BSc Computational and Data Science
CDS904 Bachelor Thesis
Dozent: Dr.sc.ETH. Florian Herzog

Magical RDF Triples and how to synthetize them

Untertitel

Florian Klessascheck[1,*]

[1] *Fachhochschule Graubünden*

[*] *E-Mail Adressen: florian.klessascheck@fhgr.ch*

April 10, 2026

Abstract

Contents

1 Introduction

Extracting structured data from unstructured text is a old and well known problem in the field of Natural Language Processing (NLP). (Ni et al., 2023) With the upcomming of Large Language Model (LLM)s in recent years, the problem has seen a new wave of interest, research and proposed solutions. (Ni et al., 2023) However many of those solutions are based on simple prompting unable to capture the full semantic and syntactic complexity of many target data structures, even when using state-of-the-art models such as GPT-4. (W. Zhang et al., 2024) Many solutions also focus on extracting one specific data schema, making them not generalizable to other schemas without further, significant effort and research put in. (Ni et al., 2023) An increasingly prominent alternative to prompting is the finetuning of LLMs, where a general-purpose base model is further trained on annotated data to specialise in data extraction tasks. When revieweing literature, the impression is made that finetuning performs better than prompting, often outperforming it by 10-20% on metrics such as exact match accuracy. (W. Zhang et al., 2024) On the downside however finetuning requires more time, computational resources and annotated datasets for training. (Norouzi et al., 2025) Furthermore this approach is often used to extract into one specific data schema, which limits its generalizability even more compared to prompting because of the higher effort and resources required. (Dagdelen et al., 2024a)

A more generalistic approach to this problem could be a finetuned LLM that is able to extract data into an arbitrary data schema, given a description of the schema as part of the input. This would allow to extract into any data schema without the need for further finetuning, making it much more generalizable. (Ni et al., 2023) Another benefit of such a model would be easier deployment into data extraction pipelines, as neither a task-specific finetuning step nor prompt engineering would be required.

The goal of this thesis is to examine the effect of two specific factors on the accuracy of finetuning a general-purpose LLM for data extraction tasks. Both factors have received limited attention in the existing literature. The first is the introduction of additional grammar terminal tokens of a data serialization language during finetuning. The second is the use of an encoder-decoder transformer architecture for data extraction tasks compared to a decoder-only architecture. The associated research questions and methodology are elaborated in the following chapter.

2 Methodology and Research Question

2.1 Methodology

2.2 Research Question

The following research questions are examined in the subsequent chapters

1 Does introducing additional grammar terminal tokens of a data serialization language during finetuning for extraction of serialized data from plain text to a transformer model yield decreased cross entropy loss and increased F1 score on the test set.

2 Does using a encoder-decoder transformer architecture for data extraction tasks yield decreased cross entropy loss and increased F1 score on the test set compared to a decoder only transformer architecture.

2.2.1 Implementation Plan

To address the research questions, four models are to be finetuned and compared. Finetuning will be conducted on a dataset of plain-text documents with RDF triples in a specific ontology as target labels. To ensure generalization across different domains, the dataset will be assembled from multiple existing datasets and augmented with additional datasets whose labels have been converted to the RDF format.

- Model 1: A decoder only transformer model finetuned for data extraction without additional grammar terminal tokens.
- Model 2: A decoder only transformer model finetuned for data extraction with additional grammar terminal tokens.
- Model 3: An encoder-decoder transformer model finetuned for data extraction without additional grammar terminal tokens.
- Model 4: An encoder-decoder transformer model finetuned for data extraction with additional grammar terminal tokens.

2.2.2 Thoughts on question 1

The attention mechanism of transformer models operates through matrices of learned weights that determine which tokens attend to which other tokens. (Vaswani et al., 2023) By introducing grammar terminal symbols as dedicated tokens, each such construct is represented by a single token rather than a variable-length sequence of multiple tokens. (Vaswani et al., 2023) describes attention as a weighted sum over all positions. From this it follows that if a grammar construct spans multiple tokens, the attention weights must be distributed and coordinated across all of them for the model to recognise the construct as a single semantic unit. Further, research on attention sinks suggests that attention heads benefit from having a single dedicated token to attend to when not actively processing, allowing them to fire sharply on semantically relevant targets when they do. (Barbero et al., 2025) This gives rise to the hypothesis that a dedicated terminal token provides exactly such an unambiguous target, reducing the coordination burden across sub-tokens and thereby enabling more precise attention allocation, which in turn yields improved extraction accuracy during finetuning.

2.2.3 Thoughts on question 2

The extraction of structured data from unstructured text is when abstracted enough a sequence to sequence problem. (Dunn et al., 2022) Sequence to sequence problems can also be viewed as a translation problem, where the input sequence provides guidance and semantic meaning and the output sequence is a representation of the input semantics in a different grammatical and syntactical form. (Vaswani et al., 2023) Encoder-decoder transformer architectures haven been initially designed for sequence to sequence translation problems (Vaswani et al., 2023) but are presently primarly used as decoder only generational models. (Alt et al., 2019) The second research question is therefore grounded in the hypothesis that encoder-decoder architectures are inherently better suited for sequence-to-sequence problems and should consequently yield superior results on data extraction tasks compared to decoder-only architectures, assuming that data extraction can be adequately modelled as a sequence-to-sequence problem. In addition, recent work adapting decoder-only models to encoder-decoder architectures shows that, under a similar inference budget, encoder-decoder LLMs achieve comparable or better pretraining performance but substantially better finetuning performance than their decoder-only counterparts. (B. Zhang, Moiseev, et al., 2025) This advantage holds consistently across pretraining objectives and model scales and is especially pronounced after instruction tuning. (B. Zhang, Moiseev, et al., 2025) Furthermore, the performance gains cannot be attributed solely to additional pretraining compute, suggesting that the structural separation between input processing and output generation inherent to the encoder-decoder design itself plays a decisive role. (B. Zhang, Moiseev, et al., 2025)

3 State of Research

The extraction of structured data from unstructured plain text is a persistent and well-documented challenge in NLP. Text structuralization is the task of converting unstructured natural language into structured data, encompassing subtasks such as named entity recognition, relation extraction, and event extraction. These structured results are beneficial for many downstream tasks including data mining and text analysis. (Ni et al., 2023) With the emergence of LLMs, the field has progressively shifted away from traditional algorithmic NLP approaches towards transformer-based decoder-only architectures. Existing deep learning based text structuralization models can be divided into different categories such as pipeline, end-to-end, and joint modeling approaches. Most of them require multiple sub-models to be cascaded, which is complicated and can lead to errors accumulating and propagating between models. (Ni et al., 2023) In the broader Information Extraction (IE) literature, methods are commonly categorized as knowledge-based, statistical machine learning, and deep learning approaches. (Norouzi et al., 2025) This chapter surveys the current state of research across three principal strategies for LLM-based structured data extraction: prompt engineering, finetuning, and retrieval-augmented assistive systems.

3.1 Prompt-Based Extraction

Prompt-based extraction draws on the pre-trained capabilities of LLMs without modifying their weights. It can be divided into zero-shot prompting, where only task instructions are provided, and few-shot prompting, where a small number of annotated input-output examples are included in the context.

3.1.1 Zero-Shot Prompting

(Ni et al., 2023) propose an instruction-based method for text structuralization and information extraction, where a prefix instruction indicating the desired task and a suffix instruction controlling the output format are added to the input text before feeding it to a large language model. Experiments using GPT-3 and FLAN-T5 show that only instruction-tuned models have the ability to recognize instructions and perform the corresponding tasks under zero-shot conditions. This renders annotated training data unnecessary, though the model must possess sufficient pre-trained knowledge of the target domain and output format.

