From Unstructured Procedural Text to Knowledge Graphs: Comparing LLM-based Extraction Strategies

Erik Sörqvist, Kenneth Obinna, Clara Bersch, Adam Altenbuchner and Jörg Krüger
Industrial Automation Technology, Technical University Berlin, Germany

soerqvist@tu-berlin.de
clara.bersch@tu-berlin.de
k.kara@campus.tu-berlin.de
a.altenbuchner@tu-berlin.de
joerg.krueger@tu-berlin.de

Abstract: As experienced workers retire across industrialized economies, organizations risk losing procedural expertise that often remains tacit, undocumented or scattered across unstructured documents. Translating this knowledge into structured, machine-readable representations is difficult to scale, labor-intensive, and prone to inconsistency when done manually. This paper addresses the automated construction of knowledge graphs from natural language procedural descriptions, developing a generic approach for transforming unstructured expert knowledge into structured knowledge graphs that support downstream retrieval and question-answering applications. Three text-to-knowledge-graph approaches were designed, implemented, and systematically evaluated. The first employed a large general-purpose language model (Qwen3-32B) with a single-stage zero-shot extraction prompt, the second applied the same strategy using a smaller base model (Llama2-13B), and the third combined supervised fine-tuning of the smaller model on synthetic extraction data with a decomposed extraction architecture targeting one to two ontology elements per phase. All approaches were evaluated across six procedural descriptions spanning multiple technical domains, with chunk size, model temperature, and ontology detail as configuration parameters. Results were assessed for intrinsic quality and extrinsic fitness for use, measured by question-answering accuracy in a Graph-RAG application. The results demonstrate that extraction strategy is a more decisive factor than model capacity. The fine-tuned model achieved a Question Answering (QA) pass rate of 55.3%, compared to 46.8% for the large model and 33% for the small base model, outperforming the general-purpose model on both intrinsic quality metrics and downstream performance. Average node degree, duplication rate, and ontology simplicity emerge as the strongest predictors of retrieval performance. Shorter ontologies consistently outperformed richer ones, suggesting that ontology design should be driven by the intended downstream application rather than semantic completeness. Over-extraction of procedural steps did not degrade performance but instead improved retrieval robustness by providing additional entry points for graph traversal. The findings offer practical guidance for designing scalable, locally deployable knowledge graph construction pipelines for procedural texts under computational and confidentiality constraints.

Keywords: Knowledge graphs, Large language models, Ontology, Procedural knowledge

1. Introduction

Demographic shifts are accelerating the loss of experienced workers across industrialized economies. In Germany alone, nearly 20 million baby boomers will retire by 2036, while only 12.5 million younger workers enter the workforce to replace them (Deschermeier and Schäfer, 2024). With each departure, organizations risk losing the operational expertise these workers have accumulated over years of practice, including established procedures, problem-solving strategies and process understanding that constitute much of an organization's institutional memory. A large part of this expertise is procedural knowledge, meaning knowledge of how to carry out defined tasks such as maintenance, assembly, or repair processes. This knowledge frequently remains tacit or scattered across unstructured natural language documents, making it inaccessible to automated systems and difficult to transfer reliably to non-experts. Consequently, there is a growing need for automated approaches that can transform unstructured procedural knowledge into structured, machine-readable representations. Knowledge graphs, structured networks of entities and their relationships, have become a pivotal technology for information retrieval, biomedical research and education, because they make the connections between concepts explicit and traversable (Hogan *et al.*, 2022). This property makes them particularly well-suited for encoding procedural knowledge, for which preserving the sequential and dependency relations between steps, tools, and conditions is essential. Unlike vector-based retrieval systems, which capture semantic similarity but discard relational structure, knowledge graphs retain these connections in queryable form. However, constructing knowledge graphs from procedural text remains often a manual process that is difficult to scale and prone to inconsistency. Recent advances in large language models have opened new possibilities for automating knowledge graph construction from text, as their zero-shot and few-shot capabilities allow information extraction without extensive domain-specific training data (Pan *et al.*, 2023). However, existing approaches have largely focused on general-purpose knowledge

