,$$

where $W_{\min}$ and $W_{\max}$ are the minimum and maximum representable weights.

Next, a weight value w is mapped to its integer representation Q through quantization:

$$Q = \text{round}\left(\frac{w - W_{\min}}{S}\right) = \text{round}\left(\frac{0.2345 - (-1)}{0.1333}\right) = 9.$$

Finally, dequantization reconstructs an approximate floating-point value from the quantized integer:

$$\hat{w} = S \times Q + W_{\min} = 0.1333 \times 9 - 1 = 0.2.$$

2.8 Agents

The concept of an intelligent agent has been central to artificial intelligence research for several decades, yet its fundamental formulation has remained remarkably stable despite profound technological transformations. Russell and Norvig define an intelligent agent as any system that can perceive its environment through sensors and act upon it via actuators in pursuit of goals [91]. This definition is intentionally broad, encompassing both simple reactive mechanisms—such as thermostats—and complex autonomous systems, including robots and software-based decision-making agents. The core principle is that agency is not defined by internal implementation, but by the observable relationship between perception, reasoning, and action.

Building on this perspective, Wooldridge and Jennings articulated four essential properties of intelligent agents: **autonomy, reactivity, proactiveness, and social ability** [92]. Autonomy refers to the agent's capacity to operate without continuous human control; reactivity denotes its ability to perceive and respond to environmental changes; proactiveness captures goal-directed initiative rather than passive reaction; and social ability involves structured interaction with other agents or humans. This formulation, often described as the "weak notion of agency," has become a foundational framework that continues to shape both theoretical and applied research in multi-agent systems.

Classical artificial intelligence further categorized agents into a hierarchy of increasing sophistication, ranging from simple reflex agents to learning agents capable of improving through experience [91]. However, the rise of large language models (LLMs) has fundamentally reconfigured this taxonomy. In contemporary LLM-powered systems, the language model functions as the central reasoning engine, supported by planning mechanisms, memory systems, and external tools [93]. Planning allows agents to decompose complex tasks into structured subgoals using techniques such as Chain-of-Thought, Tree of Thoughts, and self-reflection mechanisms like ReAct and Reflexion. Memory integrates short-term in-context reasoning with long-term retrieval from vector databases, enabling continuity across interactions. Tool use extends agent capabilities beyond text generation by enabling access to APIs, databases, and computational environments.

Recent work has also drawn an important distinction between *AI Agents* and *Agentic AI*, marking a transition from isolated autonomous systems to coordinated multi-agent ecosystems [94]. While traditional AI agents typically operate as single entities optimized

for bounded tasks, Agentic AI refers to architectures in which multiple specialized agents collaborate, share memory, and dynamically re-plan strategies. Frameworks such as AutoGen, CrewAI, and LangGraph exemplify this shift by introducing orchestration layers that manage role assignment, task decomposition, and inter-agent communication.

Table 2.1: Comparison between AI Agents and Agentic AI Systems

Feature	**AI Agents**	**Agentic AI**
Definition	Autonomous software systems designed to perform specific, bounded tasks	Coordinated systems composed of multiple AI agents collaborating toward complex, evolving goals
Autonomy Level	High autonomy within a predefined and narrow scope	High autonomy across dynamic, multi-step and adaptive workflows
Architecture	Typically single-agent, optionally tool-augmented	Multi-agent architectures with orchestration and coordination layers
Memory	Primarily short-term context buffers	Persistent episodic and semantic memory across tasks
Planning Horizon	Single-step or short sequential planning	Long-horizon planning with goal decomposition and iterative re-planning
Examples	AutoGPT, LangChain Agents, BabyAGI	CrewAI, AutoGen, MetaGPT

In practical applications, Retrieval-Augmented Generation (RAG) has become a dominant pattern for grounding agent behavior in external knowledge rather than relying solely on pre-trained representations [95]. ReAct integrates reasoning and action in a continuous loop of thought, tool invocation, and observation, while Reflexion introduces memory-based self-correction mechanisms that allow agents to learn from previous failures [96, 97]. These paradigms illustrate how modern agents blend reasoning, retrieval, and interaction into a unified decision-making process.

Despite these advances, significant limitations remain. Finite context windows constrain long-horizon reasoning, while multi-step planning is often brittle when confronted with unexpected errors. Hallucinations continue to undermine reliability, particularly in complex multi-hop scenarios. In multi-agent systems, coordination failures and error propagation present additional risks, motivating ongoing research in governance, safety, and alignment [93, 94]. These challenges suggest that while LLM-powered agents represent a major leap toward autonomous artificial intelligence, their deployment in high-stakes environments still requires rigorous evaluation and structured human oversight.

2.9 Evaluation Approaches: Semantic and Lexical Metrics

Evaluation constitutes a fundamental component in the study and development of large language models (LLMs), as it defines how model behavior is measured, compared, and interpreted. As LLMs have evolved from narrow text generation systems to versatile models capable of reasoning, instruction following, dialogue, and code synthesis, the complexity of evaluating their outputs has increased substantially. Traditional automatic metrics such as BLEU and ROUGE, which rely on n-gram overlap with reference texts, remain useful for assessing surface-level similarity. However, they often fail to capture deeper aspects of quality, such as semantic correctness, logical coherence, contextual appropriateness, and alignment with user intent [98, 99].

In response to these limitations, recent research has reframed evaluation as a semantic judgment problem. Under the paradigm commonly referred to as *LLM-as-a-Judge*, powerful language models are used to assess the outputs of other models through rubric-guided scoring, pairwise comparisons, or structured feedback [98, 100, 101]. By leveraging their internal representations of meaning and discourse, LLM-based evaluators can account for paraphrasing, implicit reasoning steps, and multi-criteria task requirements, often achieving closer alignment with human judgments than purely lexical metrics. This approach has proven particularly valuable in open-ended tasks where multiple valid responses may exist and strict reference matching is insufficient.

Nevertheless, semantic evaluation using LLM-based judges introduces its own methodological challenges. Because the evaluator is itself a generative model, assessments may be sensitive to prompt design, model configuration, and version-specific behavior, raising concerns about consistency, bias, and reproducibility [99, 101]. To mitigate these issues, recent studies have emphasized the importance of human-centered evaluation protocols, explicit scoring rubrics, checklist-style criteria, and meta-evaluation benchmarks that calibrate automated judgments against expert-annotated ground truth [102, 103, 104].

Alongside semantic approaches, lexical and statistical evaluation metrics continue to provide deterministic and interpretable measurements of textual correspondence. Such metrics offer stable baselines that are independent of model subjectivity and remain valuable for ensuring reproducibility across experimental settings. The coexistence of semantic and lexical paradigms reflects an emerging consensus in the literature: robust evaluation of LLMs benefits from balancing meaning-level assessment with transparent, model-agnostic similarity measures [98, 99, 103].

2.9.1 LLM as a Judge (Semantic Evaluation)

Semantic evaluation is the process of systematically assessing the extent to which outputs generated by large language models (LLMs) accurately convey intended meaning, contextual relevance, and coherence beyond mere lexical overlap. Unlike traditional metrics such as BLEU or ROUGE, which rely heavily on surface-level textual similarity, semantic evaluation emphasizes deeper linguistic qualities, capturing nuances in semantics, creativity, and intent alignment [105, 106].

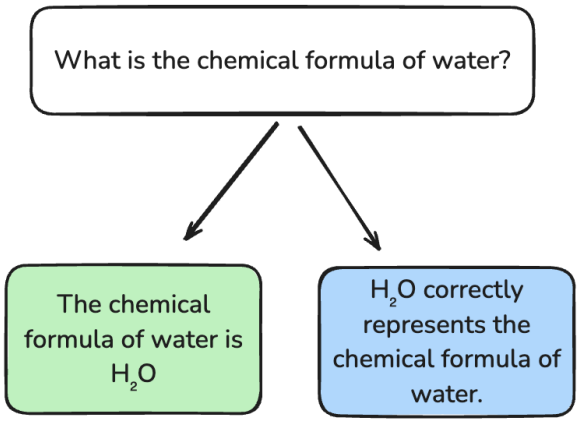

Figure 2.5: Semantically equivalent answers with differing lexical forms

The primary rationale behind semantic evaluation is addressing the inadequacy of conventional approaches in accurately reflecting the qualitative performance of advanced generative models. While traditional metrics assess performance through token matching, they often overlook subtler linguistic and cognitive dimensions—such as inferential correctness, consistency with factual knowledge, and appropriateness within a given context [107]. As LLMs are increasingly employed in nuanced tasks like open-domain question answering, dialogue systems, and creative text generation, robust semantic evaluation frameworks become essential.

Semantic evaluation techniques typically employ sophisticated methodologies to quantify semantic correspondence and relevance, leveraging advanced metrics and evaluation paradigms. Recent trends involve utilizing LLMs themselves as evaluators, a technique termed "LLM-as-a-Judge," which exploits the inherent semantic understanding capabilities of these models. This method enables more nuanced assessments by contextualizing evaluations within richer semantic frameworks, thus providing insights that are more aligned with human judgment and linguistic intuition [105, 108].

The motivation for adopting semantic evaluation is grounded in the need to reliably measure and improve model outputs concerning semantic accuracy and cognitive plau-

sibility. This ensures the outputs of generative models are not only superficially correct but also contextually meaningful, logically coherent, and practically useful, effectively aligning model behavior with human expectations and real-world applications [109, 110].

Although LLM-as-a-judge provides a practical and scalable evaluation mechanism, it is not infallible. Empirical studies show that these models can struggle in specialized or highly nuanced domains, where agreement with human experts is not always consistent [111]. Moreover, LLM judges may exhibit sensitivity to prompt phrasing and other systematic biases, which can introduce variability in their assessments [112].

For this reason, deterministic and reproducible metrics remain essential in robust evaluation pipelines. Complementing LLM-based semantic judgment with objective measures such as BM25 and formatting compliance helps mitigate subjective variability, increases transparency, and strengthens the overall reliability of experimental results.

2.9.2 BM25 (Best Matching 25)

BM25, also known as Best Matching 25, is a probabilistic information retrieval ranking function widely used in search engines and document retrieval systems. It belongs to the Okapi family of retrieval models and is designed to estimate the relevance of a document with respect to a given query based on lexical overlap rather than semantic similarity [113].

At a conceptual level, BM25 assigns a relevance score to each document by considering three key factors: (i) how often a query term appears in the document, (ii) how rare that term is across the entire corpus, and (iii) the length of the document. Terms that occur frequently in a given document but are rare in the corpus receive higher weights, while very common terms contribute less to the final score. Additionally, BM25 normalizes for document length to prevent longer texts from being unfairly favored simply because they contain more words.

Formally, given a query Q containing terms $q_1, q_2, ..., q_n$, the BM25 score for a document D is computed as:

$$\text{BM25}(Q, D) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D)\,(k_1 + 1)}{f(q_i, D) + k_1 \left(1 - b + b\frac{|D|}{\text{avgDL}}\right)}$$

where:

- $f(q_i, D)$ is the frequency of term q_i in document D;
- $|D|$ is the length of document D in tokens;
- avgDL is the average document length in the corpus;
- k_1 is a hyperparameter controlling term saturation (typically between 1.2 and 2.0);

- b is a hyperparameter controlling length normalization (commonly set to 0.75).

The inverse document frequency (IDF) component weights rare terms more heavily and is commonly defined as:

$$\mathrm{IDF}(q_i) = \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

where:

- N is the total number of documents in the corpus;
- $n(q_i)$ is the number of documents containing term q_i.

Unlike embedding-based similarity methods, BM25 operates purely on token matching and statistical properties of the corpus. This makes it fast, deterministic, interpretable, and highly reproducible, which explains its continued relevance in modern retrieval systems, even in pipelines that later incorporate neural models.

In contemporary Retrieval-Augmented Generation (RAG) systems, BM25 is frequently used either as a standalone retriever or in hybrid configurations alongside dense retrieval methods. Although it cannot capture deep semantic relationships between words, its strong performance on lexical matching and named entities makes it particularly valuable in structured domains, keyword-heavy documents, and evaluation settings that require stable and transparent similarity metrics.

In this thesis, BM25 is employed as a complementary, deterministic measure of lexical alignment between extracted quotes and reference quotes. This allows us to contrast neural, LLM-based semantic evaluation with a classical retrieval metric that is independent of any generative model, providing a more robust and interpretable assessment of quote quality.

Chapter 3

Related Work

3.1 Related Work

This chapter reviews recent advances that motivate and contextualize the proposed approach to quote extraction in RAG pipelines. We begin by examining the persistent challenge of noisy context in retrieval-augmented systems (§3.1.1), which degrades generation quality even when relevant documents are successfully retrieved. We then discuss RAFT as a prominent method for training models to identify supporting evidence while generating answers (§3.1.2), followed by recent applications of knowledge distillation that enable capability transfer from large to small models (§3.1.3). We review the growing adoption of Small Language Models in structured, task-oriented settings (§3.1.4), and conclude with evaluation methodologies that combine semantic judgment with deterministic metrics (§3.1.5). Together, these areas establish the conceptual foundations for using distillation to create specialized quote-extraction models that address RAG's noise bottleneck.

3.1.1 Noisy Context in Retrieval-Augmented Generation

A fundamental challenge in Retrieval-Augmented Generation is that retrieved documents often contain substantial amounts of content that is irrelevant or only tangentially related to the user's query. Even when the retrieval stage successfully identifies documents that contain the answer, those documents typically include additional paragraphs, sections, or passages that do not contribute to answer generation and may actively degrade output quality by introducing distracting or contradictory information. This phenomenon, commonly referred to as **contextual noise**, has been identified as a central bottleneck limiting the reliability and accuracy of RAG systems [29].

Recent work has demonstrated that even strong language models may hallucinate or be distracted when retrieved passages are off-topic, contradictory, or only partially relevant to the question at hand [29]. The presence of noisy context forces the model to perform simultaneous tasks: distinguishing relevant from irrelevant information while also reasoning over the relevant portions to generate an answer. This dual cognitive load contributes to the task interference phenomenon discussed in the background section on multi-step reasoning, where models experience performance degradation when handling multiple objectives within a single context [34].

To address this challenge, several approaches have emerged that insert explicit denoising or filtering stages between retrieval and generation. Fine-grained filtering modules attempt to locate and retain only answer-bearing sentences from long documents, while information-bottleneck style objectives learn compressed intermediate representations that preserve answer-relevant information while discarding noise [30, 31]. These methods operate on the principle that reducing contextual clutter before generation enables cleaner, more focused reasoning by the downstream model.

A complementary line of work focuses on reducing noise at the retrieval or control stage rather than applying post-hoc filters. Adaptive frameworks decide when and how much to retrieve based on the information completeness of the query, thereby avoiding unnecessary retrieval of noisy content [32, 29]. Domain-specific retrievers can be trained to filter out task-irrelevant signals, such as spurious patterns in financial documents, before passing context to the generator [33]. Together, these approaches treat RAG as a multi-stage pipeline, comprising retrieval, filtering, compression, and generation, with each stage explicitly optimized to minimize contextual noise while preserving sufficient evidence for accurate grounded answers [31].

Beyond noise reduction, recent developments have explored architectural innovations to improve how models process multiple retrieved documents. Techniques like Superposition Prompting [114] enable models to reason over multiple documents simultaneously by encoding them in parallel paths within a single prompt, avoiding the typical blending of contexts seen in concatenation-based RAG and reducing interference between sources. MiniRAG [115] adapts the RAG architecture specifically for small language models by simplifying retrieval pipelines, using lightweight rerankers, and limiting context length, making it suitable for on-device inference. Collab-RAG [116] proposes a cooperative strategy in which a smaller model decomposes complex questions and delegates subtasks to a larger model, improving performance in multi-hop question answering scenarios.

Other advancements include memory-aware architectures that maintain persistent representations of prior interactions. DH-RAG [117] extends traditional RAG by incorporating memory that stores previous retrieved passages, enhancing coherence in multi-turn or

session-based applications. CDF-RAG [118] introduces causal dynamic feedback mechanisms that refine queries through causal graphs and multi-hop reasoning, moving beyond correlation-based retrieval toward more principled causal reasoning patterns.

Despite these advances, the fundamental tension persists: retrieving broader context improves recall of relevant information but introduces noise, while narrower retrieval risks missing critical evidence. This trade-off motivates dedicated filtering or extraction stages that operate after retrieval but before generation, isolating answer-relevant passages to reduce cognitive load on downstream models while preserving factual grounding.