The SPIRES method (Structured Prompt Interrogation and Recursive Extraction of Semantics) takes a more systematic approach to zero-shot extraction by using recursive template-

based prompting. The system defaults to GPT-3.5-turbo and generates pseudo-YAML structures conforming to a specified template. The completion result is parsed through heuristic methods since the LLM output is not guaranteed to be strict YAML, and the system recursively calls itself on inlined attributes to extract nested structures. (Caufield et al., 2024)

(Carta et al., 2023) present an iterative zero-shot approach for knowledge graph construction that decomposes the task into candidate triplet extraction, entity and predicate resolution, and schema inference, each performed through task-specific prompting of GPT-3.5. The system uses detailed system prompts to tell the model how to behave and formats the data within user prompts. This pipeline achieved a precision of 98.82% for entity extraction, 85.71% for entity typing, and 75.31% for relation extraction in an open-domain zero-shot setting, demonstrating that carefully structured prompts can guide LLMs to produce meaningful structured outputs without any parameter updates. (Carta et al., 2023)

However, zero-shot approaches are subject to notable limitations. (W. Zhang et al., 2024) found that zero-shot prompting resulted in poor performance across all five chemical text mining tasks, even when using GPT-4, as the results did not meet the required formatting requirements.

3.1.2 Few-Shot Prompting

Few-shot prompting mitigates this by supplying annotated examples within the prompt context, enabling the model to infer extraction patterns at inference time.

(Hu et al., 2025) demonstrate that by providing one to three annotated examples, a few-shot learning approach significantly enhances an LLM's ability to generalize to unseen table layouts and document types with minimal reliance on extensive labeled datasets. The annotated examples included in the prompt demonstrate the mappings between input text segments and JSON fields, enabling the LLM to learn the semantic relationships necessary for extraction. Their mechanism achieves remarkable performance even on table layouts and document structures it has not encountered before.

Brach et al. systematically evaluate the ability of LLMs to convert unstructured recipe text into the structured Cooklang format. Through comprehensive testing of four models (GPT-4o, GPT-4o-mini, Llama3.1:70b, and Llama3.1:8b), they find that GPT-4o with few-shot prompting achieves breakthrough performance with a ROUGE-L score of 0.9722 and a WER of 0.0730, demonstrating that LLMs can reliably transform domain-specific unstructured text into structured formats without extensive training. (Brach et al., 2025)

Increasing the number of few-shot examples leads LLMs to extract more structured outputs.

However, the upper limit of in-context learning is constrained by the maximum input length due to memory limitations. (W. Zhang et al., 2024)

3.1.3 Structured Output Formats and Constrained Decoding

A recurring obstacle in prompt-based methods is the difficulty of guaranteeing that generated output adheres to a predefined schema. Constrained decoding describes a group of decoding algorithms that impose direct constraints on the tokens or their distributions a model can generate at each step. By pruning candidate tokens that violate format restrictions, constrained decoding can guarantee the validity of the answer even if the model does not fully understand and comply with the format instruction on its own. (Huang et al., 2025)

The OmniStruct benchmark formulates text-to-structure tasks as schema-following JSON generation, where the goal is to generate correct task output in JSON format that strictly follows the JSON schema provided by the task prompt. Answers that cannot be correctly parsed, either due to being invalid JSON or missing required fields in the schema, are treated as incorrect regardless of their content. (Huang et al., 2025)

(Hu et al., 2025) employ a two-step structured output generation process to ensure reliability. The initial input is processed to produce a preliminary result, which is validated against a predefined JSON schema using regular expressions. If the validation fails, the system invokes a second prompt to enforce strict adherence to the schema, and the reprocessed output is subjected to a second validation round.

3.2 Finetuning Based Extraction

Finetuning adapts pre-trained language models to specific extraction tasks by updating their parameters on annotated training data, generally yielding higher accuracy than prompting, particularly for domain-specific tasks.

3.2.1 Supervised Finetuning for Information Extraction

(Dagdelen et al., 2024b) present the LLM-NERRE method, where a domain expert defines the desired output structure and annotates approximately 100 to 500 text passages. The LLM is then finetuned on these examples, and the resulting model is able to accurately output extracted information in the same structured representation. This method shows strong performance using both GPT-3 and Llama-2 on both sentence-level and document-

level materials information extraction. A minimum of approximately 20 training examples is needed for GPT-3 to learn a desired output format when using simple sentence-type schemas, with a sharp increase in correctly structured outputs at that threshold. (Dagdelen et al., 2024a)

KnowCoder proposes a code-style schema representation method to uniformly transform different schemas into Python classes, enabling complex schema information such as constraints among tasks to be captured in an LLM-friendly manner. Their two-phase learning framework first enhances schema understanding via code pretraining on approximately 1.5 billion automatically constructed data points, then improves schema following via instruction tuning. After code pretraining, KnowCoder already attains remarkable generalization ability and achieves relative improvements of 49.8% F1 compared to LLaMA2 under the few-shot setting. After instruction tuning, it further achieves up to 12.5% and 21.9% improvements compared to state-of-the-art baselines under zero-shot and low-resource settings respectively. (Z. Li et al., 2024)

(Alt et al., 2019) demonstrate that introducing language modeling as an auxiliary objective during finetuning improves generalization and leads to faster convergence for relation extraction. Their approach converts the structured, task-specific input into an ordered sequence that can be fed directly to the model without architectural changes, by encoding relation arguments and the sentence containing the entity pair with special delimiters.

3.2.2 Finetuning versus Prompting

When comparing finetuning and prompting directly, the literature consistently demonstrates that finetuning yields superior results for structured extraction tasks.

(W. Zhang et al., 2024) find that even when the same number of examples is provided for in-context learning as for finetuning, the finetuned model typically outperforms by 10-20% on metrics like exact match accuracy and modified BLEU score. This is attributed to information loss in in-context learning, while finetuning adjusts parameters to learn extraction patterns and thus maintains higher accuracy. Their study also identifies two key features of finetuning LLMs: rapid performance convergence with small amounts of data, where hundreds of high-quality training examples are enough to train an effective extractor, and efficient training generalization requiring only a few epochs and a low learning rate.

(Gajo & Barrón-Cedeño, 2025) find that finetuned models achieve much greater performance than ChatGPT davinci models that are only given in-context learning examples without any parameter updates. Considering that they only finetuned their models for 200 steps (i.e., 1600 training examples per setting), they conclude that learning to predict the structure of a sentence during finetuning is much more dependent on the number of examples seen,

the training technique, and parameter count than on the zero-shot and few-shot reasoning capabilities the model possessed prior to finetuning. Notably, smaller finetuned models can outperform much larger models in a zero-shot setting, suggesting that learning the specific characteristics of the dataset is more important than the sheer size of the base model.

(Ghanem & Cruz, 2025) formalize the Text-to-Knowledge-Graph task as the transformation function $T = f_{\theta,\pi}(x)$, where θ represents the language model's parameters and π denotes the prompting strategy, and explore three strategies: zero-shot prompting, few-shot prompting, and fine-tuning. Each approach has distinct trade-offs with respect to performance, computation resources, training time, domain adaptation, and training data requirements. Table 1 summarizes these trade-offs.

Criterion	**Zero-Shot**	**Few-Shot**	**Fine-Tuning**
Extraction Performance	Low	Moderate	High
Computation Resources	None	None	High
Training Time	None	None	Moderate–High
Domain Adaptation	Limited	Moderate	Strong
Cross-Domain Generalization	High	High	Reduced
Training Data Required	None	Few examples	Hundreds–Thousands

Table 1: Trade-offs between LLM adaptation strategies for the Text-to-Knowledge-Graph task.

3.2.3 Domain Adaptation and Transfer

As shown in Table 1, finetuning offers stronger domain adaptation capabilities than prompting, as it allows the model to learn domain-specific patterns and representations through parameter updates. However, this comes at the cost of reduced cross-domain generalization, as the model may overfit to the training domain and struggle to perform well on unseen domains without further finetuning.