3.1.2 RAFT: Learning to Extract and Reason

Retrieval-Augmented Fine-Tuning (RAFT) emerged as a response to the observation that not all models, especially smaller ones, are capable of effectively handling large contexts and reasoning simultaneously [1, 119]. Even when provided with appropriate context, models often fail to generate coherent or accurate answers, revealing a gap in their ability to integrate retrieval with reasoning [120, 121, 122]. RAFT addresses this by fine-tuning LLMs specifically for RAG scenarios in domain-specific settings, with particular emphasis on healthcare, legal analysis, and technical documentation, where large specialized datasets are required [123, 124].

The core innovation of RAFT is training models to "think while quoting", that is, to simultaneously perform chain-of-thought reasoning and extract relevant portions of the retrieved context [1]. As illustrated in Figure 3.1, the RAFT methodology transforms a task structured as *context + question* into a comprehensive output comprising reasoning steps, relevant quotes, and a final answer. This dual focus enables the model to dynamically identify key parts of the input text, synthesize their meaning, and produce a well-reasoned conclusion. By teaching models to reason and quote in tandem, RAFT enhances their ability to operate effectively in RAG settings, bridging the gap between retrieval and generation to deliver more accurate, context-aware responses.

However, RAFT's approach presents several characteristics that may limit its applicability in certain scenarios. First, by training models to perform reasoning and quote extraction simultaneously within a single generation pass, RAFT increases the cognitive complexity of each inference step. This joint optimization may contribute to the task interference phenomenon identified in the background section, where models experience performance degradation when handling multiple objectives concurrently [34]. Second, the RAFT methodology relies on synthetic distractor documents during training to teach the model to distinguish relevant from irrelevant context. While this approach provides controlled training signals, it may not fully capture the diversity and subtlety of noise patterns encountered in real-world retrieval systems, where irrelevance manifests along

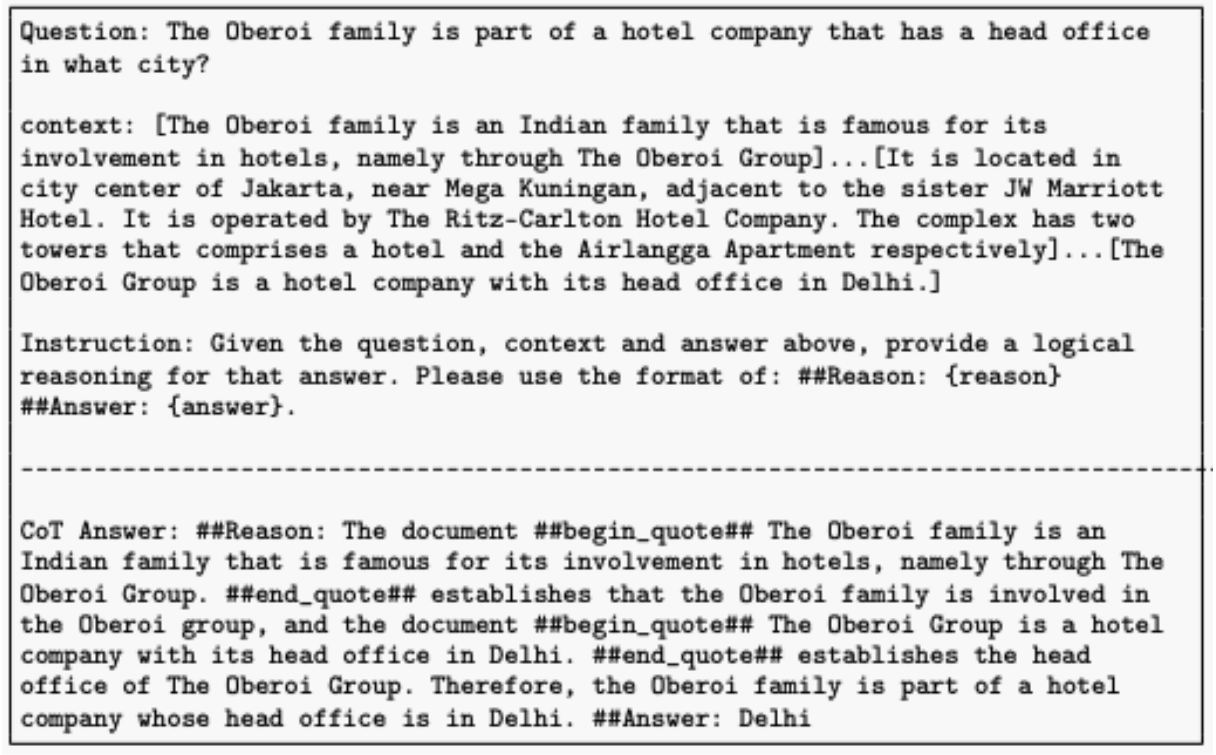

Figure 3.1: Example of how RAFT methodology works [1]

a spectrum rather than as discrete distractor/oracle categories. Third, RAFT typically assumes deployment of the same large model used during training, without investigating whether the quote-extraction capability can be isolated and distilled into smaller, specialized models optimized exclusively for evidence identification.

These observations suggest an alternative architectural possibility: rather than joint training on reasoning-plus-extraction, a dedicated extraction stage could filter context before generation, potentially reducing task interference and enabling modular optimization of each component. Furthermore, if quote extraction can be formulated as a bounded, schema-oriented task (as opposed to open-ended reasoning) it may be particularly amenable to distillation into compact models, following the principles discussed in the background section on task-specific fine-tuning for SLMs.

3.1.3 Knowledge Distillation for Task-Specific Capabilities

As discussed in the background section, knowledge distillation enables the transfer of capabilities from large teacher models to smaller student models through either white-box approaches (which require access to internal representations) or black-box approaches (which treat the teacher as a data generator). Recent work has demonstrated that black-

box distillation is particularly effective for transferring narrow, well-defined capabilities to compact models, especially when the target task is structured and does not require broad world knowledge [74, 79].

The MiniPLM framework [125] exemplifies the data-generation paradigm by performing offline teacher inference to create refined training distributions. This approach allows multiple student models to benefit from a single teacher without incurring repeated inference costs during training, making it especially practical when working with proprietary or API-based teachers. The method demonstrates that effective pre-training of SLMs can be achieved across different model families by leveraging teacher-generated data to shape the training distribution.

Similarly, Yam and Paek [126] explore various methods to enhance knowledge distillation specifically for small language models, showing that effective distillation can be achieved even with significantly smaller teacher-student pairs. Their experiments with models like DistilledGPT-44M and ContrastiveLlama-58M demonstrate that compact models can maintain competitive performance while reducing both training time and parameter count, suggesting that the distillation paradigm scales down effectively when the target capability is appropriately scoped.

Ballout et al. [127] present a particularly relevant approach that extracts influential tokens from the teacher's decision-making process and provides them as rationales to the student. This technique, which focuses on identifying and transferring the most salient aspects of the teacher's behavior rather than attempting to replicate all internal representations, has proven effective across diverse datasets and outperforms standard fine-tuning methods. The success of this selective knowledge transfer reinforces the principle that distillation works best when targeting specific, extractable patterns rather than attempting wholesale model compression.

The Preference-Aligned Distillation (PAD) framework [128] extends this idea further by modeling the teacher's preference knowledge as a probability distribution over all possible preferences, providing nuanced supervisory signals that guide the student's learning process. By encoding not just what the teacher produces but also what it prefers or disprefers, PAD enables more fine-grained alignment between teacher and student behavior.

DistiLLM [129], introduced in 2024, demonstrates significant efficiency gains through a combination of skew-KL divergence loss and an adaptive off-policy mechanism that reuses student-generated outputs during training. This framework achieves up to 4.3× speedup in distillation while preserving task accuracy, making it particularly attractive for scenarios where training time and computational resources are constrained. The efficiency gains stem from reducing redundant teacher queries and enabling the student to learn from its own generations when they are sufficiently aligned with teacher expectations.

A common thread across these recent advances is the effectiveness of distilling **narrow, well-defined capabilities** into compact models.When the target task is structured and bounded, as in function calling [130], structured text processing, or domain-specific question answering, small models can achieve near-teacher performance with dramatically fewer parameters, precisely because they need not maintain encyclopedic world knowledge or handle arbitrary open-ended reasoning. This principle has been validated across specialized domains including tool use, controlled generation, and constrained extraction tasks, suggesting that task-focused distillation offers a practical path to deployable, efficient models for specific pipeline components. [131, 132]

3.1.4 Small Language Models for Specialized Tasks

The position paper *Small Language Models are the Future of Agentic AI* argues that the dominant workloads in agentic systems are not open-ended conversation but repetitive, schema-constrained, and tool-heavy tasks such as routing, function calling, form filling, and simple planning [133, 7]. For these workloads, small language models in the 100M–8B parameter range provide sufficient capability while offering substantial efficiency advantages, making them more suitable as default components than very large models [133, 50]. The paper advocates heterogeneous architectures in which SLMs handle the majority of routine steps while large models are invoked only as teachers, reviewers, or fallbacks when richer reasoning or broad general knowledge is required [133, 134].

Recent work demonstrates several concrete applications of this principle. SLMs can be trained as reliable function callers that orchestrate APIs and device functions on edge hardware, avoiding dependence on cloud-based LLMs. The TinyAgent framework, for instance, shows how 1–7B parameter models can be fine-tuned on curated function-calling datasets to parse user requests, select appropriate tools, and emit structured arguments with low latency and minimal energy consumption [130, 135]. In domain-focused agents, SLMs serve as planners and coordinators within larger pipelines that include retrieval, simulation, or analytics components, decomposing tasks and integrating tool outputs to support experts in areas like agriculture, industry, or genomics [50, 57].

SLMs are also increasingly deployed in RAG systems to handle structured subtasks that precede or follow retrieval. In educational assistants and specialized help systems, SLMs focus on grounding, question interpretation, and controlled answer synthesis over retrieved documents rather than memorizing knowledge in their weights [55, 136]. This division of labor makes it feasible to deploy privacy-preserving, locally hosted systems that serve many users concurrently on modest hardware [7].

A particularly relevant architectural pattern is the use of SLMs as preprocessing filters or gatekeepers in multi-stage pipelines. SLMs can act as local filters that screen sensitive

or personally identifiable information from user inputs before they are sent to cloud-based large models [137, 138]. This lightweight preprocessing layer enhances privacy and data protection while preserving downstream response quality. More broadly, this pattern demonstrates that SLMs excel at bounded, deterministic operations, like format validation, entity recognition, or structured extraction, that require reliability and predictable output rather than open-ended creativity.

The *SLM-default, LLM-fallback* design has emerged as a common pattern in recent literature, wherein a locally hosted small model executes routine operations and a larger remote model is invoked only when complex reasoning or ambiguous interpretation is required [134, 139]. In practice, the small model assesses task complexity and selectively escalates difficult cases, optimizing computational resource allocation while maintaining robustness against remote model failures.

3.1.5 Evaluation Methods for Extraction and Generation Tasks

Recent advances in large language models have led to a paradigm shift in evaluation methodologies across various domains. Instead of relying solely on traditional metrics or expert human judgments, a growing body of work leverages LLMs as judges or committees of LLMs to assess the quality, relevance, or feasibility of outputs [98, 99]. This approach offers scalability, cost-effectiveness, and the ability to handle diverse and complex evaluation scenarios where multiple valid outputs may exist.

Comprehensive surveys by Gu et al. [108] and Li et al. [105] systematically review the reliability, methodologies, and practical applications of LLMs-as-judges, highlighting their increasing adoption as alternatives to expert-driven evaluations. The flexibility and scalability of LLM-based judges make them particularly valuable for open-ended tasks where strict reference matching is insufficient and semantic equivalence matters more than lexical overlap.

Specific applications demonstrate the practical utility of this paradigm. Recent work explores the use of LLMs as judges in multi-agent collaboration systems, where they select the most feasible solution among multiple candidates [140]. JudgeLRM, a family of judgment-oriented LLMs trained with reinforcement learning, demonstrates that specialized judge models can outperform general-purpose models in complex reasoning and judgment tasks [141]. Empirical studies provide large-scale benchmarks of LLM-generated relevance judgments, comparing different approaches and analyzing systematic biases [110].

Auto-Arena [142] presents a fully automated, agent-based evaluation framework in which an examiner LLM generates diverse questions, candidate models engage in head-to-head debates, and a committee of LLM judges deliberates to select winners. By replacing

static test suites and manual grading with LLM agents, this approach achieves 92% rank-correlation with human judgments on state-of-the-art models while reducing human effort to zero.

However, LLM-based evaluation is not without limitations. Empirical studies show that LLM judges can struggle in specialized or highly nuanced domains where agreement with human experts is inconsistent [111]. Moreover, these models exhibit sensitivity to prompt phrasing and other systematic biases, introducing variability in assessments [112, 109]. To mitigate these issues, recent studies emphasize the importance of human-centered evaluation protocols, explicit scoring rubrics, checklist-style criteria, and meta-evaluation benchmarks that calibrate automated judgments against expert-annotated ground truth [102, 103, 104].

For these reasons, deterministic and reproducible metrics remain essential in robust evaluation pipelines. Complementing LLM-based semantic judgment with objective measures, such as BM25 for lexical alignment and format compliance scores for structural correctness, helps mitigate subjective variability, increases transparency, and strengthens overall experimental reliability. The coexistence of semantic and lexical evaluation paradigms reflects an emerging consensus: robust assessment benefits from balancing meaning-level judgment with transparent, model-agnostic similarity measures [98, 99, 103].

Chapter 4

Proposed Model: LLMQuoter

The core objective is to train a compact language model capable of identifying the most semantically relevant excerpts (referred to as *quotes*) from a given textual context, such that these excerpts directly support accurate answer generation in downstream tasks. Rather than passing the entire retrieved context to the generator, the proposed approach interposes a lightweight extraction step that filters out irrelevant or distracting content, thereby reducing the cognitive burden on the downstream model and improving factual grounding.

The methodology unfolds in five stages: (i) formalization of the quote extraction task as a distillation problem; (ii) automated dataset generation using a teacher LLM; (iii) parameter-efficient fine-tuning of compact student models on the curated dataset; (iv) a rigorous evaluation framework combining semantic and lexical metrics; and (v) a controlled comparison demonstrating the practical utility of quote extraction over full-context input. Together, these stages define a complete pipeline from data curation to empirical validation.

4.1 Problem Formalization

Let us consider a dataset of instances, denoted by $D = \{(C, Q, A)\}$, where C is a large unstructured text context, Q is a user-specified natural language question, and A is the expected (reference) answer.

The task is to train a model capable of identifying a minimal subset of sentences $\mathcal{R} \subset C$ that are semantically aligned with A, given the pair (Q, C). The extracted quotes should serve as an evidential basis that enables accurate downstream answer generation.

A critical aspect of this formalization is the **asymmetry between teacher and student inputs**. During dataset construction, the teacher model has access to the triplet (Q, A, C), that is, it knows both the question and the expected answer when selecting

supporting quotes. The student model, by contrast, is trained to produce the same quote selections given only (Q, C), without access to the answer A. This asymmetry is deliberate, since at inference time the answer is unknown, so the student must learn to identify answer-supporting evidence from the question and context alone. In effect, the distillation process transfers the teacher's ability to recognize answer-relevant passages into a model that operates under realistic deployment conditions where the answer has not yet been generated.

To achieve this, we employ a teacher-student distillation framework in which a large LLM generates quote-level supervision, and a compact student model is fine-tuned to replicate this behavior efficiently.

4.2 LLM Distillation for Dataset Generation

The dataset construction process relies on one or more high-capacity teacher models, collectively denoted as:

$$f_{\text{high}} : (Q, A, C) \to \mathcal{R} \tag{4.1}$$

where $\mathcal{R}$ is the set of semantically relevant quotes extracted from context C that support answer A. In practice, f_{high} may represent a single model or an ensemble of complementary teacher models orchestrated through a pipeline framework, allowing the distillation process to benefit from the strengths of multiple architectures.

This procedure yields a curated reference dataset:

$$\mathcal{D}_{\text{ref}} = \{(Q, A, C, \mathcal{R}) \mid \mathcal{R} = f_{\text{high}}(Q, A, C)\} \tag{4.2}$$

which serves as supervision for fine-tuning a smaller model f_{small}.