(Norouzi et al., 2025) show further evidence that domain shift presents a significant challenge for finetuned models. They find that models finetuned on general-purpose training data exhibit a performance decline when evaluated on domain-specific social science data, with decreases ranging from 4-18% in F1 macro average depending on the model. Their research also reports an reverse effect: models finetuned on social science data perform worse when evaluated on general-purpose test data, indicating bidirectional domain mismatch. Notably, finetuning on domain-specific data can increase in-domain performance significantly, with BERT finetuned on social science data achieving an F1 score of 0.89 compared to 0.74 when finetuned only on general-purpose data.

(B. Li et al., 2024) study a Transformer-based architecture that is first pre-trained on mixed data and then equipped with task-specific adaptation layers for structured and unstructured data integration and relation extraction, demonstrating that task-adaptive pre-training can improve performance on downstream extraction tasks.

(Norouzi et al., 2025) provide additional evidence that increasing training-set diversity can mitigate domain-shift effects. In their causal-claim extraction experiments, models trained on merged general-purpose and social-science data achieve more balanced cross-domain performance than single-domain finetuned variants, indicating improved robustness under distribution shift. Likewise, in structured knowledge grounding, adding general instruction data to the training mixture improves held-out performance and reduces overfitting to task-specific formats, suggesting that data diversity supports better transfer to unseen tasks. (Zhuang et al., 2024) Together, these findings suggest that diverse or mixed-domain training is a practical mitigation strategy for domain shift, even if it does not fully eliminate cross-domain degradation.

3.2.4 Parameter-Efficient Finetuning

Full finetuning of large language models requires substantial computational resources, making parameter-efficient finetuning (Parameter Efficient Finetuning (PEFT)) techniques an attractive alternative as they update only a small subset of parameters.

PEFT focuses on updating a selective subset of the model's parameters, contrasting with conventional finetuning where a substantial portion of the parameters is modified, leading to high computational costs. Low-Rank Adaptation (LoRA) refines the model's efficiency by optimizing a small, critical subset of parameters, thus minimizing the computational overhead while maintaining or even enhancing the model's performance in domain-specific applications. NEFTune introduces an innovative approach by incorporating random noise into the embedding vectors during finetuning, thereby improving the model's ability to follow instructions and enhancing its generalization capacity. (Susnjak et al., 2024)

(Norouzi et al., 2025) employ Quantized Low-Rank Adaptation (QLoRA) to finetune LLAMA2-7b and Mistral-7b models for causal claim extraction. This method allows finetuning of large models on a single GPU by adjusting only a small subset of the model's parameters. Their best-performing model, BERT finetuned on domain-specific data, achieves an F1 macro score of 0.89, highlighting that smaller encoder-based models can still outperform larger decoder-only models when properly finetuned on targeted datasets.

(Huang et al., 2025) demonstrate the possibility of distilling the text-to-structure capabilities of GPT-4o into a much smaller, more cost-efficient model. They finetune Llama3.1-8B-

Instruct for one epoch with a learning rate of 1×10^{-5} and a batch size of 64 on synthetic instruction tuning data. Without using any supervised data, the resulting OmniStruct-8B outperforms GPT-4o on 5 of 6 NER datasets and on relation extraction tasks, and generalizes well to unseen tasks and schemas never seen during fine-tuning.

3.3 Assistive Systems for Improving Extraction Precision

Beyond standalone prompting or finetuning, a number of approaches augment LLM-based extraction with external systems, most notably retrieval-augmented generation (RAG) and schema retrieval mechanisms.

3.3.1 Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) is a framework that combines the capabilities of large language models with external knowledge sources through a retrieval mechanism. Unlike traditional language models that generate text based solely on the text's internal representation, RAG models retrieve relevant information from a knowledge base and integrate this into the generation process. The theoretical principles of RAG stem from the need to enhance language models with the ability to access and utilize external, structured knowledge, in response to the limitations of traditional LLMs that rely solely on their pre-trained parameters for knowledge, which can be outdated or incomplete. (Susnjak et al., 2024)

Susnjak et al. experimentally compare finetuning and RAG approaches for systematic literature review automation. They find that NEFTune achieves the highest factual accuracy at 89.2% of responses being supported, followed by LoRA at 87.7%. RAG with auto-extracted data achieves 84.7%. However, combining the best finetuned model with RAG actually reduces performance to 63.2%, suggesting that finetuned LLMs may struggle to integrate additional context from the RAG process. (Susnjak et al., 2024) This finding underscores that naively combining RAG with fine-tuning does not necessarily improve results and may introduce conflicting signals.

3.3.2 Schema Retrieval for Extraction

(B. Zhang & Soh, 2024) propose the EDC (Extract-Define-Canonicalize) framework that employs a trained Schema Retriever to efficiently search schemas in a manner similar to retrieval-augmented generation. The Schema Retriever projects the schema components and the input text to a vector space such that cosine similarity captures the relevance between

them. For refinement, the schema retriever retrieves the top-10 most relevant relations from the schema as candidate relations, providing richer context for the LLM's extraction phase.

The refinement process in EDC leverages data generated in earlier iterations to enhance extraction quality. Candidate entities and relations from both the LLM and the schema retriever provide a richer pool of candidates, addressing issues where the absence of entities or relations impairs the LLM's extraction effectiveness. An ablation study shows that removing the schema retriever's relations from the refinement hint consistently reduces F1 scores across all three evaluated datasets, confirming the importance of the retrieval component. (B. Zhang & Soh, 2024)

(Gajo & Barrón-Cedeño, 2025) use a knowledge base constructed from previous extractions which they query via RAG in their knowledge graph construction system. Using small LLMs such as GPT-2, Falcon 7B, and LLaMA 13B to predict triples in zero-shot and few-shot settings, they demonstrate that RAG can supplement smaller models' capabilities. However, RAG still involves additional parameters and latency in the forward pass as well as a computational cost for the initial vectorization necessary to store the knowledge base.

3.3.3 Iterative Refinement and Verification

A number of approaches employ iterative strategies in which the LLM's output is verified and refined across multiple passes.

(B. Zhang & Soh, 2024) demonstrates that a single refinement iteration with schema-retrieved hints improves F1 scores consistently for all tested LLMs (GPT-4, GPT-3.5-turbo, and Mistral-7b) across the WebNLG, REBEL, and Wiki-NRE datasets. For example, on WebNLG with GPT-4, the Partial F1 score improves from 0.783 for EDC to 0.820 for EDC+R. Additional refinement iterations yield further improvements, though with diminishing returns.

The two-step validation approach employed by Yang et al. for structured output generation demonstrates that post-processing pipelines can enforce schema compliance in cases where the LLM's initial output contains extraneous or malformed content. If the initial validation fails, a secondary prompt enforces strict adherence to the JSON schema, ensuring that only compliant data is retained. (Hu et al., 2025)

3.4 Summary

The surveyed literature reveals a broad methodological spectrum for LLM-based structured data extraction. Prompt-based methods present the lowest barrier to entry, as they require neither annotated training data nor parameter updates, but are constrained by context window limitations and consistently underperform on domain-specific tasks. Finetuning yields substantially higher accuracy, with reported improvements of 10-20% over in-context learning on comparable data budgets, and parameter-efficient techniques such as LoRA and QLoRA make this approach feasible even under limited hardware constraints. Assistive systems such as RAG and schema retrieval can further enhance extraction precision by supplying relevant external context, though integrating RAG with finetuning requires deliberate design to avoid the introduction of conflicting signals. The field continues to advance toward unified frameworks capable of handling diverse schemas and output formats, with constrained decoding emerging as a complementary mechanism to enforce the structural validity of generated outputs.

4 KGs in RAG systems

One of the most promising usages of KGs is the current emergence of RAG systems. This chapter will therefore explore the purpose of RAG systems in IE and Question Answering (QA) tasks, the common issues classically associated with RAG systems, and how KG retrieval can help to mitigate a subset of these issues.

4.1 Why RAG

LLMs are known to produce ungrounded fact-like statements, a phenomenon commonly referred to as hallucination. Hallucination in generative AI refers to the generation of content that is fabricated, misleading, or not based on factual data, and has become a major issue as LLMs are increasingly used in real-world applications. (Yu & McQuade, 2025) This is especially a problem in QA tasks where grounding the answer in factual evidence is mandatory. It has been shown that providing an LLM during inference with relevant factually verified information leads to a significant reduction in hallucination and an increase in answer accuracy, as RAG explicitly grounds the generation in retrieved documents that serve as up-to-date evidence. (Oche et al., 2025) This shifts the problem of factual knowledge from the LLM's internal weights to an outside retrieval system which can employ established search methods such as Term Frequency-Inverse Document Frequency (TF-IDF), Best Matching 25

(BM25), or dense vector search to retrieve relevant information from a large knowledge base. Retrieval augmentation serves as a live memory for the LLM: it supplies factual grounding from an external knowledge base while the language model creates fluent and contextually relevant text. (Oche et al., 2025) This approach, commonly referred to as RAG, has become increasingly popular in recent years.

4.2 Basic function of a DocumentRAG System

A RAG pipeline in its most fundamental form operates across two distinct phases: an offline ingestion phase, in which a document corpus is preprocessed and indexed, and an online inference phase, in which a user query is answered by retrieving relevant passages and conditioning a generative model on them. Both phases are illustrated in 1.

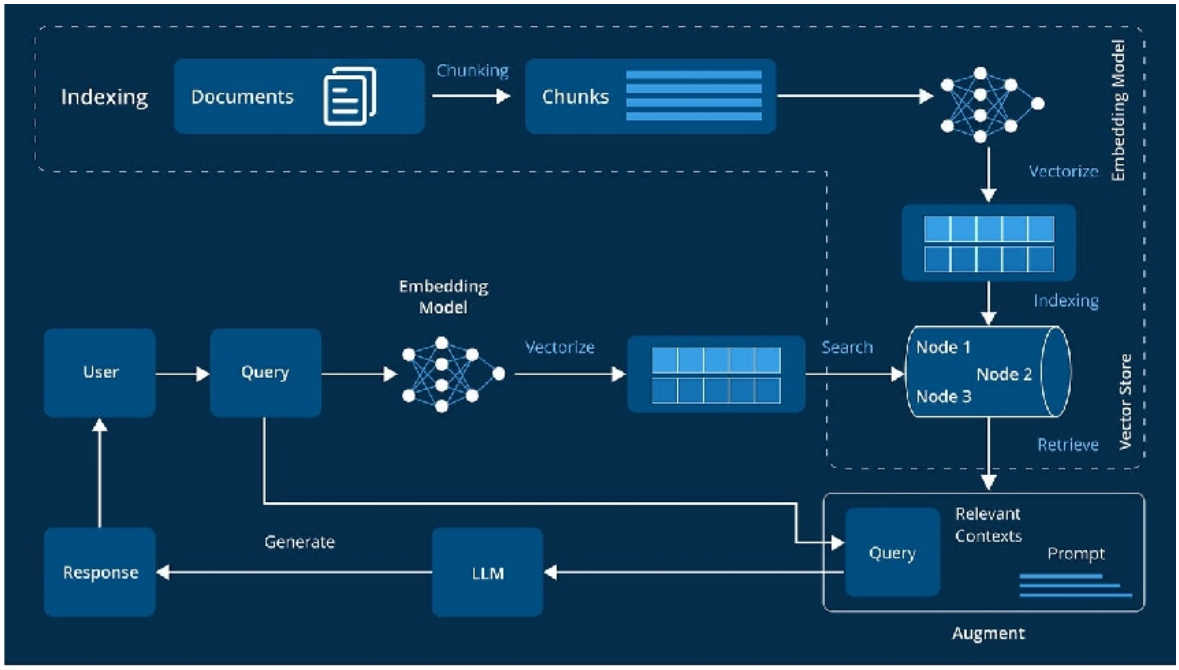

Figure 1: Illustration of a RAG Architecture.

Figure 1: Overview of a DocumentRAG pipeline, illustrating the flow from document ingestion and embedding to query-time retrieval and answer generation (Oche et al., 2025)(Oche et al., 2025)

During the ingestion phase, large documents are first segmented into smaller, self-contained passages—commonly referred to as *chunks*—which serve as the atomic retrieval units of the system. Each chunk is then embedded into a dense, high-dimensional vector representation that encodes its semantic content using a transformer-based encoder, and the resulting embeddings are stored in a vector index that supports efficient nearest-neighbour lookup at query time. (Oche et al., 2025) The chunk size is typically tuned to balance context completeness

against retrieval specificity: chunks must be large enough to contain useful information, yet small enough to match queries narrowly and fit within the generative model's context window. (Sarmah et al., 2024)

During the inference phase, the user query x is encoded into the same embedding space by a dedicated query encoder. Modern embedding-based retrievers such as Dense Passage Retrieval employ a bi-encoder architecture, in which a query encoder E_q and a passage encoder E_p independently map their respective inputs to the same d-dimensional vector space. The relevance of a passage d_i with respect to query x is then assessed via the dot product of their respective embeddings: (Oche et al., 2025)

$$\text{sim}(x, d_i) = E_q(x)^\top E_p(d_i) \tag{1}$$

The top-K passages $Z = \{z_1, \ldots, z_K\}$ with the highest similarity scores are retrieved from the index, typically accelerated by approximate nearest-neighbour structures such as FAISS, which enables efficient search over corpora of millions of passages in the order of milliseconds. (Oche et al., 2025) The retrieved passages are subsequently provided alongside the original query to a generative language model. Formally, the system models the output distribution by marginalising over the retrieved passages, weighting each candidate answer by the probability that the corresponding passage was correctly retrieved: (Oche et al., 2025)

$$P(y \mid x) = \sum_{i=1}^{K} P_\theta(y \mid x, z_i)\, P_\phi(z_i \mid x) \tag{2}$$

where $P_\phi(z_i \mid x)$ is the retrieval probability of passage z_i given the query, derived from the embedding similarity scores, and $P_\theta(y \mid x, z_i)$ is the conditional probability assigned by the generator to producing response y given both the query and the retrieved passage. The generator learns to attend to and synthesise the retrieved evidence into a fluent, contextually grounded answer, while the knowledge base can be updated independently of the generative model: modifications to the document corpus require only a re-indexing of the affected passages without retraining the underlying LLM.

4.3 Common issues in RAG systems

RAG systems have shown significant improvements in answer accuracy and reduction in hallucination within narrow scopes or domains. (Oche et al., 2025) They however suffer when a question requires broader but precise information across multiple domains, since the

retrieval system typically surfaces the most relevant but not **all** relevant information. The quality of retrieved documents significantly impacts the accuracy of RAG-generated answers, with high recall and precision being critical since poor retrieval leads directly to incorrect or irrelevant answers. This is especially problematic when the question requires nuanced information that may appear less relevant during retrieval but is actually crucial for the answer. Existing retrieval systems are often constrained by the retrieval mechanism, which may fail to provide adequate granularity for complex queries or may introduce irrelevant information, commonly referred to as hard negatives, that can mislead the LLM during the generation process. (Yu & McQuade, 2025)

Another issue is that retrieved passages are loose chunks of information which are not necessarily well structured, and their relations to one another may require further context or domain knowledge to interpret. This also makes it difficult to understand how the LLM might draw conclusions and connections from the different retrieved passages. This can partly be mitigated by including metadata about the chunks to provide context about where they reside in the larger knowledge base. Additionally, the retrieval step can be transformed into an agentic task, where the LLM itself uses the knowledge base to issue search queries and retrieve information iteratively until it has accumulated sufficient information to answer the question. A survey on agentic RAG synthesised emerging patterns including reflection, planning, and tool use, arguing that autonomous agents can orchestrate multi-hop retrieval more effectively than static pipelines. (Oche et al., 2025) While this shows significant improvement over a single retrieval step, it still suffers from the absence of explicit connections between retrieved passages and the lack of a global view over the retrieved information.