Rather than relying on informal or ad-hoc prompts, the teacher model follows a structured instruction template in which it is explicitly asked to return only delimited quotes that directly support the answer, using a fixed boundary format (`##begin_quote## ... ##end_quote##`). This standardized output format serves two purposes: it facilitates automatic parsing of quotes during both training and evaluation, and it establishes a formatting convention that the student model must internalize, ensuring that extracted quotes can be programmatically separated from the rest of the response in a deployed RAG pipeline.

The decision to use simple text delimiters rather than structured JSON was motivated by recent evidence that format restrictions can degrade reasoning quality. Constraining generation to rigid schemas like JSON reduces performance on complex knowledge-

intensive tasks [143, 144]. Additionally, enforcing JSON correctness introduces operational overhead, since recent work documents nontrivial failure rates, added validation complexity, and increased latency when strict structural constraints are imposed [145, 146]. In contrast, delimiter-based formats are trivial to parse, require no schema validation, and fail gracefully when malformed. This approach aligns with RAFT [1], which similarly employs delimiters for quote extraction, balancing machine readability with minimal constraints on model reasoning.

Because the quality of the student model is fundamentally bounded by the quality of the teacher-generated data [74], manual review of a representative sample of the curated dataset is an integral part of this stage. This inspection verifies that the extracted quotes are concise, semantically aligned with the reference answers, and free from systematic errors introduced by the teacher.

4.3 Fine-Tuning the Quote Extraction Model

The distilled model f_{small} is trained to predict the quote set $\mathcal{R}$ from inputs (Q, C), without access to the answer A:

$$f_{\text{small}} : (Q, C) \to \mathcal{R} \tag{4.3}$$

Training proceeds by presenting instances from $\mathcal{D}_{\text{ref}}$ as input pairs (Q, C), with the corresponding teacher-generated quotes $\mathcal{R}$ as targets. The learning objective encourages the model to identify which passages in C belong to $\mathcal{R}$, effectively learning to replicate the teacher's extraction behavior under the more constrained inference-time setting.

To make fine-tuning practical for compact models with limited computational budgets, the training procedure employs **Low-Rank Adaptation (LoRA)** [69] in combination with quantized base weights, as discussed in the background sections on parameter-efficient fine-tuning and quantization. In this configuration, the original model parameters are frozen in a low-precision format (e.g., 4-bit integers), and only the injected low-rank adapter matrices are updated during training in higher precision (e.g., FP16). This hybrid strategy reduces GPU memory consumption by approximately four-fold compared to full-precision fine-tuning [147], enabling the entire training pipeline to execute on a single GPU in a matter of minutes. The LoRA adapters are applied to both the attention and feed-forward layers of the transformer architecture, following empirical evidence that updating both components yields stronger performance than modifying attention projections alone [147].

A deliberate aspect of the experimental design is the evaluation of the distillation approach across multiple student models of varying capacity. As discussed in the back-

ground section on Small Language Models, the SLM regime, roughly 10^8 to 5×10^9 parameters [7], spans a wide performance to efficiency spectrum. By selecting student models at distinct points along this spectrum (ranging from approximately 1B to 4B parameters), the methodology enables a systematic analysis of how model scale influences extraction quality, formatting compliance, and practical deployability. This multi-model design tests whether the distillation approach generalizes across architectures and parameter budgets, or whether a minimum model capacity is required to achieve acceptable quote extraction performance.

4.4 Evaluation Framework and Metrics

To assess the performance of the quote extraction models, we adopt a hybrid evaluation strategy that combines semantic judgment via LLM-based evaluation with deterministic lexical and structural metrics. This multi-faceted approach ensures that extracted quotes are evaluated not only for content quality but also for format compliance and reproducibility.

Let R_{model} denote the set of quotes predicted by the model and R_{ref} the curated reference quotes. We compute:

$$P = \frac{|R_{\text{model}} \cap R_{\text{ref}}|}{|R_{\text{model}}|} \tag{4.4}$$

$$R = \frac{|R_{\text{model}} \cap R_{\text{ref}}|}{|R_{\text{ref}}|} \tag{4.5}$$

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{4.6}$$

However, strict string matching often penalizes correct extractions that are merely paraphrased or slightly reformulated. A model may extract a passage that conveys the same supporting evidence as the reference but with different sentence boundaries or minor lexical variation, resulting in a false negative under exact matching. This limitation motivates the use of semantic evaluation as the primary assessment mechanism.

The central evaluation mechanism employs a state-of-the-art language model as an automated semantic judge. The judge operates through pairwise comparison: given a candidate quote from the model's output and the set of reference quotes, the judge determines whether the candidate is semantically present among the references, returning a binary or graded match score. This process is performed in both directions to compute precision and recall independently.

Concretely, **semantic precision** measures what fraction of the model's extracted quotes are supported by the reference set. For each quote $r \in R_{\text{model}}$, the judge receives both r and the full set of reference quotes R_{ref}, and is asked to determine whether r conveys information that is substantively contained in R_{ref}. The precision score is then:

$$P_s = \frac{\sum_{r \in R_{\text{model}}} \text{Judge}(r, R_{\text{ref}})}{|R_{\text{model}}|} \tag{4.7}$$

Semantic recall measures the converse, namely what fraction of the reference quotes are captured by the model's output. For each reference quote $r \in R_{\text{ref}}$, the judge evaluates whether r is semantically represented in R_{model}:

$$R_s = \frac{\sum_{r \in R_{\text{ref}}} \text{Judge}(r, R_{\text{model}})}{|R_{\text{ref}}|} \tag{4.8}$$

The **semantic F1-score** combines both into a single balanced measure:

$$F_{1s} = 2 \cdot \frac{P_s \cdot R_s}{P_s + R_s} \tag{4.9}$$

This approach captures semantic equivalence that would be missed by strict string matching, for instance, when the model extracts a slightly different sentence boundary or a paraphrased passage that conveys the same supporting evidence. By delegating the matching decision to a capable language model, the evaluation accounts for paraphrasing, partial overlap, and implicit semantic correspondence, achieving closer alignment with human judgment than purely lexical metrics [105, 106]. To ensure independence, the LLM judge must not be the same model used during the distillation stage, preventing circular evaluation bias.

Formatting Score is a binary metric (0 or 1) that verifies whether the model's output strictly adheres to the required quote delimitation format (## **begin_quote** ## **...** ##**end_quote**##). The evaluation procedure strips whitespace from the raw response and attempts to reconstruct it exclusively from the extracted quote blocks. If the reconstructed text exactly matches the original output, the score is 1.0; otherwise, it is 0.0.

This metric is particularly important from a deployment perspective, since in a production RAG pipeline, extracted quotes must be programmatically parseable to be passed as input to the downstream generator. A model that produces high-quality quotes but fails to wrap them in the expected delimiters is functionally unusable without additional post-processing. Formatting Score thus evaluates instruction-following compliance and structural reliability independently of content quality.

BM25 Score provides a deterministic, model-agnostic measure of lexical alignment between extracted and reference quotes. A BM25 index is constructed from the tokenized

reference quotes, and each model-generated quote is scored against this index using the BM25Okapi ranking function [113]. The resulting score is normalized by the maximum achievable BM25 value (obtained by scoring the reference quotes against themselves), yielding a value between 0 and 1. A score near 1.0 indicates strong token-level overlap, while values near 0.0 indicate weak lexical correspondence.

By operating purely on token statistics without any learned representations, BM25 Score complements the semantic evaluation by providing a stable, reproducible baseline that is independent of any generative model's subjective judgment. The combination of semantic F1, Formatting Score, and BM25 Score ensures that evaluation captures content quality, structural compliance, and lexical fidelity simultaneously.

4.5 Demonstrating the Utility of Quote Extraction

The evaluation framework described above measures how well the student model reproduces the teacher's quote selections. However, a separate and equally important question is whether providing extracted quotes, rather than the full retrieved context, actually improves downstream answer generation. This question directly addresses the RAG noise bottleneck identified in the background: if retrieved contexts contain irrelevant or distracting passages, a generator that receives only the relevant excerpts should perform better than one that must process the entire context [29, 30].

To test this hypothesis, we conduct a controlled comparison using a baseline generator model f_{base} under two conditions. In the first condition, referred to as quote-only input, the generator receives only the curated reference quotes:

$$f_{\text{base}} : (Q, \mathcal{R}_{\text{ref}}) \to A_{\text{quote}} \tag{4.10}$$

In the second condition, referred to as full-context input, the generator receives the entire retrieved context:

$$f_{\text{base}} : (Q, C) \to A_{\text{full}} \tag{4.11}$$

The generated answers under both conditions are evaluated using semantic accuracy, computed by the same LLM judge used in the quote evaluation stage:

$$S_{\text{acc}} = \frac{\sum_{a \in A_{\text{base}}} \text{Judge}(a, A_{\text{ref}})}{|A_{\text{ref}}|} \tag{4.12}$$

By comparing S_{acc} across the two settings, we empirically validate whether quote extraction reduces contextual noise, lowers the cognitive load on the generator, and improves answer quality. This comparison is conducted across multiple baseline models of different

sizes, enabling analysis of whether the benefit of quote extraction is uniform or disproportionately large for smaller, more resource-constrained models, a finding that would further reinforce the practical value of the proposed pipeline for SLM-based deployments.

4.6 Experimental Setup

This section describes the concrete experimental setup that operationalizes the methodology presented above, including datasets, distillation procedure, fine-tuning configuration, and evaluation protocol. An overview of the entire process, from data distillation to evaluation, is illustrated in Figure 4.1. The experiments designed to validate the effectiveness of using relevant quotes instead of full context are depicted in Figure 4.2. The code utilized in this work is available on GitHub[1]. Concrete examples of the experimental results can be found in the appendix for further clarification. In addition to the experimental pipeline described above, a simple web application was developed to enable interactive testing of the fine-tuned models in a user-friendly environment, whose code[2] is also available.

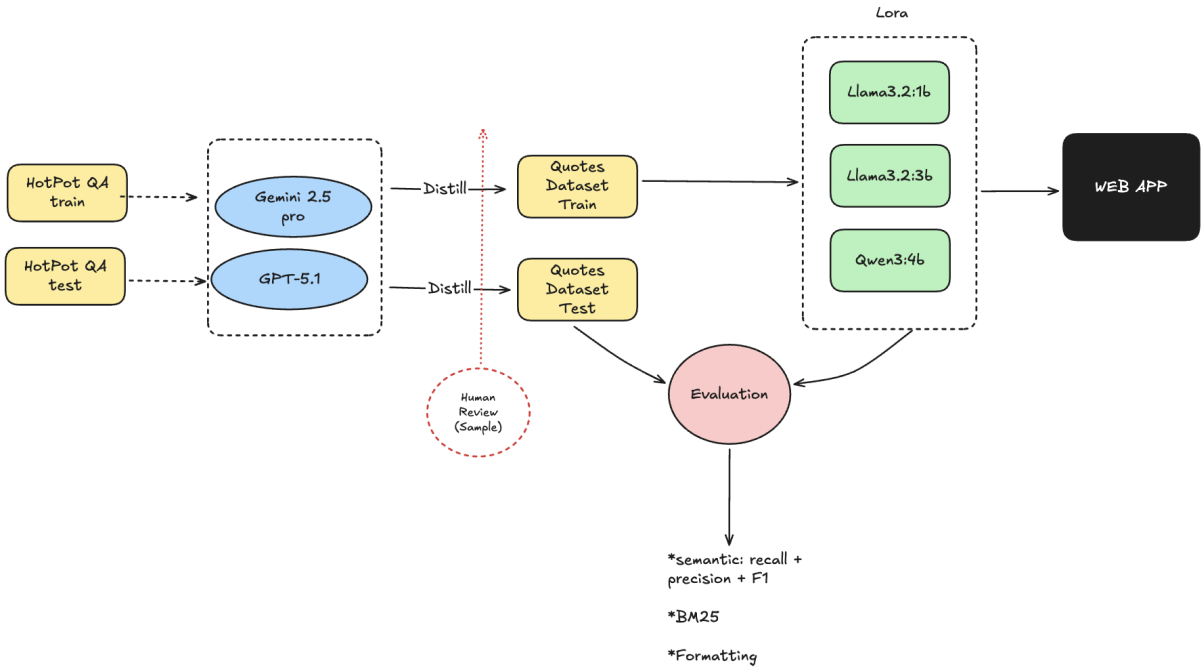

Figure 4.1: Experiment Overview

Datasets. Our method was evaluated on the HotpotQA dataset (Yang et al., 2018), an open-domain question-answering benchmark derived from Wikipedia that covers common-

[1]https://github.com/yurifacanha/llmquoter
[2]https://github.com/yurifacanha/llmquoter-web

knowledge topics such as movies, sports, and general trivia. Each sample consists of three elements: a **question**, a large **context** containing multiple supporting passages, and the corresponding **answer**.

To balance experimental rigor and computational feasibility, we selected a random subset of 25,000 samples for training and retained 600 samples as a held-out test set. The test set was used exclusively for evaluating quote extraction quality and for assessing the impact of using extracted quotes instead of full context in downstream question answering.

Table 4.1: Summary of dataset characteristics used in the experiments.

Attribute	**Value**
Dataset Name	HotpotQA
Total Samples Used	25,600
Test Set Size	600
Training Size	25,000
Source	Wikipedia
Topics	Common knowledge

Data Distillation. The distillation process was conducted using a hybrid teacher setup composed of **Gemini 2.5 Pro** and **GPT-5.1**. These models acted jointly as high-capacity teachers f_{high}, with LangChain orchestrating the overall pipeline. For each sample in both training and test sets, the teacher models were prompted to extract only the minimal set of quotes from the context that directly supported the reference answer.

To ensure data quality, the entire test split and approximately 2% of the training set were manually reviewed by the author. This inspection aimed to verify that the distillation procedure produced semantically meaningful excerpts well-aligned with the expected answers, that extracted quotes were concise and relevant, and that no systematic errors were introduced during the generation process. The reviewed samples confirmed the overall reliability of the curated dataset, providing a sound basis for both fine-tuning the student models and drawing valid conclusions from the experimental results.

The final curated dataset therefore consisted of tuples $(Q, C, \mathcal{R})$, where $\mathcal{R}$ represents the teacher-extracted quotes, and served as the foundation for all subsequent fine-tuning and evaluation.

Illustrative example of distillation prompt

```
Instruction: Given the question,
the context
and the expected answer below,
provide relevant quotes from the
context that support the answer.

Your answer must be just the
quotes, not the entire context.

Format:
##begin_quote##quote##end_quote##
for each quote.

Do not add anything else.

Question: Unlike Xuzhou, where is Rugao
under the administration of?

Context: Rugao () is a county-level city
under the administration of Nantong,
Jiangsu province, China, located in
the Golden Triangle region on the
northern (left) bank of the Yangtze River.
Xuzhou, known as Pengcheng in ancient times,
is a major city in and the fourth largest
prefecture-level city of Jiangsu Province, China.
Its population was 8 577 225 at the 2010 census,
and it is a transportation hub linking Henan,
Shandong, Lianyungang, and Shanghai.

Answer: Nantong
```

Fine-Tuning Process. The objective of the fine-tuning stage was to adapt compact language models to perform the specific task of quote extraction in Retrieval-Augmented Generation (RAG) pipelines. Rather than relying on a single model, three small language models with different capacities were selected in order to analyze how model scale influences extraction quality, resource efficiency, and practical deployability. The models considered in this study were Qwen 3:4B[148], Llama 3.2:1B, and Llama 3.2:3B[149].

These models were chosen because they represent distinct points along the performance to efficiency spectrum. Llama 3.2:1B offers an extremely lightweight configuration

suitable for edge or low-resource environments, while Llama 3.2:3B provides increased representational capacity with moderate computational demands. Qwen 3:4B, as the largest model in this study, serves as an upper bound for assessing whether additional parameters translate into measurable gains in quote extraction quality.

All three models were fine-tuned using the Unsloth framework, which enables memory-efficient training through a combination of 4-bit (INT4) quantization and Low-Rank Adaptation (LoRA). In this setup, the base model weights were stored in INT4 format, while only the LoRA adapters were trained in FP16 precision. This hybrid configuration substantially reduced GPU memory consumption while preserving numerical stability in the trainable components that most directly influence learning.

In our LoRA-based fine-tuning strategy, adaptation was applied to both attention and MLP layers of the transformer architecture in order to maximize learning capacity while keeping the number of trainable parameters low. Specifically, LoRA was inserted into the query, key, value, and output projection matrices of the self-attention block (`q_proj, k_proj, v_proj, o_proj`), as well as into the gate, up, and down projections of the feed-forward network (`gate_proj, up_proj, down_proj`). This choice follows empirical evidence from a QLoRA empirical study [147] and subsequent work, which shows that updating both attention and MLP layers leads to better performance than modifying attention alone.