4.4 KG retrieval

(Sarmah et al., 2024) KGs are a structured representation of real-world entities, their attributes, and their relations, usually stored in a graph database or a triplet store. Entries are represented as triples consisting of a subject, a predicate, and an object. This allows capturing not only the information itself but also the relations between different pieces of information. Formally, a homogeneous KG can be defined as $G = (E, R)$, where E and R represent the sets of entities and relationships, respectively. A triple t can be represented as $t = (e, r, e')$, where e is the subject entity, r the predicate relationship, and e' the object entity. (Sanmartin, 2024) This structure can be especially beneficial for RAG systems as it enables the retrieval of not only relevant information but also the connections between them, allowing a QA system to develop an understanding of the interrelations between information across all domains covered by the knowledge graph.

KG retrieval can be broadly categorized into passive and active retrieval.

4.4.1 Passive retrieval

Passive retrieval is the process of retrieving relevant triples from a KG based on a given query without involving the LLM in the retrieval logic itself. This can be done using various methods such as graph traversal, subgraph matching, or embedding-based retrieval. In GraphRAG, the query is used to search the KG to retrieve relevant nodes (entities) and edges (relationships). A subgraph consisting of these relevant nodes and edges is extracted from the full KG to provide context, which is then integrated with the language model's internal knowledge to generate responses informed by both the structured information from the KG and its pre-trained knowledge. (Sarmah et al., 2024)

4.4.2 Active retrieval

During active retrieval, the reasoning and code-generating capabilities of an LLM are used to guide the retrieval process rather than relying on predefined queries. The Chain of Explorations (CoE) approach strategically traverses the KG by navigating through nodes and edges that are directly relevant to the query, performing a step-by-step exploration consisting of planning, KG lookups, and evaluation. The evaluation phase checks the current state of the traversal against the initial query, deciding whether to continue exploring, refine the exploration strategy, or synthesize an answer based on the gathered context. (Sanmartin, 2024) This allows such systems to adapt to vast and complex KGs and permits the extension or replacement of the underlying graph without the need to adapt the retrieval system. Similarly, the KRAGEN framework combines a Graph of Thoughts decomposition with a RAG system, where the LLM develops a graph structure plan to answer a question by decomposing it into subproblems, and uses the RAG system to search the knowledge base for each subproblem requiring data. (Matsumoto et al., 2024) However, active retrieval is limited by the LLM's ability to generate correct queries in terms of grammar and relevance, as well as the available computational resources.

4.5 KG retrieval vs DocumentRAG

KG retrieval has several benefits over DocumentRAG. First, it captures the connections between different pieces of information, which can be crucial for answering complex questions and for the traceability of answers. Second, it retrieves not only relevant information but also the relations between them, which can provide a deeper understanding of domain-specific interactions between retrieved entities. The primary advantage of KGs lies in their ability to offer a structured representation, which facilitates efficient querying and reasoning. (Sarmah

et al., 2024) Third, it inherently reduces noise in the retrieved information, as KGs are usually curated, structured, and information-focused, while DocumentRAG typically surfaces loose chunks of unstructured prose. The transition from unstructured dense text representations to a dynamic, structured knowledge representation via KGs can significantly reduce the occurrence of hallucinations, as it ensures that agents rely on explicit information rather than generating responses based on knowledge stored implicitly in their weights. (Sanmartin, 2024) Experimentally, the hallucination rate for a KG-based RAG pipeline was found to be 15%, compared to 30% for embedding-based RAG, suggesting that KG-based retrieval is more adept at adhering to factual content. (Sanmartin, 2024)

However, there are also notable downsides to KG retrieval. (Sarmah et al., 2024) First, it requires the existence of a curated and structured KG, and building and maintaining KGs while integrating data from different sources into a coherent KG poses significant challenges. This process has until now been mostly manual or at least heavily human-supervised, which is a costly and time-consuming process. Second, it requires an LLM capable of generating well-formed queries in the specific query language of the KG's database system, which can be a significant challenge especially for non-standard query languages or dialects. Third, while KGs excel at capturing connections between information, a query usually requires some form of entry point into the graph, i.e., a known entity. This can be problematic if an exact entity identifier is not known at query time, necessitating looser queries that retrieve more information in the hope of finding a relevant entry point. GraphRAG generally underperforms in abstractive QA tasks or when there is no explicit entity mentioned in the question. (Sarmah et al., 2024)

If a task requires the retrieval of specific, non-connected information, DocumentRAG may be more suitable as it can surface specific chunks without the need for an entry point into a graph. However, if the task requires reasoning over the connections and domain structure of the data, using a KG can significantly improve the performance of a QA system.

(Sarmah et al., 2024) Since both approaches have their respective strengths, a hybrid approach is also viable. HybridRAG integrates DocumentRAG and GraphRAG by combining the broad similarity-based retrieval of DocumentRAG with the structured, relationship-rich contextual data from GraphRAG. In a comparative evaluation, HybridRAG outperformed both DocumentRAG and GraphRAG individually in key metrics such as faithfulness (0.96) and answer relevancy (0.96), while maintaining high context recall. GraphRAG alone performs better in extractive questions, while DocumentRAG excels in abstractive questions; the hybrid approach compensates for the weaknesses of each by falling back to the other when one fails to retrieve correct context. Such a hybrid system could also use the KG to store metadata on how different chunks and documents in DocumentRAG are interconnected, thereby mitigating the lack of context inherent in purely DocumentRAG. (Xiao et al., 2024)

4.6 Summary

This chapter has examined the role of KGs in RAG systems. RAG addresses the hallucination problem of LLMs by grounding generation in externally retrieved evidence, but conventional DocumentRAG is limited by its inability to recover all relevant information and its lack of explicit connections between retrieved passages. KG-based retrieval mitigates these limitations by structuring information as triples that capture both entities and their relations, enabling retrieval of not only relevant facts but also the connections between them. Two retrieval paradigms were distinguished: passive retrieval, which extracts subgraphs matching a query, and active retrieval, which leverages LLM reasoning to iteratively traverse the graph. While KG retrieval reduces hallucination rates and facilitates more traceable answers, it requires curated KGs and may underperform when queries lack explicit entity references. Hybrid approaches that combine DocumentRAG and graph-based retrieval offer a balanced alternative, achieving superior performance across both extractive and abstractive QA tasks.

5 Research Plan

5.1 Models to finetune

(Team et al., 2025) All three models evaluated in this work are derived from the same model lineage: `google/t5gemma-2-4b-4b` is directly adapted from the pretrained `google/gemma-3-4b` checkpoint, sharing the same weights and training data. This tight coupling means the architectural difference between decoder-only and encoder-decoder is the primary variable under study, providing a uniquely controlled basis for comparison.