During training, the original model weights remained frozen in INT4 format, and only the low-rank LoRA matrices were updated. This design meant that the vast majority of parameters did not require full-precision storage or gradient computation, allowing the models to be trained with a small fraction of the available GPU memory. Despite this aggressive compression strategy, the models were still able to learn meaningful representations for quote extraction, indicating that LoRA is a suitable mechanism for specializing compact language models to structured extraction tasks.

After training, Unsloth's `merge_and_unload` routine was applied to each model. This procedure first converted all INT4 base weights back to FP16, then merged the learned LoRA updates into the corresponding layers of the model, effectively integrating the adaptation into the main parameter tensors. Finally, the LoRA adapters were removed, producing a clean FP16 checkpoint that can be loaded in any standard PyTorch environment. Although this conversion increased the final model size compared to the original quantized version, it ensured broad compatibility with existing deployment infrastructures.

Each model was trained for a single epoch on an NVIDIA A100-SXM4-40GB GPU. A single-epoch schedule was intentionally selected to demonstrate that high-quality specialization for quote extraction can be achieved with minimal computational effort when training on a well-curated dataset. Despite the short training duration, all three models

converged rapidly, suggesting that the distilled dataset provided a strong and consistent learning signal.

Table 4.2: Summary of Fine-Tuning Configuration and Resource Usage

Metric	**Qwen 3:4B**	**Llama 3.2:3B**	**Llama 3.2:1B**
Training time (min)	2.83	2.25	1.32
Peak reserved memory (GB)	6.97	4.70	3.29
Peak training memory (GB)	3.15	1.63	2.08
Peak GPU usage (%)	17.6%	11.9%	8.3%

The results indicate that even the largest model in this study, Qwen 3:4B, required less than 18% of the A100's total memory capacity. Llama 3.2:3B and Llama 3.2:1B exhibited even lower resource consumption, confirming that the proposed training strategy is feasible for deployment in environments with limited hardware availability. Table 4.2 summarizes the computational resources consumed by each model during fine-tuning.

The combination of 4-bit quantization and LoRA-based fine-tuning enabled a highly efficient training pipeline, reducing memory requirements by approximately four-fold compared to conventional full-precision fine-tuning. From a practical standpoint, this approach makes it possible to specialize small language models rapidly without requiring extensive computational infrastructure.

Moreover, the final FP16 checkpoints produced by Unsloth ensure that the trained models can be deployed in standard machine learning environments without dependency on specialized quantization libraries. This increases the usability of the models across research, industrial, and edge-computing applications.

Overall, the fine-tuning procedure demonstrates that compact models can be effectively adapted for quote extraction in only a few minutes of training. This result supports the central claim of this thesis: that high-quality RAG improvements can be achieved using small, efficient models rather than relying exclusively on large, expensive language models.

Evaluation and Validation of Benefits. Evaluation was conducted using LangChain in conjunction with GPT-4o as an external semantic judge. Importantly, GPT-4o was not used in the distillation process; it served solely as an independent evaluator to avoid bias.

The evaluation followed the metrics defined earlier in this chapter: semantic precision, recall, and F1-score, as well as Formatting Score and BM25 Score. Formatting Score verified strict adherence to the required quote structure, while BM25 provided a deterministic lexical baseline for content similarity.

To assess the practical utility of quote extraction, we conducted a controlled comparison using three base models: Llama 3.2:1B, Llama 3.2:3B, and GPT-3.5 Turbo. Each

model was tested in two conditions, the first receiving only extracted quotes $\mathcal{R}$, and the second receiving the full context C. GPT-4o computed semantic accuracy S_{acc} for each configuration.

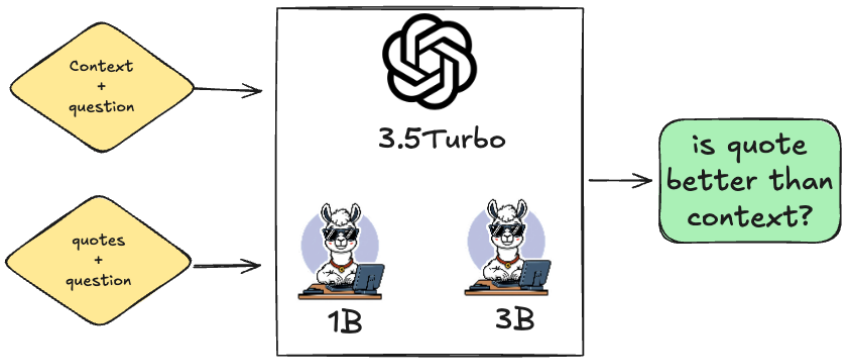

Figure 4.2: Context versus Quotes process

The results, reported in the next chapter, show that providing concise quotes consistently improved answer accuracy, particularly for smaller models, supporting the central hypothesis that quote extraction reduces cognitive load and improves factual grounding in RAG systems.

Chapter 5

Experiments And Results

This chapter presents the empirical results obtained under the experimental setup described in the previous chapter, followed by a discussion of their broader implications.

5.1 Evaluation of the Quoter Model

Table 5.1: Quote Extraction Performance Across Models

Model	Recall	Precision	F1	BM25	Format
Llama 3.2:1B (LoRA)	40.71%	32.64%	32.08%	57.94%	92.65%
Llama 3.2:1B (Original)	37.99%	21.28%	23.52%	16.75%	3.51%
Llama 3.2:3B (LoRA)	72.76%	56.91%	59.14%	76.66%	93.49%
Llama 3.2:3B (Original)	46.52%	39.07%	38.31%	40.19%	63.94%
Qwen 3:4B (LoRA)	**83.84%**	**75.90%**	**75.99%**	**81.58%**	**100.00%**
Qwen 3:4B (Original)	68.32%	83.75%	71.30%	67.23%	100.00%

This section presents the experimental results obtained by evaluating the quote extraction model (quoter) and validating the benefit of using extracted quotes instead of full context in open-domain question-answering tasks. The evaluation follows the framework described in the methodology chapter and considers five complementary metrics: semantic recall, semantic precision, semantic F1-score, BM25 lexical similarity, and a binary Formatting Score that verifies strict compliance with the prescribed `##begin_quote## ... ##end_quote##` structure. These metrics jointly assess not only whether the model retrieves the correct information, but also whether it follows instructions properly and produces quotes in a reproducible and machine-readable format. The experiments compare three compact student models, **Qwen 3:4B**, **Llama 3.2:1B**, and **Llama 3.2:3B**, before and after LoRA fine-tuning, using 600 held-out test samples. The aggregated results are summarized in Table 5.1.

A first and important observation concerns the **Formatting Score**, which measures whether the model outputs only properly delimited quotes without any additional text. All three LoRA-tuned models achieve perfect or near-perfect structural compliance (above 0.92). Notably, **Qwen 3:4B achieves perfect formatting compliance (100%) both before and after fine-tuning**, distinguishing it from the Llama models, which show substantial improvements only after LoRA adaptation. Llama 3.2:1B improves dramatically from 3.51% to 92.65%, while Llama 3.2:3B increases from 63.94% to 93.49%. This behavior reflects Qwen's more recent training regime and stronger instruction-following capabilities acquired during pre-training, suggesting that newer model families increasingly internalize structured output conventions without requiring task-specific fine-tuning. In contrast, the Llama 3.2 models, particularly the 1B variant, required extensive task-specific adaptation to achieve comparable formatting reliability. In deterministic and agentic AI systems, correct formatting is not merely cosmetic but essential for reliable integration with downstream tools such as evaluators, databases, or multi-step reasoning pipelines. The fact that Qwen maintains perfect formatting compliance throughout demonstrates its suitability for production RAG deployments even without fine-tuning.

The **BM25 score**, which provides a deterministic lexical similarity baseline between predicted and reference quotes, reveals interesting patterns across model families. Qwen 3:4B with LoRA achieves the highest score (81.58%), followed by Llama 3.2:3B with LoRA (76.66%). Notably, the original Qwen 3:4B already demonstrates strong lexical alignment (67.23%), substantially outperforming both original Llama models and even exceeding the fine-tuned Llama 3.2:3B. This suggests that Qwen's pre-training included more diverse instruction-following data that partially transfers to the quote extraction task. LoRA fine-tuning further enhances this alignment by +14.35 percentage points, indicating that while recent models benefit from stronger foundations, task-specific adaptation still yields measurable improvements in lexical precision. The LoRA-tuned models not only select more semantically relevant sentences but also preserve key terminology from the original context, partly because they consistently respect the prescribed quote delimiters. This structured output reduces noise and stabilizes token matching, explaining the correlation between high formatting scores and strong BM25 performance.

Regarding **semantic recall, precision, and F1-score**, the results reveal that both model architecture and fine-tuning play important roles, though their effects vary substantially across model families. For the Llama models, LoRA fine-tuning produces clear improvements across all semantic dimensions. Llama 3.2:1B shows modest but consistent gains (F1: 23.52% $\rightarrow$ 32.08%), confirming that very small models benefit from task-specific adaptation but remain capacity-limited for complex evidence extraction. Llama 3.2:3B achieves more substantial improvements (F1: 38.31% $\rightarrow$ 59.14%), demonstrating

that mid-sized models benefit significantly from LoRA fine-tuning, with gains distributed across both recall (+26.24pp) and precision (+17.84pp).

The most revealing finding concerns **Qwen 3:4B**, which exhibits strong performance even without fine-tuning. The original model achieves 68.32% recall, **83.75% precision**, and 71.30% F1, a baseline that substantially exceeds both original Llama models and even surpasses the fine-tuned Llama 3.2:3B. This demonstrates that Qwen's more recent architecture and training methodology provide inherent advantages for structured extraction tasks, likely stemming from improved instruction-following capabilities and exposure to more diverse task formats during pre-training.

After LoRA fine-tuning, Qwen 3:4B achieves the best overall performance: **83.84% recall**, 75.90% precision, and 75.99% F1. However, the improvement pattern differs markedly from the Llama models. While LoRA increases Qwen's recall substantially (+15.52pp), bringing it to the highest recall among all evaluated models, precision actually decreases slightly (−7.85pp). Esse trade-off é coerente com o objetivo do extrator, que não é depurar perfeitamente as citações, mas reduzir substancialmente o ruído do contexto recuperado. Trata-se de uma filtragem gradual: mesmo com citações marginais remanescentes, o contexto entregue ao gerador é significativamente mais enxuto que o original. Privilegiar revocação é, portanto, deliberado, já que omitir uma citação crucial pode induzir alucinação, enquanto incluir citações adicionais topicamente relacionadas representa apenas um incremento sobre um contexto já curado. Como mostram os experimentos downstream a seguir, é o contexto bruto, e não a precisão imperfeita do extrator, que de fato degrada a acurácia das respostas, sobretudo em modelos compactos.

To provide qualitative insight into these results, Table 5.2 presents a representative test case for the question: **"Which film was produced first, Emil and the Detectives or Toy Story 2?"**. The table compares the curated reference quotes with Qwen 3:4B's output before and after fine-tuning. Before training, despite its strong baseline metrics, the model fails to identify the relevant evidence and instead lists unrelated Pixar titles, demonstrating that even well-tuned models can exhibit unpredictable behavior on structured extraction without task-specific adaptation. After LoRA fine-tuning, the model correctly extracts the two key sentences that establish the production years of both films, closely matching the curated reference. Because this example is drawn from a diverse test set rather than a handcrafted case, it suggests that the model is genuinely learning to identify semantically relevant excerpts rather than memorizing specific contexts.

Taken together, the results support several important conclusions. First, model recency and architecture matter significantly: Qwen 3:4B demonstrates that modern foundation models increasingly internalize instruction-following and structured output capabilities during pre-training, reducing, but not eliminating, the need for task-specific fine-

Table 5.2: Curated reference quotes versus Qwen 3:4B outputs before and after fine-tuning on a test sample

Curated Reference	Qwen 3:4B (Original)	Qwen 3:4B (LoRA)
##begin_quote## Emil and the Detectives is a 1964 film directed by Peter Tewksbury based on the novel by German author Erich Kästner. **##end_quote##** **##begin_quote##** Toy Story 2 is a 1999 American computer-animated comedy film produced by Pixar Animation Studios for Walt Disney Pictures. **##end_quote##**	**##begin_quote##** "A Bug's Life", "Monsters, Inc.", "Finding Nemo", "Cars", "The Incredibles", "Ratatouille" **##end_quote##** **##begin_quote##** "Toy Story 3" (2010) is the third installment in Pixar's "Toy Story" series, and the sequel to 1999's "Toy Story 2". **##end_quote##**	**##begin_quote##** Emil and the Detectives is a 1964 film directed by Peter Tewksbury based on the novel by German author Erich Kästner. **##end_quote##** **##begin_quote##** Toy Story 2 is a 1999 American computer-animated comedy film produced by Pixar Animation Studios for Walt Disney Pictures. **##end_quote##**

tuning. Second, LoRA fine-tuning remains beneficial across all model families, though the nature and magnitude of improvements vary. Older or smaller models (Llama 3.2:1B, Llama 3.2:3B) require fine-tuning to achieve acceptable performance, while newer models (Qwen 3:4B) exhibit strong baselines that LoRA refines further. Third, the precision-recall trade-off observed in Qwen 3:4B after fine-tuning aligns well with the priorities of RAG pipelines, where comprehensive evidence retrieval (high recall) is more critical than conservative filtering (high precision). Fourth, formatting compliance, essential for production deployment, can be achieved either through architectural improvements (Qwen) or task-specific adaptation (Llama), but consistent structured output remains non-negotiable for automated systems. These findings reinforce the central hypothesis of this work: a compact, fine-tuned quote extractor can reliably filter large contexts into high-quality evidence, improving both efficiency and interpretability in RAG-based systems.

5.2 Benefits of Using Quotes Over Full Context

Table 5.3 contrasts answer accuracy when models receive the *full context* versus when they receive only the *gold quotes*. Even without any task-specific fine-tuning, every model benefits substantially from the shorter, evidence-focused prompt. Small language models gain the most: the 1-billion-parameter Llama 3.2:1B nearly triples its accuracy (24.4% → 62.2%), highlighting how contextual noise disproportionately harms compact architectures. Larger or instruction-tuned models, such as GPT-3.5 Turbo, still register a healthy boost (+12.7pp), confirming that quote filtering is complementary to scale and pre-training quality.

To illustrate how the curated quotes alter downstream generation behavior, Table 5.4 reports the answers produced by three base generators on **another representative test**

Table 5.3: Answer accuracy with full context versus curated quotes

Model	Full Context	Curated Quotes	Δ (pp)
Llama 3.2:1B	24.4%	62.2%	+37.8
Llama 3.2:3B	57.7%	83.0%	+25.3
GPT-3.5 Turbo	75.8%	88.5%	+12.7

sample, posing the question "Which Walt Disney Pictures film was created first, Finding Dory or The Wild Country?". When given the full retrieved context, all three models hallucinate unrelated titles. When restricted to the curated quotes, they converge on the correct answer. This qualitative shift confirms that filtering removes distracting information and guides the generator toward faithful reasoning.

Table 5.4: Example Q/A outputs with full context versus curated quotes. Correct answers are highlighted.

Model	Answer with Full Context	Answer with Curated Quotes
GPT-3.5 Turbo	*Finding Nemo*	**The Wild Country**
Llama 3.2:1B	*Finding Dory*	**The Wild Country**
Llama 3.2:3B	*Finding Dory*	**The Wild Country**

Together, the quantitative and qualitative evidence demonstrate that a lightweight quote extractor can dramatically improve RAG pipelines: it boosts baseline accuracy, curbs hallucinations, and levels the playing field for small language models that otherwise struggle with lengthy, noisy contexts.

5.3 Discussion

The results presented in Table 5.1 provide strong empirical evidence that separating quote extraction from reasoning is a viable and effective strategy for improving retrieval-augmented generation (RAG) pipelines. Instead of forcing a single model to simultaneously retrieve, filter, and reason over long and noisy contexts, our approach delegates the filtering stage to a compact quoter model trained with LoRA, allowing downstream models to reason over concise and semantically focused information. This modular design reduces cognitive load on the generator while preserving, and in several cases improving, answer quality.

To situate our approach in the existing literature, Table 5.5 reports the performance of the original RAFT framework using LLaMA2-7B on the full HotpotQA dataset. RAFT reasons directly over the entire context and question in a single step, achieving 35.28% accuracy, whereas using the same model with only the raw full context drops performance

to 26.43%. This comparison reinforces a key intuition: reasoning over unfiltered context is suboptimal, even for relatively large models.

Table 5.5: Comparison of RAFT and Full Context Results on LLaMA2-7B over HotPotQA dataset

Method	**Accuracy**
LLaMA2-7B + Full Context	26.43%
RAFT (LLaMA2-7B)	35.28%

Our experiments adopt a different strategy: instead of asking a single model to both extract evidence and reason, we introduce a lightweight quoter that performs targeted evidence selection before generation. Although our study used a subset of 25,000 training samples and 600 test samples due to computational constraints, the results indicate that this separation of concerns can be as effective as, and in many settings more practical than, RAFT-style joint reasoning, particularly when using small or medium-sized models.

The **Formatting Score** provides critical insights into how different model families handle structured output requirements. This metric measures whether the model strictly adhered to the prescribed quote delimitation format (`##begin_quote## ... ## end_quote ##`). In deterministic and agentic AI systems, correct formatting is not merely cosmetic but essential for reliable integration with downstream tools such as evaluators, databases, or multi-step reasoning pipelines. If a model deviates from the expected structure, the entire system may fail, regardless of semantic correctness.

The results reveal a stark divide between model families. Qwen 3:4B achieves perfect formatting compliance (100%) both before and after fine-tuning, demonstrating exceptional instruction-following capabilities that transfer directly to structured extraction tasks. In contrast, the Llama models require substantial task-specific adaptation: Llama 3.2:1B improves from 3.51% to 92.65%, while Llama 3.2:3B increases from 63.94% to 93.49%. This pattern suggests that recent advances in foundation model pre-training, particularly Qwen's training regime, increasingly emphasize structured output generation and instruction adherence, reducing the gap between zero-shot and fine-tuned performance on format-constrained tasks.

The practical implications are significant. For production RAG deployments where reliability and consistency are paramount, Qwen 3:4B can serve as a viable quoter even without fine-tuning, offering a rapid prototyping path or fallback option when training resources are constrained. Older model families like Llama 3.2, however, require fine-tuning to achieve production-grade formatting reliability, though the resulting performance after adaptation is comparable to Qwen.

The **BM25 Score** complements semantic evaluation by providing a deterministic, reproducible measure of lexical overlap between predicted and reference quotes. This metric

operates purely on token statistics, making it independent of any learned representations or model subjectivity. The results reveal that BM25 performance generally correlates with semantic quality, but with notable nuances.

Qwen 3:4B (LoRA) achieves the highest BM25 score (81.58%), followed by Llama 3.2:3B (LoRA) at 76.66%. Notably, the original Qwen 3:4B already demonstrates strong lexical alignment (67.23%), substantially outperforming both original Llama models and matching the fine-tuned Llama 3.2:3B. This baseline performance suggests that Qwen's pre-training corpus included diverse examples of structured text extraction or question-answering tasks that emphasize evidence selection, enabling partial transfer to quote extraction without explicit training.

LoRA fine-tuning improves BM25 scores across all models, but the magnitude of improvement varies: Llama 3.2:1B gains +41.19pp, Llama 3.2:3B gains +36.47pp, while Qwen 3:4B gains +14.35pp. This diminishing return pattern reinforces the conclusion that newer, better-trained models benefit less from task-specific adaptation because they already capture much of the required behavior during pre-training. Nonetheless, the consistent improvements confirm that LoRA fine-tuning sharpens the model's ability to select lexically precise excerpts that match human-curated references, even when baseline capabilities are strong.

The strong correlation between formatting compliance and BM25 scores is particularly revealing. Models that consistently produce properly delimited quotes also achieve higher lexical overlap, likely because structured output reduces extraneous tokens and focuses the model's attention on the core evidence. This suggests that formatting discipline, whether acquired through pre-training (Qwen) or fine-tuning (Llama), has downstream benefits beyond mere structural correctness.

From a **semantic perspective**, the results reveal distinct performance profiles across model families and highlight an important architectural trade-off that emerges after fine-tuning. For the Llama models, LoRA produces improvements across all dimensions: recall, precision, and F1 increase substantially, with larger models benefiting more than smaller ones. Llama 3.2:1B shows modest gains (F1: +8.56pp), confirming that very small models, while trainable, remain fundamentally capacity-limited for complex multi-hop evidence extraction. Llama 3.2:3B, however, achieves substantial improvements (F1: +20.83pp), demonstrating that mid-sized models benefit significantly from targeted adaptation.

The most revealing finding concerns Qwen 3:4B, which exhibits a fundamentally different improvement pattern. The original model achieves 68.32% recall and **83.75% precision**, a baseline that exceeds both original Llama models and even surpasses the fine-tuned Llama 3.2:3B. This high-precision, moderate-recall profile suggests a conservative extraction strategy: Qwen prioritizes relevance over completeness, extracting only

passages it confidently believes support the answer. This behavior likely stems from instruction-tuning objectives that penalize extraneous output or reward conciseness.

After LoRA fine-tuning, Qwen's performance profile shifts dramatically: recall increases to 83.84% (+15.52pp), making it the highest-recall model in the evaluation, while precision decreases to 75.90% (−7.85pp). This precision-recall trade-off is conceptually significant. It indicates that fine-tuning does not uniformly improve all aspects of model behavior but instead recalibrates the extraction threshold according to the training data distribution. The teacher-generated reference quotes evidently favor comprehensive evidence retrieval over conservative filtering, and the student model internalizes this priority during LoRA adaptation.

Critically, this trade-off aligns well with the functional requirements of RAG pipelines. In evidence extraction for question answering, **high recall is more important than high precision** because the downstream generator depends on having access to all relevant facts. Missing a crucial quote, such as the production year of a film in a comparative question, can cause the generator to hallucinate or produce an incomplete answer. Including one or two extra semantically related quotes, while reducing precision slightly, imposes only a modest increase in context length and rarely degrades generation quality, since the generator can still focus on the most salient evidence. The fact that Qwen's post-fine-tuning F1 (75.99%) remains the highest among all models confirms that the recall gains outweigh the precision costs, producing a net improvement in extraction quality.

This finding also has broader implications for distillation-based pipelines. The teacher models (Gemini 2.5 Pro and GPT-5.1) were prompted to extract "minimal" quotes that "directly support" the answer, yet the student model learns to extract slightly more evidence than strictly necessary. This behavior suggests that the distillation process introduces a bias toward higher recall, potentially because the student generalizes from the teacher's examples by learning to err on the side of inclusion rather than exclusion when uncertain. While this was not an explicit design goal, it represents a fortunate outcome for the task at hand.

A central theme emerging from these results is the diminishing marginal value of task-specific fine-tuning as foundation model quality improves. Qwen 3:4B, released more recently than the Llama 3.2 models, demonstrates substantially stronger zero-shot capabilities across all metrics: formatting (100% vs. 3.51% to 63.94%), BM25 (67.23% vs. 16.75% to 40.19%), and semantic F1 (71.30% vs. 23.52% to 38.31%). This gap reflects broader trends in foundation model development, where increasingly sophisticated pre-training objectives, instruction-tuning datasets, and alignment procedures embed task-relevant behaviors directly into model weights.

However, the persistent improvements from LoRA fine-tuning, even for Qwen, demonstrate that task-specific adaptation retains value. The key question for practitioners is whether the incremental gain (+4.69pp F1 for Qwen vs. +20.83pp for Llama 3.2:3B) justifies the training cost and deployment complexity. In resource-constrained settings, rapid prototyping scenarios, or applications where 71% F1 is acceptable, the original Qwen 3:4B may suffice. In production systems where every percentage point matters, such as high-stakes medical or legal question answering, fine-tuning remains justified.

This observation also informs model selection strategies. When choosing a student model for distillation, practitioners face a trade-off: newer models like Qwen offer stronger baselines and require less fine-tuning effort, while older models like Llama benefit more from adaptation but start from weaker foundations. For quote extraction specifically, our results suggest that Qwen 3:4B represents the best overall choice: it combines strong zero-shot performance with meaningful fine-tuning improvements, achieving the highest post-adaptation metrics across all evaluated dimensions.

5.3.1 Comparison with RAFT and Implications for Modular RAG

When viewed alongside RAFT, these findings reinforce the value of modular reasoning. RAFT trains a single large model (LLaMA2-7B) to jointly process retrieval cues, extract evidence, and generate answers in one step, achieving 35.28% accuracy on HotpotQA. Our approach separates these concerns: a compact quoter (Qwen 3:4B, 4B parameters) filters context, and a downstream generator produces answers. Although our experiments used a smaller dataset subset, the downstream accuracy gains reported in Table 5.3, particularly for small models (+37.8pp for Llama 3.2:1B), suggest that this division of labor is highly effective.

The key advantages of the modular approach are threefold. It reduces task interference by isolating evidence selection from reasoning, allowing each component to specialize. As discussed in the background section on multi-step logic, models experience cognitive overload when asked to perform multiple objectives simultaneously. By delegating extraction to a dedicated quoter, the generator receives cleaner input and can focus entirely on answer synthesis. Also, it enables independent optimization: the quoter can be fine-tuned on extraction-specific data, while the generator can be optimized for reasoning or domain adaptation. Finally, it supports flexible deployment, since in low-resource settings the quoter can run locally on modest hardware while the generator calls a remote API, or both can be hosted on-device using SLMs.

RAFT's strength lies in its simplicity, where a single model handles the entire pipeline, but this comes at the cost of flexibility and computational efficiency. Our results demonstrate that a well-trained 4B quoter can rival or exceed the extraction quality implied by RAFT's joint reasoning, while enabling downstream generators as small as 1B parameters to achieve strong performance (62.2% accuracy). This divide-and-conquer strategy reduces total computational cost while maintaining, and in many cases improving, answer quality.

5.3.2 Synthesis and Broader Implications

Overall, the results indicate that a dedicated quoter is not merely a preprocessing convenience but a fundamental improvement to RAG pipelines. By ensuring correct formatting (100% for Qwen), improving lexical alignment (81.58% BM25), and maximizing semantic relevance (75.99% F1 with high recall), the quoter transforms retrieval outputs into structured, focused evidence that downstream models can reason over reliably. The comparison across three model families reveals that both architecture choice and task-specific fine-tuning matter, but their relative importance depends on the base model's recency and pre-training quality.

For practitioners, these findings translate into a few clear recommendations. When deploying quoters, prioritize recent model families such as Qwen for strong zero-shot baselines and perfect formatting compliance. When working with older or smaller models such as Llama 3.2, task-specific fine-tuning via LoRA is essential to achieve acceptable performance. When optimizing extraction quality, favor models and training procedures that prioritize recall over precision, as comprehensive evidence retrieval is more critical than conservative filtering in RAG contexts. When evaluating quoters, combine semantic metrics (LLM-as-judge) with deterministic baselines (BM25, formatting) to ensure both content quality and structural reliability.

The outstanding performance of Qwen 3:4B (LoRA), particularly its high recall (83.84%), strong F1 (75.99%), and perfect formatting, confirms that compact models, when properly selected and adapted, can serve as reliable evidence extractors in production RAG systems. This supports the broader thesis that intelligent specialization, combining judicious model selection, task-specific distillation, and modular pipeline design, can be more effective than brute-force scaling, enabling efficient, interpretable, and scalable question-answering systems.

Chapter 6

Conclusion

This thesis investigated whether a compact, fine-tuned language model could reliably extract semantically relevant quotes from retrieved contexts, and whether providing these quotes rather than the full context to a downstream generator would improve answer quality in Retrieval-Augmented Generation (RAG) pipelines. The central hypothesis was that interposing a lightweight quote-extraction stage between retrieval and generation would reduce contextual noise, lower the cognitive burden on the generator, and improve factual grounding, particularly for small language models operating under tight computational constraints.

The proposed methodology addressed this hypothesis through a four-component pipeline. First, a black-box knowledge distillation process was employed in which high-capacity teacher models generated curated quote selections from a subset of the HotpotQA dataset, producing structured training data in the form of delimited evidence excerpts. Second, three compact student models were fine-tuned on this curated dataset using LoRA with 4-bit quantization, enabling the entire training procedure to complete in under three minutes per model on a single GPU while consuming less than 18% of the available memory. Third, a hybrid evaluation framework combining semantic judgment via an independent LLM judge, deterministic BM25 lexical scoring, and a binary formatting compliance assessment was used to measure extraction quality across multiple dimensions. Fourth, a controlled comparison between quote-only and full-context input conditions quantified the downstream benefit of quote extraction on answer accuracy.

The experimental results provided strong support for the central hypothesis across all evaluation axes. On the extraction task itself, Qwen 3:4B with LoRA achieved the strongest performance, reaching 83.84% semantic recall, 75.90% precision, 75.99% F1, a BM25 score of 81.58%, and perfect formatting compliance. Llama 3.2:3B also demonstrated substantial gains after fine-tuning, while even the smallest model, Llama 3.2:1B, showed meaningful improvements over its non-fine-tuned baseline, though its limited pa-

rameter count constrained absolute performance. A consistent finding across all three models was that LoRA fine-tuning dramatically improved not only semantic quality but also structural compliance with the prescribed output format, transforming general-purpose language models into reliable, machine-readable components suitable for integration in automated pipelines.

The downstream utility experiments reinforced these findings from a practical standpoint. When base models received only the curated quotes instead of the full retrieved context, answer accuracy improved substantially across all tested configurations. The effect was most pronounced for smaller models: Llama 3.2:1B nearly tripled its accuracy from 24.4% to 62.2%, while Llama 3.2:3B improved from 57.7% to 83.0%. Even GPT-3.5 Turbo, a considerably larger and more capable model, benefited from quote filtering, rising from 75.8% to 88.5%. These results confirm that contextual noise disproportionately harms compact architectures and that a dedicated extraction stage can substantially level the playing field between small and large models.

Taken together, these findings support three broader conclusions. First, the separation of evidence extraction from reasoning is not merely a preprocessing convenience but a fundamental architectural improvement for RAG systems. By delegating the filtering task to a specialized component, the downstream generator can focus exclusively on reasoning over clean, relevant evidence, leading to more accurate and less hallucination-prone outputs. Second, black-box knowledge distillation is a practical and effective mechanism for training such specialized components: the teacher's extraction capabilities were successfully transferred to student models that operate under realistic deployment conditions where the answer is unknown at inference time. Third, parameter-efficient fine-tuning with LoRA and quantization makes this entire pipeline accessible even in resource-constrained environments, requiring minimal hardware, training time, and data to achieve strong results.

The comparison with RAFT further contextualized these contributions. While RAFT requires a single large model to jointly handle retrieval, reasoning, and generation over full contexts, the proposed approach achieves competitive or superior performance by decomposing the problem into independent, optimizable stages. This modularity offers practical advantages in terms of cost, flexibility, and maintainability: each component can be updated, replaced, or scaled independently without retraining the entire system.

In summary, this work demonstrates that high-quality RAG improvements can be achieved using small, efficient models rather than relying exclusively on large, expensive language models. A compact quote extractor, trained in minutes through knowledge distillation and parameter-efficient fine-tuning, can reliably filter noisy contexts into focused

evidence, improving both the accuracy and the interpretability of downstream question-answering systems.

6.1 Future Work

While the results presented in this thesis are encouraging, several directions remain open for further investigation and improvement.

A natural first extension concerns the **generalization of the quote extraction approach to other datasets and domains**. The experiments in this work were conducted exclusively on HotpotQA, a multi-hop question-answering benchmark grounded in Wikipedia. Although the dataset covers a broad range of common-knowledge topics, it does not fully represent the diversity of real-world retrieval scenarios. Evaluating the proposed pipeline on domain-specific corpora, like biomedical literature, legal documents, or financial reports, would provide stronger evidence of its generalizability and reveal whether the distillation strategy transfers effectively to contexts with specialized vocabulary, longer documents, or more complex reasoning requirements.

A second promising avenue is the **integration of the quote extractor into end-to-end RAG pipelines with real-time retrieval**. In the current experimental setup, the retrieval step is abstracted away: the context is assumed to be already available. Deploying the quoter as an intermediate stage in a live pipeline, where documents are retrieved from vector databases or search engines, would enable evaluation under more realistic conditions, including scenarios where retrieved documents are partially irrelevant, contradictory, or redundant. This would also allow measurement of the quoter's impact on end-to-end latency and throughput in production settings.