5.1.1 Baseline

`google/gemma-3-4b`

Gemma 3 4B is a multimodal decoder-only transformer with approximately 3.2 billion non-embedding parameters, using Grouped-Query Attention with QK-norm, interleaved local and global self-attention layers in a 5:1 ratio, and a 128K token context window, trained with knowledge distillation from a stronger teacher model. (Team et al., 2025)

5.1.2 Finetune

`google/gemma-3-4b`

The same Gemma 3 4B checkpoint is used as the starting point for supervised finetuning on the RDF triple extraction task, establishing a decoder-only finetuning baseline against which the encoder-decoder architecture can be directly compared. (Team et al., 2025)

5.1.3 Encoder-Decoder Finetune

`google/t5gemma-2-4b-4b`

T5Gemma 2 4B-4B is an encoder-decoder model adapted directly from the Gemma 3 4B pretrained checkpoint via the UL2 objective, introducing tied word embeddings across encoder and decoder and a merged attention mechanism that unifies decoder self-attention and cross-attention into a single module, resulting in approximately 7.5 billion total parameters with a 4B encoder and 4B decoder sharing the same underlying Gemma 3 architecture. (B. Zhang, Suganthan, et al., 2025)

5.2 Finetuning Plan

Finetuning is performed using Low-Rank Adaptation (LoRA), a parameter-efficient method that keeps the pretrained weight matrices frozen and instead introduces two low-rank trainable matrices per module, thereby greatly reducing memory usage while still allowing every module in the network to be adapted through the low-rank update. (Linder, 2026)

In addition to finetuning the existing parameters, the model vocabulary is extended with new special tokens that serve as structural markers in the output sequence. The idea of adding new tokens with trainable embeddings to a generative language model and then learning the associated representations has been explored in prior work on planning tokens, where such tokens guide the model's generation process with negligible additional learnable parameters (less than 0.001% of the total), integrating well with parameter-efficient finetuning methods like LoRA. (Wang et al., 2024)

Concretely, four new tokens are introduced to delimit the components of each extracted RDF triple: `<triple_start>`, `<predicate_marker>`, `<object_marker>`, and `<triple_end>`. The output format for a single triple is:

```
<triple_start> subject <predicate_marker> predicate
    <object_marker> object <triple_end>
```

This token scheme draws on the semantics of the RDF Turtle serialization format, where each statement consists of a subject, a predicate, and an object. In Turtle, when multiple statements share the same subject, predicates and objects can be listed after the subject without repeating it, enabling a compact representation. (McBride, 2004) Analogously, repeated predicates and or objects can be compressed within the marker-based format:

```
<triple_start> subject <predicate_marker> predicate
    <object_marker> object1 <object_marker> object2 <triple_end>
<triple_start> subject <predicate_marker> predicate1
    <object_marker> object1 <predicate_marker> predicate2
    <object_marker> object2 <triple_end>
```

Introducing new tokens to a pretrained model poses a particular challenge: the new tokens are added as special tokens to the existing tokenizer, and their embeddings and corresponding output head rows must be initialized and trained from scratch. This is handled separately from the LoRA adapters: the embedding and output head layers are explicitly unfrozen and trained, while LoRA low-rank adapter matrices are applied in parallel to all transformer blocks. (Linder, 2026) This two-part approach is necessary because restricting adaptation to the embedding layers alone is not sufficient: the self-attention layers are pretrained to process particular token patterns, and without also adapting the transformer blocks the internal computation remains partially misaligned with the new tokenization scheme. LoRA is preferred over full finetuning for the transformer blocks because its low-rank constraint acts as an implicit regularizer, reducing overfitting and mitigating catastrophic forgetting of pretrained knowledge. (Linder, 2026) In a comparable vocabulary expansion setting, this combined strategy of unfrozen embeddings together with LoRA on the transformer blocks was shown to outperform all other configurations with a statistically significant margin, including models that underwent full finetuning of all parameters. (Linder, 2026)

5.3 Ontology Text Format

To ensure the model can learn to extract triples consistent with a given schema, the input must include a representation of the ontology which defines a set of allowed triple patterns. (B. Zhang & Soh, 2024) A good schema for the purpose of data extraction must be easily newly definable as well as easily be inferred from existing KGs, to support both: the construction of new KGs and the augmention of existing ones. The following chapter describes the proposed format and how it can be inferred from existing KGs. To better visualize the concept, consider

the following triples:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex:  <http://example.com/>
PREFIX cat: <http://example.com/Cat>
PREFIX pred: <http://example.com/pred#>
cat:cat1 a ex:Cat ;
    pred:name "Whiskers" ;
    pred:aged "5"^^xsd:integer .

cat:cat2 a ex:Cat ;
    pred:name "Felix"@de ;
    pred:name "Felice"@en ;
    pred:aged "3"^^xsd:integer ;
    pred:isFriendsWith ex:cat1 .
```

Graph 1: Example RDF graph

From observing the above triples, we can infer that the allowed triple patterns are:

```
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX ex:  <http://example.com/>
PREFIX cat: <http://example.com/Cat>
PREFIX pred: <http://example.com/pred#>
ex:Cat pred:name "xsd:string" .
ex:Cat pred:name "xsd:langString" .
ex:Cat pred:aged "xsd:integer" .
ex:Cat pred:isFriendsWith ex:Cat .
```

Graph 2: Inferred triple patterns

This meta abstracted schema can be infered from any existing knowledge graph using the SPARQL Protocol and RDF Query Language (SPARQL) query Query 1

```
SELECT DISTINCT ?st ?p ?ot ?ol WHERE {
    ?s a ?st .
    OPTIONAL {
        ?s ?p ?o .
        OPTIONAL { ?o a ?ot . }
        BIND(IF(isLiteral(?o), DATATYPE(?o), "") AS ?ol)
    }
}
```

Query 1: Extract the meta schema of an RDF graph

When executed against the example graph in Graph 1, the query returns the results shown in 5.3.

The query Query 1 extracts all unique combinations of subject types, predicates, and object types that exist in the graph, effectively summarizing the schema of the data. This meta schema can be further summarized by grouping by the subject type, predicate and a list of possible object types, effectively allowing inference of a schema from every existing knowledge graph which aids in the creation of training datasets and ensures output consistency from the model into existing knowledge graphs.

?st	?p	?ot	?ol
ex:Cat	ex:pred#name		xsd:string
ex:Cat	ex:pred#name		xsd:langString
ex:Cat	ex:pred#aged		xsd:integer
ex:Cat	ex:pred#isFriendsWith	ex:Cat	

Table 2: Result of running Query 1 against the example graph in Graph 1

Ontologies use Internationalized Resource Identifier (IRI)s as identifiers, alltough semantically well structured and self-describing, they are verbose and lead to redundant repetition of the same base IRI across every non literal term. (McBride, 2004) To mitigate this, inpiration is drawn from the Turtle serialization format, which allows for a compact representation with the introduction of a base IRI which abbreviates a common prefix of all IRIs in the same ontology. (McBride, 2004) As of now there currently does not exist a standard for representing the schema of an ontology, so the following format is proposed for the input which is based on JavaScript Object Notation (JSON) to draw upon the LLMs existing pretraining (Huang et al., 2025).

```
{
    "prefixes": {
        "ex": "http://example.com/",
        "pred": "http://example.com/pred#",
    },
    "ex:Cat": {
        "pred:name": ["xsd:string", "xsd:langString"],
        "pred:aged": "xsd:integer",
        "pred:isFriendsWith": "ex:Cat"
    }
}
```

Listing 2: Proposed JSON format for representing the ontology schema

5.4 Dataset

Using the JSON described in 2 a training dataset can be constructed from any existing textular data to KG mapping or dataset by running the process described in 5.3 over the existing KG. Additionally parts of the dervied ontology can be dropped to create new subsets of the same ontology, increasing the diversity and size of available training samples.

The following subsections survey publicly available datasets that pair natural language text with some form of semi-structured output — RDF triples, relation tuples, or structured tables. These datasets represent the primary resources used to train and benchmark models for structured information extraction in the literature reviewed for this work.

5.4.1 WebNLG

WebNLG is a domain-agnostic benchmark dataset sourced from DBpedia, originally designed for the task of data-to-text generation from RDF triples, but widely adopted bidirectionally for text-to-triple extraction. The 2020 version covers 411 relation classes and pairs sets of DBpedia RDF triples with corresponding natural language verbalisations composed by human annotators, yielding a direct and exhaustive correspondence between text and triples. (Gajo & Barrón-Cedeño, 2025) Because annotators were instructed to compose text solely from the given triples, each sentence contains exactly the information encoded in the corresponding triple set, making it one of the cleanest benchmarks for closed-schema triple extraction.

5.4.2 T-REx

T-REx (Elsahar et al., 2018) is a large-scale alignment dataset that links Wikipedia abstracts to Wikidata and DBpedia relation triples via distant supervision, covering 633 distinct relation types. (Gajo & Barrón-Cedeño, 2025) By aligning natural language sentences to their corresponding Wikidata facts, T-REx provides broad coverage but also inherits the noise and incompleteness characteristic of distant-supervision methods: reference triple sets may be non-exhaustive relative to the text.

5.4.3 REBEL

REBEL is a large-scale distantly supervised dataset constructed from Wikipedia and Wikidata, designed to support end-to-end relation extraction into RDF-like triples. (B. Zhang & Soh, 2024) Like T-REx it suffers from incomplete reference annotations — valid triples extractable from a sentence may not be present in the reference set if the corresponding Wikidata fact was not used during alignment — which can lead to overly pessimistic evaluation of models that correctly extract additional facts.

5.4.4 Wiki-NRE

Wiki-NRE is a Wikipedia-based named relation extraction dataset also constructed via distant supervision over Wikidata. It provides a moderately-sized closed schema and has been used as a benchmark for schema-constrained triple extraction. Because the reference triplets in this dataset are non-exhaustive due to the distant supervision construction methodology, methods that generate semantically correct but unlisted triples are penalised, which complicates faithful evaluation. (B. Zhang & Soh, 2024)

5.4.5 KELM

KELM (Agarwal et al., 2021) is a knowledge-enhanced language model pretraining corpus derived from Wikidata and Wikipedia, covering approximately 1500 relation classes. It pairs sentences with corresponding Wikidata triples at a large scale, making it suitable for schema-understanding pretraining for information extraction models. (Z. Li et al., 2024)

5.4.6 NYT

The NYT dataset (Riedel et al., 2010) consists of New York Times news articles annotated with relation triples via distant supervision against Freebase, covering 24 relation types. (Gajo & Barrón-Cedeño, 2025) Its general-domain journalistic text and moderate schema size make it a standard benchmark for distantly supervised relation extraction.

5.4.7 ADE

ADE (Gurulingappa et al., 2012) is a medical corpus annotated with relations linking drugs and dosages to adverse effects. The dataset contains 2562 training and 300 test documents, with a single relation type (*adverseEffect*) between drug and disease entities. (Gajo & Barrón-Cedeño, 2025) Its narrow single-relation schema and specialised biomedical terminology make it a focused benchmark for domain-specific structured extraction.

5.4.8 CoNLL04

CoNLL04 (Roth & Yih, 2004) is a corpus for joint entity and relation extraction comprising news articles with span-level annotations for three entity types (Organisation, Person, Location) and five relation types (kill, locatedIn, workFor, orgBasedIn, liveIn), totalling 922 training and 288 test documents. (Gajo & Barrón-Cedeño, 2025)

5.4.9 SciERC

SciERC (Luan et al., 2018) is a multi-task corpus in the domain of scientific literature used for entity recognition, relation classification, and coreference. It covers six entity types and seven relation types across 1366 training and 397 test documents, and is considered one of the more challenging benchmarks due to specialised technical language and a high number of semantically subtle relation types. (Gajo & Barrón-Cedeño, 2025)

5.4.10 FewREL

FewREL (Han et al., 2018) is a large-scale few-shot relation classification dataset sourced from Wikipedia and Wikidata, containing 100 relation types with 700 annotated instances per type. (Gajo & Barrón-Cedeño, 2025) It is specifically designed to evaluate model generalisation to unseen relation classes, but for the present work it is less suitable than explicit extraction datasets because the target is a relation label rather than a fully structured output representation.

5.4.11 DocRED

DocRED (Yao et al., 2019) is a document-level relation extraction dataset derived from Wikipedia and Wikidata, covering 96 relation types. Unlike sentence-level datasets, DocRED requires models to extract relations between entities that may span multiple sentences within a document, posing an additional challenge for structured extraction. (Gajo & Barrón-Cedeño, 2025) However, because the dataset provides explicit labeled relation annotations, these labels can be reused to validate outputs after mapping them to the target ontology.

5.4.12 WikiBio

WikiBio pairs Wikipedia biography pages with their corresponding structured infobox data, representing a text-to-table extraction task where the target output is a set of key-value pairs describing a person. (Huang et al., 2025) It has been used to benchmark text structuralisation models that convert unstructured biographical text into a structured record.

5.4.13 RotoWire

RotoWire (Wiseman et al., 2017) contains instances from the sports domain where each sample consists of a natural language game report paired with two structured tables recording the scores and statistics of teams and players. (Huang et al., 2025) It is used to benchmark models that must extract and align multiple structured records from a single narrative text.

Dataset	Domain	Schema Size	Original Format	Strategy
WebNLG	General (DBpedia)	411 relations	plaintext → RDF (Turtle)	use
T-REx	General (Wikidata)	633 relations	plaintext → RDF	clean
REBEL	General (Wikidata)	large	plaintext → RDF	clean
Wiki-NRE	General (Wikidata)	moderate	plaintext → tuples	clean
KELM	General (Wikidata)	~1500 rel.	plaintext → RDF	use
NYT	News	24 relations	plaintext → Freebase	map
ADE	Biomedical	1 relation	plaintext → spans (JSON)	adapt
CoNLL04	News	5 relations	plaintext → spans (JSON)	adapt
SciERC	Scientific lit.	7 relations	plaintext → graph (JSON)	adapt
FewREL	General (Wikipedia)	100 classes	plaintext → label	drop
DocRED	General (Wikipedia)	96 relations	document → tuples (JSON)	map
WikiBio	Biographies	varies	plaintext → key-value	adapt
RotoWire	Sports	fixed stats	plaintext → tables	adapt

Table 3: Summary of datasets for structured information extraction, including domain, schema size, original data format, and transformation strategy. Strategy: *use* = used as-is; *clean* = noisy samples dropped; *map* = existing schema mapped to RDF triples; *adapt* = data adapted to a purpose-built ontology; *drop* = excluded because validation or adaptation would require multiple unreliable steps.

Glossary

BM25 Best Matching 25. 15

IE Information Extraction. 7, 15

IRI Internationalized Resource Identifier. 25

JSON JavaScript Object Notation. 25