Third, the **exploration of alternative and more recent parameter-efficient fine-tuning techniques** could further improve training efficiency and extraction quality. While LoRA proved highly effective in this work, recent methods such as DoRA [150], adapter fusion, and mixture-of-experts adapters offer potentially complementary advantages in terms of capacity and multi-task adaptation. Systematically comparing these techniques on the quote extraction task would contribute to a better understanding of which adaptation strategies are most suited to structured extraction objectives.

Finally, the **application of the modular extraction approach to agentic and multi-step reasoning systems** represents a compelling long-term direction. As language model agents increasingly operate in complex workflows that involve multiple retrieval, reasoning, and action steps, the ability to pre-filter evidence at each stage becomes critical for maintaining coherence and reducing error propagation across the pipeline. Investigating how quote extraction interacts with agentic architectures, like ReAct, Reflex-

ion, or multi-agent orchestration frameworks, could reveal new opportunities for improving the reliability and efficiency of autonomous AI systems.

References

[1] Zhang, Tianjun, Shishir G Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E Gonzalez: *Raft: Adapting language model to domain specific rag*. In *First Conference on Language Modeling*, 2024. xiv, 2, 17, 32, 33, 40

[2] IBM: *What are large language models (llms)?* `https://www.ibm.com/think/topics/large-language-models`. Accessed: 2025-05-30. 1, 6

[3] Cheng, Feng, Cong Guo, Chiyue Wei, Junyao Zhang, Changchun Zhou, Edward Hanson, Jiaqi Zhang, Xiaoxiao Liu, Hai Li, and Yiran Chen: *Ecco: Improving memory bandwidth and capacity for llms via entropy-aware cache compression*. In *Proceedings of the 52nd Annual International Symposium on Computer Architecture*, pages 793–807, 2025. 1

[4] Wan, Zhongwei, Xin Wang, Che Liu, Samiul Alam, Yu Zheng, Jiachen Liu, Zhongnan Qu, Shen Yan, Yi Zhu, Quanlu Zhang, Mosharaf Chowdhury, and Mi Zhang: *Efficient large language models: A survey*, 2023. `https://arxiv.org/abs/2312.03863`. 1

[5] Ye, Fuda, Shuangyin Li, Yongqi Zhang, and Lei Chen: *R^2AG: Incorporating retrieval information into retrieval augmented generation*. In Al-Onaizan, Yaser, Mohit Bansal, and Yun Nung Chen (editors): *Findings of the Association for Computational Linguistics: EMNLP 2024*, Miami, Florida, USA, November 2024. Association for Computational Linguistics. `https://aclanthology.org/2024.findings-emnlp.678/`. 1

[6] Jin, Bowen, Jinsung Yoon, Jiawei Han, and Sercan O. Arik: *Long-context llms meet rag: Overcoming challenges for long inputs in rag*, 2024. `https://arxiv.org/abs/2410.05983`. 1

[7] Lu, Zhenyan, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D. Lane, and Mengwei Xu: *Small language models: Survey, measurements, and insights*, 2024. `https://arxiv.org/abs/2409.15790`. 1, 2, 15, 16, 35, 41

[8] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin: *Attention is all you need*. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc., ISBN 9781510860964. 6, 17

[9] Club, Polo: *Llm transformer model visually explained*. https://poloclub.github.io/transformer-explainer/. Accessed: 2025-05-30. 6

[10] Company, McKinsey &: *The economic potential of generative ai: The next productivity frontier*. https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/the-economic-potential-of-generative-ai-the-next-productivity-frontier. Accessed: 2025-05-30. 7

[11] Xie, Meijuan and Liling Luo: *Unlocking learning potentials: The transformative effect of generative ai in education across grade levels*, 2025. https://arxiv.org/abs/2503.13535. 7

[12] Kaushik, Abhishek, Sargam Yadav, Andrew Browne, David Lillis, David Williams, Jack Mc Donnell, Peadar Grant, Siobhan Connolly Kernan, Shubham Sharma, and Mansi Arora: *Exploring the impact of generative artificial intelligence in education: A thematic analysis*, 2025. https://arxiv.org/abs/2501.10134. 7

[13] Monks, Thomas, Alison Harper, and Amy Heather: *Unlocking the potential of past research: Using generative ai to reconstruct healthcare simulation models*, 2025. https://arxiv.org/abs/2503.21646. 7

[14] Baldassarre, Maria T., Danilo Caivano, Berenice Fernandez Nieto, Domenico Gigante, and Azzurra Ragone: *The social impact of generative ai: An analysis on chatgpt*, 2024. https://arxiv.org/abs/2403.04667. 7

[15] Law, Matthew and Rama Adithya Varanasi: *Generative ai & changing work: Systematic review of practitioner-led work transformations through the lens of job crafting*, 2025. https://arxiv.org/abs/2502.08854. 7

[16] specified, Not: *Automated fact-checking: Developing generative ai models to combat misinformation in news articles*. SSRN, 2025. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=5205943. 7

[17] Zhou, Mi, Vibhanshu Abhishek, Timothy Derdenger, Jaymo Kim, and Kannan Srinivasan: *Bias in generative ai*, 2024. https://arxiv.org/abs/2403.02726. 7

[18] Huang, Yutan, Chetan Arora, Wen Cheng Houng, Tanjila Kanij, Anuradha Madulgalla, and John Grundy: *Ethical concerns of generative ai and mitigation strategies: A systematic mapping study*, 2025. https://arxiv.org/abs/2502.00015. 7

[19] Singh, Aditi, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei: *Agentic retrieval-augmented generation: A survey on agentic rag*, 2025. https://arxiv.org/abs/2501.09136. 8

[20] Jeong, Soyeong, Kangsan Kim, Jinheon Baek, and Sung Ju Hwang: *Videorag: Retrieval-augmented generation over video corpus*, 2025. https://arxiv.org/abs/2501.05874. 8

[21] Su, Jinyan, Jennifer Healey, Preslav Nakov, and Claire Cardie: *Fast or better? balancing accuracy and cost in retrieval-augmented generation with flexible user control*, 2025. https://arxiv.org/abs/2502.12145. 8

[22] Wang, Liang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei: *Chain-of-retrieval augmented generation*, 2025. https://arxiv.org/abs/2501.14342. 8

[23] Gupta, Shailja, Rajesh Ranjan, and Surya Narayan Singh: *A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions*, 2024. https://arxiv.org/abs/2410.12837. 8

[24] Xie, Weijian, Xuefeng Liang, Yuhui Liu, Kaihua Ni, Hong Cheng, and Zetian Hu: *Weknow-rag: An adaptive approach for retrieval-augmented generation integrating web search and knowledge graphs*, 2024. https://arxiv.org/abs/2408.07611. 8

[25] Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela: *Retrieval-augmented generation for knowledge-intensive nlp tasks*, 2020. https://arxiv.org/abs/2005.11401. 8

[26] Zhu, Zulun, Tiancheng Huang, Kai Wang, Junda Ye, Xinghe Chen, and Siqiang Luo: *Graph-based approaches and functionalities in retrieval-augmented generation: A comprehensive survey*, 2025. https://arxiv.org/abs/2504.10499. 8

[27] Opoku, David Osei, Ming Sheng, and Yong Zhang: *Do-rag: A domain-specific qa framework using knowledge graph-enhanced retrieval-augmented generation*, 2025. https://arxiv.org/abs/2505.17058. 8

[28] Gao, Yunfan, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang: *Retrieval-augmented generation for large language models: A survey*, 2023. https://arxiv.org/abs/2312.10997. 8

[29] Wu, Jinyang, Shuai Zhang, Feihu Che, Mingkuan Feng, Chuyuan Zhang, Pengpeng Shao, and Jianhua Tao: *Pandora's box or aladdin's lamp: A comprehensive analysis revealing the role of rag noise in large language models*, 2024. https://arxiv.org/abs/2408.13533. 8, 9, 30, 31, 43

[30] Zhang, Qianchi, Hainan Zhang, Liang Pang, Yongxin Tong, Hongwei Zheng, and Zhiming Zheng: *Less is more: Compact clue selection for efficient retrieval-augmented generation reasoning*, 2025. https://arxiv.org/abs/2502.11811. 9, 31, 43

[31] Zhu, Kun, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin: *An information bottleneck perspective for effective noise filtering on retrieval-augmented generation*, 2024. https://arxiv.org/abs/2406.01549. 9, 31

[32] Liu, X. *et al.*: *Ica-rag: Information completeness guided adaptive retrieval-augmented generation for disease diagnosis*. IEEE Journal of Biomedical and Health Informatics, 2025. https://ieeexplore.ieee.org/document/11357045, doi to be added. 9, 31

[33] Xiao, Mengxi, Zihao Jiang, Lingfei Qian, Zhengyu Chen, Yueru He, Yijing Xu, Yuecheng Jiang, Dong Li, Ruey Ling Weng, Min Peng, Jimin Huang, Sophia Ananiadou, and Qianqian Xie: *Retrieval-augmented large language models for financial time series forecasting*, 2025. https://arxiv.org/abs/2502.05878. 9, 31

[34] Mirzadeh, Iman, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar: *Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models*, 2024. https://arxiv.org/abs/2410.05229. 9, 14, 31, 32

[35] Hu, Shengding, Yuge Tu, Xu Han, Chaoqun He, Ganqu Cui, Xiang Long, Zhi Zheng, Yewei Fang, Yuxiang Huang, Weilin Zhao, Xinrong Zhang, Zheng Leng Thai, Kaihuo Zhang, Chongyi Wang, Yuan Yao, Chenyang Zhao, Jie Zhou, Jie Cai, Zhongwu Zhai, Ning Ding, Chao Jia, Guoyang Zeng, Dahai Li, Zhiyuan Liu, and Maosong Sun: *Minicpm: Unveiling the potential of small language models with scalable training strategies*, 2024. https://arxiv.org/abs/2404.06395. 9, 15

[36] Zhang, Kun, Le Wu, Kui Yu, Guangyi Lv, and Dacao Zhang: *Evaluating and improving robustness in large language models: A survey and future directions*, 2025. https://arxiv.org/abs/2506.11111. 10

[37] Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, *et al.*: *Chain-of-thought prompting elicits reasoning in large language models*. Advances in neural information processing systems, 35:24824–24837, 2022. 10, 11

[38] Li, Tianle, Ge Zhang, Quy Duc Do, Xiang Yue, and Wenhu Chen: *Long-context llms struggle with long in-context learning*, 2024. https://arxiv.org/abs/2404.02060. 10

[39] Wang, Xuezhi, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou: *Self-consistency improves chain of thought reasoning in language models*, 2022. https://arxiv.org/abs/2203.11171. 11

[40] Yao, Shunyu, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan: *Tree of thoughts: Deliberate problem solving with large language models*, 2023. https://arxiv.org/abs/2305.10601. 11, 12

[41] Han, Jiayi, Liang Du, Hongwei Du, Xiangguo Zhou, Yiwen Wu, Weibo Zheng, and Donghong Han: *Slim: Let llm learn more and forget less with soft lora and identity mixture*, 2024. https://arxiv.org/abs/2410.07739. 13

[42] Shin, Haebin, Lei Ji, Yeyun Gong, Sungdong Kim, Eunbi Choi, and Minjoon Seo: *Generative prompt internalization*, 2024. https://arxiv.org/abs/2411.15927. 13, 22

[43] Yao, Ben, Yazhou Zhang, Qiuchi Li, and Jing Qin: *Is sarcasm detection a step-by-step reasoning process in large language models?* In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25651–25659, 2025. 13

[44] Ling, Zhan, Kang Liu, Kai Yan, Yifan Yang, Weijian Lin, Ting Han Fan, Lingfeng Shen, Zhengyin Du, and Jiecao Chen: *Longreason: A synthetic long-context reasoning benchmark via context expansion*, 2025. https://arxiv.org/abs/2501.15089. 14

[45] Chen, Guizhen, Weiwen Xu, Hao Zhang, Hou Pong Chan, Chaoqun Liu, Lidong Bing, Deli Zhao, Anh Tuan Luu, and Yu Rong: *Finereason: Evaluating and improving llms' deliberate reasoning through reflective puzzle solving*, 2025. https://arxiv.org/abs/2502.20238. 14

[46] Niu, Boye, Yiliao Song, Kai Lian, Yifan Shen, Yu Yao, Kun Zhang, and Tongliang Liu: *Flow: Modularized agentic workflow automation*, 2025. https://arxiv.org/abs/2501.07834. 14, 15

[47] Zhang, Guibin, Kaijie Chen, Guancheng Wan, Heng Chang, Hong Cheng, Kun Wang, Shuyue Hu, and Lei Bai: *Evoflow: Evolving diverse agentic workflows on the fly*, 2025. https://arxiv.org/abs/2502.07373. 14, 15

[48] Yin, Li and Zhangyang Wang: *Llm-autodiff: Auto-differentiate any llm workflow*, 2025. https://arxiv.org/abs/2501.16673. 15

[49] Li, Yifei, Rong Chen, Maruan Al-Shedivat, Ting Wang, and Min Wu: *BioMaster: Multi-agent system for automated bioinformatics analysis workflow.* bioRxiv, 2025. https://doi.org/10.1101/2025.01.23.634608. 15

[50] Subramanian, Shreyas, Vikram Elango, and Mecit Gungor: *Small language models (slms) can still pack a punch: A survey*, 2025. https://arxiv.org/abs/2501.05465. 15, 35

[51] Allal, Loubna Ben, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, Joshua Lochner, Caleb Fahlgren, Xuan Son Nguyen, Clémentine Fourrier, Ben Burtenshaw, Hugo Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and Thomas Wolf: *Smollm2: When smol goes big – data-centric training of a small language model*, 2025. https://arxiv.org/abs/2502.02737. 15

[52] Wang, Fali, Minhua Lin, Yao Ma, Hui Liu, Qi He, Xianfeng Tang, Jiliang Tang, Jian Pei, and Suhang Wang: *A survey on small language models in the era of large language models: Architecture, capabilities, and trustworthiness.* In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, page 6173–6183, New York, NY, USA, 2025. Association for Computing Machinery, ISBN 9798400714542. https://doi.org/10.1145/3711896.3736563. 15, 16, 17

[53] Vemulapalli, Venkata Kiran: *Choosing the right model for enterprise ai applications: A comprehensive analysis of small language models versus large language models in enterprise architecture*. European Modern Studies Journal, 9:275–284, September 2025. 16

[54] Kandala, Savitha Viswanadh, Pramuka Medaranga, and Ambuj Varshney: *Tinyllm: A framework for training and deploying language models at the edge computers*, 2024. https://arxiv.org/abs/2412.15304. 16

[55] Yu, Zezhu, Suqing Liu, Paul Denny, Andreas Bergen, and Michael Liut: *Integrating small language models with retrieval-augmented generation in computing education: Key takeaways, setup, and practical insights*. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, SIGCSETS 2025, page 1302–1308, New York, NY, USA, 2025. Association for Computing Machinery, ISBN 9798400705311. https://doi.org/10.1145/3641554.3701844. 16, 17, 35

[56] Scherer, Moritz, Luka Macan, Victor Jung, Philip Wiese, Luca Bompani, Alessio Burrello, Francesco Conti, and Luca Benini: *Deeploy: Enabling energy-efficient deployment of small language models on heterogeneous microcontrollers*, 2024. https://arxiv.org/abs/2408.04413. 16

[57] Aralimatti, Rakshit, Syed Abdul Gaffar Shakhadri, Kruthika KR, and Kartik Basavaraj Angadi: *Fine-tuning small language models for domain-specific ai: An edge ai perspective*, 2025. https://arxiv.org/abs/2503.01933. 16, 35

[58] Park, Chansung, Juyong Jiang, Fan Wang, Sayak Paul, and Jing Tang: *Llamaduo: Llmops pipeline for seamless migration from service llms to small-scale local llms*, 2025. https://arxiv.org/abs/2408.13467. 16

[59] Souza, Débora, Rohit Gheyi, Lucas Albuquerque, Gustavo Soares, and Márcio Ribeiro: *Code generation with small language models: A codeforces-based study*, 2025. https://arxiv.org/abs/2504.07343. 16

[60] Sheng, Lei and Shuai Shuai Xu: *Slm-sql: An exploration of small language models for text-to-sql*, 2025. https://arxiv.org/abs/2507.22478. 16

[61] Song, Huan, Deeksha Razdan, Yiyue Qian, Arijit Ghosh Chowdhury, Parth Patwa, Aman Chadha, Shinan Zhang, Sharlina Keshava, and Hannah Marlowe: *Learning from generalization patterns: An evaluation-driven approach to enhanced data augmentation for fine-tuning small language models*, 2025. https://arxiv.org/abs/2510.18143. 16

[62] Zhang, Hanling, Yayu Zhou, Tongcheng Fang, Zhihang Yuan, Guohao Dai, Wanli Ouyang, and Yu Wang: *Vocabtailor: Dynamic vocabulary selection for downstream tasks in small language models*, 2026. https://arxiv.org/abs/2508.15229. 16

[63] Malach, Eran: *Auto-regressive next-token predictors are universal learners*, 2023. https://arxiv.org/abs/2309.06979. 17

[64] Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu: *Exploring the limits of transfer learning with a unified text-to-text transformer*, 2023. https://arxiv.org/abs/1910.10683. 17

[65] Howard, Jeremy and Sebastian Ruder: *Universal language model fine-tuning for text classification*. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018. 17

[66] Lee, Jinhyuk, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang: *Biobert: a pre-trained biomedical language representation model for biomedical text mining*. Bioinformatics, 36(4):1234–1240, September 2019, ISSN 1367-4811. http://dx.doi.org/10.1093/bioinformatics/btz682. 17

[67] Ofer, Moshe, Orel Zamler, and Amos Azaria: *Tall – a trainable architecture for enhancing llm performance in low-resource languages*, 2025. https://arxiv.org/abs/2506.05057. 18

[68] Dong, Yi, Ronghui Mu, Gaojie Jin, Yi Qi, Jinwei Hu, Xingyu Zhao, Jie Meng, Wenjie Ruan, and Xiaowei Huang: *Building guardrails for large language models*, 2024. https://arxiv.org/abs/2402.01822. 18

[69] Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen: *Lora: Low-rank adaptation of large language models*, 2021. https://arxiv.org/abs/2106.09685. 18, 19, 40

[70] Wang, Luping, Sheng Chen, Linnan Jiang, Shu Pan, Runze Cai, Sen Yang, and Fei Yang: *Parameter-efficient fine-tuning in large models: A survey of methodologies*, 2024. https://arxiv.org/abs/2410.19878. 18

[71] Houlsby, Neil, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly: *Parameter-efficient transfer learning for nlp*, 2019. https://arxiv.org/abs/1902.00751. 18

[72] Ansell, Alan, Ivan Vulić, Hannah Sterz, Anna Korhonen, and Edoardo M. Ponti: *Scaling sparse fine-tuning to large language models*, 2024. https://arxiv.org/abs/2401.16405. 18

[73] Liu, Pengfei, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig: *Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing*, 2021. https://arxiv.org/abs/2107.13586. 18

[74] Xu, Xiaohan, Ming Li, Chongyang Tao, Tao Shen, Reynold Cheng, Jinyang Li, Can Xu, Dacheng Tao, and Tianyi Zhou: *A survey on knowledge distillation of large language models*, 2024. https://arxiv.org/abs/2402.13116. 20, 34, 40

[75] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean: *Distilling the knowledge in a neural network*, 2015. https://arxiv.org/abs/1503.02531. 20

[76] Gogate, Vaibhav, Ankit Sharma, and Rohan Patel: *Reducing hallucinations in llms via temperature scaling and intermediate layer matching*. TechRxiv preprint, 2024. https://www.techrxiv.org/articles/preprint/Reducing_Hallucinations_in_LLMs_via_Temperature_Scaling_and_Intermediate_Layer_Matching/2405.12345. 21

[77] Zhang, Songming, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu: *Dual-space knowledge distillation for large language models*, 2024. https://arxiv.org/abs/2406.17328. 21

[78] Binici, Kuluhan, Weiming Wu, and Tulika Mitra: *Generalizing teacher networks for effective knowledge distillation across student architectures*, 2024. https://arxiv.org/abs/2407.16040. 21

[79] Zhu, Xuekai, Biqing Qi, Kaiyan Zhang, Xinwei Long, Zhouhan Lin, and Bowen Zhou: *Pad: Program-aided distillation can teach small models reasoning better than chain-of-thought fine-tuning*, 2024. https://arxiv.org/abs/2305.13888. 21, 34

[80] Zheng, Lianmin, Wei Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica: *Judging llm-as-a-judge with mt-bench and chatbot arena*, 2023. https://arxiv.org/abs/2306.05685. 21

[81] Wang, Yizhong, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi: *Self-instruct: Aligning language models with self-generated instructions*, 2022. https://arxiv.org/abs/2212.10560. 21

[82] Wang, Xinfeng, Jin Cui, Yoshimi Suzuki, and Fumiyo Fukumoto: *Rdrec: Rationale distillation for llm-based recommendation*, 2024. https://arxiv.org/abs/2405.10587. 21

[83] Wu, Zhuofeng, He Bai, Aonan Zhang, Jiatao Gu, VG Vinod Vydiswaran, Navdeep Jaitly, and Yizhe Zhang: *Divide-or-conquer? which part should you distill your llm?*, 2024. https://arxiv.org/abs/2402.15000. 21

[84] Cui, Yu, Feng Liu, Pengbo Wang, Bohao Wang, Heng Tang, Yi Wan, Jun Wang, and Jiawei Chen: *Distillation matters: Empowering sequential recommenders to match the performance of large language models*. In *18th ACM Conference on Recommender Systems*, page 507–517. ACM, October 2024. http://dx.doi.org/10.1145/3640457.3688118. 22

[85] Jin, Renren, Jiangcun Du, Wuwei Huang, Wei Liu, Jian Luan, Bin Wang, and Deyi Xiong: *A comprehensive evaluation of quantization strategies for large language models*, 2024. https://arxiv.org/abs/2402.16775. 22, 23

[86] Milvus AI Quick Reference: *What is the role of quantization in llms?* https://milvus.io/ai-quick-reference/what-is-the-role-of-quantization-in-llms, 2025. 22

[87] Deci AI: *Post-training quantization (ptq) and quantization-aware training (qat)*. https://docs.deci.ai/super-gradients/V3_5/documentation/source/ptq_qat.html, 2024. 23

[88] Proskurina, Irina *et al.*: *When quantization affects confidence of large language models?* In *Findings of NAACL 2024*, 2024. https://aclanthology.org/2024.findings-naacl.124.pdf. 23

[89] Basyl, Nikita *et al.*: *Llm-fp4: 4-bit floating-point quantized transformers*. arXiv preprint arXiv:2310.16836, 2025. https://huggingface.co/papers/2310.16836. 23

[90] Lee, Dongyoung, Seungkyu Choi, and Ik Joon Chang: *Qrazor: Reliable and effortless 4-bit llm quantization by significant data razoring*, 2025. https://arxiv.org/abs/2501.13331. 23

[91] Russell, Stuart and Peter Norvig: *Artificial Intelligence: A Modern Approach*. Pearson, 4th edition, 2021. 24

[92] Wooldridge, Michael and Nicholas R. Jennings: *Intelligent agents: Theory and practice*. The Knowledge Engineering Review, 10(2):115–152, 1995. 24

[93] Weng, Lilian: *Llm powered autonomous agents*. https://lilianweng.github.io/posts/2023-06-23-agent/, 2023. 24, 25

[94] Masterman, Tula, Sandi Besen, Mason Sawtell, and Alex Chao: *The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey*, 2024. https://arxiv.org/abs/2404.11584. 24, 25

[95] Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela: *Retrieval-augmented generation for knowledge-intensive nlp tasks*, 2021. https://arxiv.org/abs/2005.11401. 25

[96] Yao, Shunyu, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao: *React: Synergizing reasoning and acting in language models*, 2023. https://arxiv.org/abs/2210.03629. 25

[97] Shinn, Noah, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao: *Reflexion: Language agents with verbal reinforcement learning*, 2023. https://arxiv.org/abs/2303.11366. 25

[98] Gu, Jiawei, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo: *A survey on llm-as-a-judge*, 2024. https://arxiv.org/abs/2411.15594. 26, 36, 37

[99] Li, Haitao, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu: *Llms-as-judges: A comprehensive survey on llm-based evaluation methods*, 2024. https://arxiv.org/abs/2412.05579. 26, 36, 37

[100] Zhu, Lianghui, Xinggang Wang, and Xinlong Wang: *Judgelm: Fine-tuned large language models are scalable judges*, 2023. https://arxiv.org/abs/2310.17631. 26

[101] Li, Dawei, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu: *From generation to judgment: Opportunities and challenges of llm-as-a-judge*, 2024. https://arxiv.org/abs/2411.16594. 26

[102] Pan, Qian, Zahra Ashktorab, Michael Desmond, Martin Santillan Cooper, James Johnson, Rahul Nair, Elizabeth Daly, and Werner Geyer: *Human-centered design recommendations for llm-as-a-judge*, 2024. https://arxiv.org/abs/2407.03479. 26, 37

[103] Tan, Sijun, Siyuan Zhuang, Kyle Montgomery, William Y. Tang, Alejandro Cuadron, Chenguang Wang, Raluca Ada Popa, and Ion Stoica: *Judgebench: A benchmark for evaluating llm-based judges*, 2024. https://arxiv.org/abs/2410.12784. 26, 37

[104] Lee, Yukyung, Joonghoon Kim, Jaehee Kim, Hyowon Cho, Jaewook Kang, Pilsung Kang, and Najoung Kim: *Checkeval: A reliable llm-as-a-judge framework for evaluating text generation using checklists*, 2024. https://arxiv.org/abs/2403.18771. 26, 37

[105] Li, Haitao, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu: *Llms-as-judges: A comprehensive survey on llm-based evaluation methods*, 2024. https://arxiv.org/abs/2412.05579. 27, 36, 42

[106] Fu, Jinlan, See Kiong Ng, Zhengbao Jiang, and Pengfei Liu: *Gptscore: Evaluate as you desire*, 2023. https://arxiv.org/abs/2302.04166. 27, 42

[107] Khattab, Omar, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts: *Dspy: Compiling declarative language model calls into self-improving pipelines*, 2023. https://arxiv.org/abs/2310.03714. 27

[108] Gu, Jiawei, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo: *A survey on llm-as-a-judge*, 2024. https://arxiv.org/abs/2411.15594. 27, 36

[109] Ye, Jiayi, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin Yu Chen, Nitesh V Chawla, and Xiangliang Zhang: *Justice or prejudice? quantifying biases in llm-as-a-judge*, 2024. https://arxiv.org/abs/2410.02736. 28, 37

[110] Shi, Lin, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi: *Judging the judges: A systematic study of position bias in llm-as-a-judge*, 2024. https://arxiv.org/abs/2406.07791. 28, 36

[111] Szymanski, Annalisa, Noah Ziems, Heather A. Eicher-Miller, Toby Jia Jun Li, Meng Jiang, and Ronald A. Metoyer: *Limitations of the llm-as-a-judge approach for evaluating llm outputs in expert knowledge tasks*, 2024. https://arxiv.org/abs/2410.20266. 28, 37

[112] Ye, Jiayi, Yanbo Wang, Yue Huang, Dongping Chen, Qihui Zhang, Nuno Moniz, Tian Gao, Werner Geyer, Chao Huang, Pin Yu Chen, Nitesh V Chawla, and Xiangliang Zhang: *Justice or prejudice? quantifying biases in llm-as-a-judge*, 2024. https://arxiv.org/abs/2410.02736. 28, 37

[113] Robertson, Stephen and Hugo Zaragoza: *The probabilistic relevance framework: Bm25 and beyond*. Foundations and Trends in Information Retrieval, 3:333–389, January 2009. 28, 43

[114] Merth, Thomas, Qichen Fu, Mohammad Rastegari, and Mahyar Najibi: *Superposition prompting: Improving and accelerating retrieval-augmented generation*, 2024. https://arxiv.org/abs/2404.06910. 31

[115] Fan, Tianyu, Jingyuan Wang, Xubin Ren, and Chao Huang: *Minirag: Towards extremely simple retrieval-augmented generation*, 2025. https://arxiv.org/abs/2501.06713. 31

[116] Xu, Ran, Wenqi Shi, Yuchen Zhuang, Yue Yu, Joyce C. Ho, Haoyu Wang, and Carl Yang: *Collab-rag: Boosting retrieval-augmented generation for complex question answering via white-box and black-box llm collaboration*, 2025. https://arxiv.org/abs/2504.04915. 31

[117] Zhang, Feiyuan, Dezhi Zhu, James Ming, Yilun Jin, Di Chai, Liu Yang, Han Tian, Zhaoxin Fan, and Kai Chen: *Dh-rag: A dynamic historical context-powered retrieval-augmented generation method for multi-turn dialogue*, 2025. https://arxiv.org/abs/2502.13847. 31

[118] Khatibi, Elahe, Ziyu Wang, and Amir M. Rahmani: *Cdf-rag: Causal dynamic feedback for adaptive retrieval-augmented generation*, 2025. https://arxiv.org/abs/2504.12560. 32

[119] Oliveira, Vinícius Di, Yuri Façanha Bezerra, Li Weigang, Pedro Carvalho Brom, and Victor Rafael R. Celestino: *Slim-raft: A novel fine-tuning approach to improve cross-linguistic performance for mercosur common nomenclature*, 2024. https://arxiv.org/abs/2408.03936. 32

[120] Levy, Mosh, Alon Jacoby, and Yoav Goldberg: *Same task, more tokens: the impact of input length on the reasoning performance of large language models*. In Ku, Lun Wei, Andre Martins, and Vivek Srikumar (editors): *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15339–15353, Bangkok, Thailand, August 2024. Association for Computational Linguistics. https://aclanthology.org/2024.acl-long.818/. 32

[121] Li, Zhuowan, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky: *Retrieval augmented generation or long-context LLMs? a comprehensive study and hybrid approach*. In Dernoncourt, Franck, Daniel Preoţiuc-Pietro, and Anastasia Shimorina (editors): *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 881–893, Miami, Florida, US, November 2024. Association for Computational Linguistics. https://aclanthology.org/2024.emnlp-industry.66/. 32

[122] Hu, Yunhai, Yilun Zhao, Chen Zhao, and Arman Cohan: *Mcts-rag: Enhancing retrieval-augmented generation with monte carlo tree search*, 2025. https://arxiv.org/abs/2503.20757. 32

[123] Zhao, Xuejiao, Siyan Liu, Su Yin Yang, and Chunyan Miao: *Medrag: Enhancing retrieval-augmented generation with knowledge graph-elicited reasoning for healthcare copilot*, 2025. https://arxiv.org/abs/2502.04413. 32

[124] Pipitone, Nicholas and Ghita Houir Alami: *Legalbench-rag: A benchmark for retrieval-augmented generation in the legal domain*, 2024. https://arxiv.org/abs/2408.10343. 32

[125] Gu, Yuxian, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang: *Miniplm: Knowledge distillation for pre-training language models*, 2024. https://arxiv.org/abs/2410.17215. 34

[126] Yam, Hong Meng and Nathan Paek: *Teaching tiny minds: Exploring methods to enhance knowledge distillation for small language models*. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 302–307, Miami, FL, USA, November 2024. Association for Computational Linguistics. https://aclanthology.org/2024.conll-babylm.27/. 34

[127] Ballout, Mohamad, Ulf Krumnack, Gunther Heidemann, and Kai Uwe Kühnberger: *Efficient knowledge distillation: Empowering small language models with teacher model insights*, 2024. https://arxiv.org/abs/2409.12586. 34

[128] Gu, Yanggan, Junzhuo Li, Sirui Huang, Xin Zou, Zhenghua Li, and Xuming Hu: *Capturing nuanced preferences: Preference-aligned distillation for small language models*, 2025. https://arxiv.org/abs/2502.14272. 34

[129] Ko, Jongwoo, Sungnyun Kim, Tianyi Chen, and Se Young Yun: *Distillm: Towards streamlined distillation for large language models*. In *International Conference on Machine Learning*, pages 24872–24895, 2024. https://proceedings.mlr.press/v235/ko24c.html. 34

[130] Erdogan, Lutfi Eren, Nicholas Lee, Siddharth Jha, Sehoon Kim, Ryan Tabrizi, Suhong Moon, Coleman Hooper, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami: *Tinyagent: Function calling at the edge*, 2024. https://arxiv.org/abs/2409.00608. 35

[131] Hashemzadeh, Maryam, Elias Stengel-Eskin, Sarath Chandar, and Marc Alexandre Cote: *Sub-goal distillation: A method to improve small language agents*, 2024. https://arxiv.org/abs/2405.02749. 35