KG Knowledge Graph. 2, 15, 18, 19, 20, 21, 23, 25

LLM Large Language Model. 4, 6, 7, 8, 9, 10, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 25

LoRA Low-Rank Adaptation. 12

NLP Natural Language Processing. 4, 7

PEFT Parameter Efficient Finetuning. 12

QA Question Answering. 15, 18, 20, 21

QLoRA Quantized Low-Rank Adaptation. 12, 15

RAG Retrieval-Augmented Generation. 2, 15, 16, 17, 18, 19, 20, 21

SPARQL SPARQL Protocol and RDF Query Language. 24

TF-IDF Term Frequency-Inverse Document Frequency. 15

References

Alt, C., H"ubner, M., & Hennig, L. (2019). Fine-tuning pre-trained transformer language models to distantly supervised relation extraction. In A. Korhonen, D. Traum, & L. Màrquez (Eds.), *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 1388–1398). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1134

Barbero, F., Arroyo, Á., Gu, X., Perivolaropoulos, C., Bronstein, M., Veličković, P., & Pascanu, R. (2025). Why do llms attend to the first token? https://doi.org/10.48550/arXiv.2504.02732

Brach, W., Košťál, K., & Ries, M. (2025). The effectiveness of large language models in transforming unstructured text to standardized formats. https://doi.org/10.48550/arXiv.2503.02650

Carta, S., Giuliani, A., Piano, L., Podda, A. S., Pompianu, L., & Tiddia, S. G. (2023). Iterative zero-shot llm prompting for knowledge graph construction. https://doi.org/10.48550/arXiv.2307.01128

Caufield, J. H., Hegde, H., Emonet, V., Harris, N. L., Joachimiak, M. P., Matentzoglu, N., Kim, H., Moxon, S., Reese, J. T., Haendel, M. A., Robinson, P. N., & Mungall, C. J. (2024). Structured prompt interrogation and recursive extraction of semantics (spires): A method for populating knowledge bases using zero-shot learning (J. Wren, Ed.). *Bioinformatics*, *40*(3). https://doi.org/10.1093/bioinformatics/btae104

Dagdelen, J., Dunn, A., Lee, S., Walker, N., Rosen, A. S., Ceder, G., Persson, K. A., & Jain, A. (2024a). Structured information extraction from scientific text with large language models. *Nature Communications*, *15*(1). https://doi.org/10.1038/s41467-024-45563-x

Dagdelen, J., Dunn, A., Lee, S., Walker, N., Rosen, A. S., Ceder, G., Persson, K. A., & Jain, A. (2024b). Structured information extraction from scientific text with large language models. *Nature Communications*, *15*(1). https://doi.org/10.1038/s41467-024-45563-x

Dunn, A., Dagdelen, J., Walker, N., Lee, S., Rosen, A. S., Ceder, G., Persson, K., & Jain, A. (2022). Structured information extraction from complex scientific text with fine-tuned large language models. https://doi.org/10.48550/arXiv.2212.05238

Gajo, P., & Barrón-Cedeño, A. (2025). Natural vs programming language in llm knowledge graph construction. *Information Processing & Management*, *62*(5), 104195. https://doi.org/10.1016/j.ipm.2025.104195

Ghanem, H., & Cruz, C. (2025). Fine-tuning or prompting on llms: Evaluating knowledge graph construction task. *Frontiers in Big Data*, *8*. https://doi.org/10.3389/fdata.2025.1505877

Hu, R., Yang, Y., Liu, S., Li, Z., Liu, J., Ding, X., Sun, H., & Ren, L. (2025). Large language model driven transferable key information extraction mechanism for nonstandardized tables. *Scientific Reports*, *15*(1). https://doi.org/10.1038/s41598-025-15627-z

Huang, J. Y., Zhou, W., Xu, N., Wang, F., Liu, Q., Zhang, S., Poon, H., & Chen, M. (2025). Omnistruct: Universal text-to-structure generation across diverse schemas. https://doi.org/10.48550/arXiv.2511.18335

Li, B., Jiang, G., Li, N., & Song, C. (2024). Research on large-scale structured and unstructured data processing based on large language model. *Proceedings of the International Conference on Machine Learning, Pattern Recognition and Automation Engineering*, 111–116. https://doi.org/10.1145/3696687.3696707

Li, Z., Zeng, Y., Zuo, Y., Ren, W., Liu, W., Su, M., Guo, Y., Liu, Y., Li, X., Hu, Z., Bai, L., Li, W., Liu, Y., Yang, P., Jin, X., Guo, J., & Cheng, X. (2024). Knowcoder: Coding structured knowledge into llms for universal information extraction. https://doi.org/10.48550/arXiv.2403.07969

Linder, M. R. (2026). Vocabulary expansion of large language models via kullback-leibler-based self-distillation. https://doi.org/10.48550/arXiv.2508.15807

Matsumoto, N., Moran, J., Choi, H., Hernandez, M. E., Venkatesan, M., Wang, P., & Moore, J. H. (2024). Kragen: A knowledge graph-enhanced rag framework for biomedical problem solving using large language models (C. Kendziorski, Ed.). *Bioinformatics*, *40*(6). https://doi.org/10.1093/bioinformatics/btae353

McBride, B. (2004). The resource description framework (rdf) and its vocabulary description language rdfs. In S. Staab & R. Studer (Eds.), *Handbook on ontologies* (pp. 51–65). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-24750-0_3

Ni, X., Li, P., & Li, H. (2023). Unified text structuralization with instruction-tuned language models. https://doi.org/10.48550/arXiv.2303.14956

Norouzi, R., Kleinberg, B., Vermunt, J. K., & van Lissa, C. J. (2025). Capturing causal claims: A fine-tuned text mining model for extracting causal sentences from social science papers. *Research Synthesis Methods*, *16*(1), 139–156. https://doi.org/10.1017/rsm.2024.13

Oche, A. J., Folashade, A. G., Ghosal, T., & Biswas, A. (2025). A systematic review of key retrieval-augmented generation (rag) systems: Progress, gaps, and future directions. https://doi.org/10.48550/arXiv.2507.18910

Sanmartin, D. (2024). Kg-rag: Bridging the gap between knowledge and creativity. https://doi.org/10.48550/arXiv.2405.12035

Sarmah, B., Mehta, D., Hall, B., Rao, R., Patel, S., & Pasquali, S. (2024). Hybridrag: Integrating knowledge graphs and vector retrieval augmented generation for efficient information extraction. *Proceedings of the 5th ACM International Conference on AI in Finance*, 608–616. https://doi.org/10.1145/3677052.3698671

Susnjak, T., Hwang, P., Reyes, N. H., Barczak, A. L. C., McIntosh, T. R., & Ranathunga, S. (2024). Automating research synthesis with domain-specific large language model fine-tuning. https://doi.org/https://doi.org/10.1145/3715964

Team, G., Kamath, A., Ferret, J., Pathak, S., Vieillard, N., Merhej, R., Perrin, S., Matejovicova, T., Ramé, A., Rivière, M., Rouillard, L., Mesnard, T., Cideron, G., Grill, J.-b., Ramos, S., Yvinec, E., Casbon, M., Pot, E., Penchev, I., . . . Hussenot, L. (2025). Gemma 3 technical report. https://doi.org/10.48550/arXiv.2503.19786

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). Attention is all you need. https://doi.org/10.48550/arXiv.1706.03762

Wang, X., Caccia, L., Ostapenko, O., Yuan, X., Wang, W. Y., & Sordoni, A. (2024). Guiding language model reasoning with planning tokens. https://doi.org/10.48550/arXiv.2310.05707

Xiao, W., Liu, Y., Li, X., Gao, F., & Gu, J. (2024). Tkg-rag: A retrieval-augmented generation framework with text-chunk knowledge graph. *2024 25th International Arab Conference on Information Technology (ACIT)*, 1–9. https://doi.org/10.1109/acit62805.2024.10877117

Yu, H. Q., & McQuade, F. (2025). Rag-kg-il: A multi-agent hybrid framework for reducing hallucinations and enhancing llm reasoning through rag and incremental knowledge graph learning integration. https://doi.org/10.48550/arXiv.2503.13514

Zhang, B., Moiseev, F., Ainslie, J., Suganthan, P., Ma, M., Bhupatiraju, S., Lebron, F., Firat, O., Joulin, A., & Dong, Z. (2025). Encoder-decoder gemma: Improving the quality-efficiency trade-off via adaptation. https://doi.org/10.48550/arXiv.2504.06225

Zhang, B., Suganthan, P., Liu, G., Philippov, I., Dua, S., Hora, B., Black, K., Martins, G., Sanseviero, O., Pathak, S., Hardin, C., Visin, F., Zhang, J., Kenealy, K., Yin, Q., Song, X., Lacombe, O., Joulin, A., Warkentin, T., & Roberts, A. (2025). T5gemma 2: Seeing, reading, and understanding longer. https://doi.org/10.48550/arXiv.2512.14856

Zhang, B., & Soh, H. (2024). Extract, define, canonicalize: An llm-based framework for knowledge graph construction. https://doi.org/10.48550/arXiv.2404.03868

Zhang, W., Wang, Q., Kong, X., Xiong, J., Ni, S., Cao, D., Niu, B., Chen, M., Li, Y., Zhang, R., Wang, Y., Zhang, L., Li, X., Xiong, Z., Shi, Q., Huang, Z., Fu, Z., & Zheng, M. (2024). Fine-tuning large language models for chemical text mining. *Chem. Sci.*, *15*, 10600–10611. https://doi.org/10.1039/D4SC00924J

Zhuang, A., Zhang, G., Zheng, T., Du, X., Wang, J., Ren, W., Huang, S. W., Fu, J., Yue, X., & Chen, W. (2024). Structlm: Towards building generalist models for structured knowledge grounding. https://doi.org/10.48550/arXiv.2402.16671