[132] Fan, Tao, Guoqiang Ma, Yuanfeng Song, Lixin Fan, and Qiang Yang: *Ppc-gpt: Federated task-specific compression of large language models via pruning and chain-of-thought distillation*, 2025. https://arxiv.org/abs/2502.15857. 35

[133] Belcak, Peter, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov: *Small language models are the future of agentic ai*, 2025. https://arxiv.org/abs/2506.02153. 35

[134] Chen, Yi, JiaHao Zhao, and HaoHao Han: *A survey on collaborative mechanisms between large and small language models*, 2025. https://arxiv.org/abs/2505.07460. 35, 36

[135] Lamaakal, Ismail, Maleh Yassine, Khalid El Makkaoui, Ibrahim Ouahbi, Paweł Pławiak, Osama Alfarraj, May Almousa, and Ahmed Abd El-Latif: *Tiny language models for automation and control: Overview, potential applications, and future research directions*. Sensors, 25:34, February 2025. 35

[136] Zhou, Zhengping, Lezhi Li, Xinxi Chen, and Andy Li: *Mini-giants: "small" language models and open source win-win*, 2023. https://arxiv.org/abs/2307.08189. 35

[137] Uzor, GodsGift, Hasan Al-Qudah, Ynes Ineza, and Abdul Serwadda: *Guarding your conversations: Privacy gatekeepers for secure interactions with cloud- based ai models*. In *2025 19th International Conference on Semantic Computing (ICSC)*, page 235–242. IEEE, February 2025. http://dx.doi.org/10.1109/ICSC64641.2025.00040. 36

[138] Wang, Fali, Jihai Chen, Shuhua Yang, Ali Al-Lawati, Linli Tang, Hui Liu, and Suhang Wang: *A survey on collaborating small and large language models for performance, cost-effectiveness, cloud-edge privacy, and trustworthiness*, 2025. https://arxiv.org/abs/2510.13890. 36

[139] Narayan, Avanika, Dan Biderman, Sabri Eyuboglu, Avner May, Scott Linderman, James Zou, and Christopher Re: *Minions: Cost-efficient collaboration between on-device and cloud language models*, 2025. https://arxiv.org/abs/2502.15964. 36

[140] Li, Dawei, Bohan Jiang, Liangjie Huang, Alimohammad Beigi, Chengshuai Zhao, Zhen Tan, Amrita Bhattacharjee, Yuxuan Jiang, Canyu Chen, Tianhao Wu, Kai Shu, Lu Cheng, and Huan Liu: *From generation to judgment: Opportunities and challenges of llm-as-a-judge*, 2024. https://arxiv.org/abs/2411.16594. 36

[141] Chen, Nuo, Zhiyuan Hu, Qingyun Zou, Jiaying Wu, Qian Wang, Bryan Hooi, and Bingsheng He: *Judgelrm: Large reasoning models as a judge*, 2025. https://arxiv.org/abs/2504.00050. 36

[142] Zhao, Ruochen, Wenxuan Zhang, Yew Ken Chia, Weiwen Xu, Deli Zhao, and Lidong Bing: *Auto-arena: Automating llm evaluations with agent peer battles and committee discussions*, 2024. https://arxiv.org/abs/2405.20267. 36

[143] Tam, Zhi Rui, Cheng Kuang Wu, Yi Lin Tsai, Chieh Yen Lin, Hung yi Lee, and Yun Nung Chen: *Let me speak freely? a study on the impact of format restrictions on performance of large language models*, 2024. https://arxiv.org/abs/2408.02442. 40

[144] Banerjee, Debangshu, Tarun Suresh, Shubham Ugare, Sasa Misailovic, and Gagandeep Singh: *Crane: Reasoning with constrained llm generation*, 2025. https://arxiv.org/abs/2502.09061. 40

[145] Shorten, Connor, Charles Pierse, Thomas Benjamin Smith, Erika Cardenas, Akanksha Sharma, John Trengrove, and Bob van Luijt: *Structuredrag: Json response formatting with large language models*, 2024. https://arxiv.org/abs/2408.11061. 40

[146] Wang, Zhaoyang, Jinqi Jiang, Huichi Zhou, Wenhao Zheng, Xuchao Zhang, Chetan Bansal, and Huaxiu Yao: *Verifiable format control for large language model generations*, 2025. https://arxiv.org/abs/2502.04498. 40

[147] Dettmers, Tim, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer: *Qlora: Efficient finetuning of quantized llms*, 2023. https://arxiv.org/abs/2305.14314. 40, 47

[148] Yang, An, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu: *Qwen3 technical report*, 2025. https://arxiv.org/abs/2505.09388. 46

[149] Grattafiori, Aaron, Abhimanyu Dubey, Abhinav Jauhri, *et al.*: *The llama 3 herd of models*, 2024. https://arxiv.org/abs/2407.21783. 46

[150] Mao, Yulong, Kaiyu Huang, Changhao Guan, Ganglin Bao, Fengran Mo, and Jinan Xu: *Dora: Enhancing parameter-efficient fine-tuning with dynamic rank distribution*, 2024. https://arxiv.org/abs/2405.17357. 62

Appendix A

Appendix A: LLMQuoter Web APP

To facilitate interactive testing and demonstration of the fine-tuned quote extraction models evaluated in this thesis, a lightweight web application named **LLMQuoter Web** was developed. The application allows users to input a question and a context passage, select one of the available models, and observe the extracted quotes highlighted directly within the original context. The source code is publicly available on GitHub[1].

A.1 Purpose and Design

The primary goal of the application is to provide a practical, visual interface for comparing the behavior of the three LoRA fine-tuned student models—Llama 3.2:1B, Llama 3.2:3B, and Qwen 3:4B—alongside a GPT-4o-mini baseline. By enabling side-by-side experimentation with the same question and context across different models, the tool makes it possible to qualitatively observe how model scale and fine-tuning affect quote selection, formatting compliance, and overall extraction quality.

The application was intentionally kept minimal: it requires no database, no authentication, and no frontend framework. This design ensures that the focus remains entirely on the model inference pipeline rather than on application infrastructure.

A.2 Architecture

LLMQuoter Web is built as a Flask-based single-page application. The backend orchestrates inference through LangChain, which abstracts the communication layer between the application and the underlying model providers. The three LoRA-adapted models are served locally via Ollama using GGUF-formatted checkpoints, while the GPT-4o-mini

[1] https://github.com/yurifacanha/llmquoter-web

baseline is accessed through the OpenAI API. The prompt template used at inference time is identical to the one employed during fine-tuning, ensuring consistency between training and deployment conditions.

When a user submits a query, the backend invokes the selected model with the question and context, parses the structured response to extract quotes delimited by `##begin_quote## ... ##end_quote##` markers, and maps each quote back to its position in the original context. The mapping uses a two-pass strategy: an exact string match is attempted first, and if it fails—as may occur with minor tokenization artifacts from smaller models—a fuzzy matching fallback based on RapidFuzz is applied. Overlapping spans are merged, and the result is returned as an HTML fragment with highlighted regions that the frontend renders directly.

A.3 Relevance to the Thesis

The web application serves as a complementary validation tool for the quantitative results reported in Chapter 6. While the experimental evaluation relies on automated metrics computed over 600 test samples, the application enables qualitative inspection of individual cases, making it easier to identify patterns such as over-extraction, missed evidence, or formatting inconsistencies. It also demonstrates that the fine-tuned models can be deployed in a standard inference environment using off-the-shelf tools (Flask, LangChain, Ollama), reinforcing the thesis claim that the proposed pipeline is practical and accessible for real-world use.

To further illustrate the generalization capacity of the fine-tuned models beyond the training distribution, Figures A.1 and A.2 present two examples of the system operating on entirely unseen contexts drawn from real-world sources unrelated to the dataset used during training. In Figure A.1, the input context is the introductory section of the Wikipedia article on Santos Dumont, and the model is queried about his life and contributions. In Figure A.2, the context is a public health report from the European Centre for Disease Prevention and Control (ECDC) concerning Nipah virus disease cases reported in West Bengal, India, and the model is queried about the virus and associated risk levels. Both examples involve topics and writing styles that were not present during fine-tuning, yet the models demonstrate coherent quote extraction and answer grounding behavior consistent with what was observed on the test set. These cases illustrate how the proposed pipeline could be applied in practice across diverse domains and document types, supporting the broader claim that the approach is not narrowly tailored to a specific corpus but generalizes to arbitrary retrieval contexts.

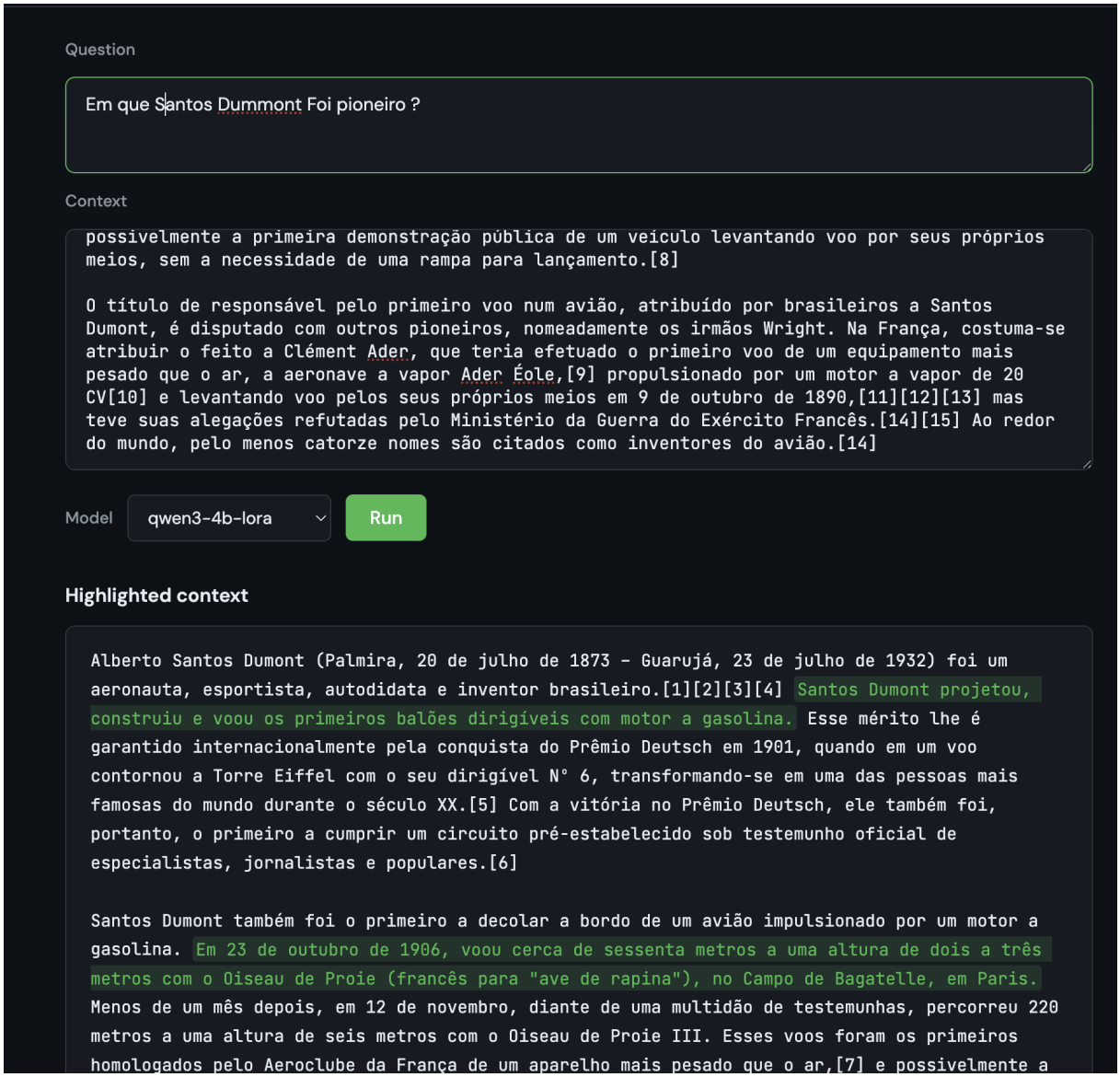

Figure A.1: Web application screenshot showing the model applied to an unseen context: the introductory section of the Wikipedia article on Santos Dumont. The query and retrieved quotes are entirely outside the training distribution, demonstrating real-world generalization.

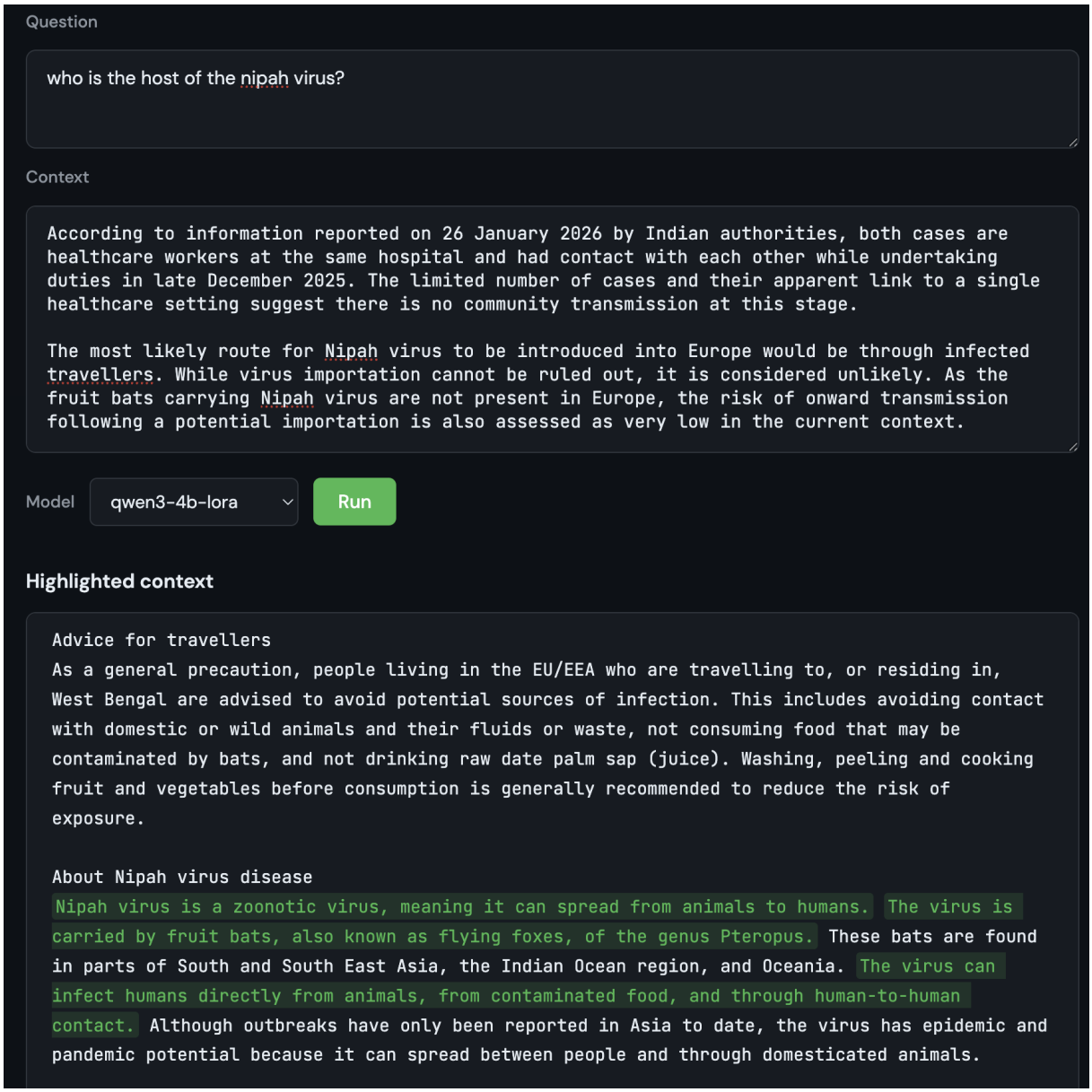

Figure A.2: Web application screenshot showing the model applied to a public health document: an ECDC report on Nipah virus disease cases in West Bengal, India. As with the previous example, neither the topic nor the document style appeared during training, yet the model produces well-grounded and relevant extractions.