extraction, while the systematic transformation of procedural text into ontology-conformant knowledge graphs remains underexplored. In particular, it is unclear how model capacity, extraction strategy, and ontology design interact to determine the quality of automatically generated procedural knowledge graphs. Further, most existing quality assessments rely on intrinsic structural metrics such as ontology conformance, connectivity, or completeness (Issa *et al.*, 2021; Seo *et al.*, 2022; R. Zhu *et al.*, 2023). Whether a knowledge graph that scores well on these metrics provides actual value in downstream applications, such as question answering for non-expert users, is rarely assessed (Zhu *et al.*, 2019). The disparity between structural quality and practical utility hinders the development of actionable design guidelines for automated knowledge graph construction pipelines. This paper addresses both gaps by designing, implementing and evaluating three LLM-based approaches for transforming unstructured procedural text into knowledge graphs conformant with the Procedural Knowledge Ontology. The approaches range from a single-stage zero-shot extraction using a large general-purpose model to a decomposed multi-stage pipeline built on a smaller model adapted through supervised fine-tuning on synthetic data. The most capable models are typically only available through cloud-based APIs, yet procedural knowledge in industrial settings is often confidential, favoring smaller models that can be deployed locally. Whether such models, when combined with task decomposition and fine-tuning, can match or exceed the extraction quality of larger cloud-hosted models is a central question of this work. All approaches are evaluated across six procedural descriptions from multiple domains, both through intrinsic quality metrics and through downstream question-answering accuracy in a Graph-RAG pipeline. The results show that extraction strategy is a more decisive factor than model capacity. The fine-tuned model outperforms the general-purpose model nearly three times its size on both evaluation dimensions. Average node degree, duplication rate, and ontology simplicity emerge as the strongest predictors of downstream retrieval performance. The rest of the paper is structured as follows: First, a general overview of related work that shapes the foundation of our research. Second, an overview of the methodology followed to reach the results. Third, results of implemented methodology. Fourth, a discussion on the relevance of the results. Finally, a summary, conclusion and directions for future work on this topic.

2. Related Work

2.1 Procedural Knowledge

(Sternberg, 2013) states that all knowledge can be divided into two categories: declarative knowledge, which is knowledge of facts, and procedural knowledge, which is knowledge of skill. (Surif, Ibrahim and Mokhtar, 2012; Curry *et al.*, 2025) describe procedural knowledge as ”how-to” knowledge or knowledge covering how to perform a certain task. (Eiriksdottir and Catrambone, 2011) add that procedural knowledge details the precise steps and methods required to complete a task, including the system states and the actions that change them. (Celino *et al.*, 2025) frame this in an industry context, defining procedural knowledge as the procedures employees must comply with to correctly perform tasks, with the aim of reducing risks and human errors. (Georgeff and Lansky, 1986) characterize procedural knowledge as sequences of actions for achieving goals, noting that representing this knowledge explicitly offers computational efficiency over reconstructing plans repeatedly from knowledge of atomic actions. (Carriero *et al.*, 2025; Sörqvist *et al.*, 2025) emphasize that procedural knowledge, frequently implicit or tacit, models a procedure as an ordered sequence of actions executed to achieve an outcome.

2.2 Procedural Knowledge Ontology

The ontology used to structure a knowledge graph determines how reliably machines can interpret and reason over the represented knowledge. (Carriero *et al.*, 2025) developed the Procedural Knowledge Ontology (PKO), a modular schema designed to represent procedural workflows in a structured, machine-readable format, following the Linked Open Terms (LOT) methodology (Poveda-Villalón *et al.*, 2022) for industry-domain ontologies. By expressing procedural knowledge in a formal semantic format, PKO ensures interoperability, enables reasoning and supports knowledge reuse, improving both compliance and employee training. A key innovation of PKO is its distinction between procedure definition and execution, an aspect not addressed by standards such as BPMN (Kurz, 2016). The ontology is modular and organized into a core section and industry sections. The PKO Core defines a procedure as a sequence of actions intended to achieve a specific outcome and models its execution using the ProcedureExecution and StepExecution classes, both of which are subclasses of the Activity class. It also captures unforeseen events using the IssueOccurrence class to model errors, causes, and solutions, and the UserQuestionOccurrence class to track questions encountered by the executing agent. As an extension of the PKO Core, PKO Industry was developed to address requirements that extend into the broader industrial domain. It primarily represents the physical and operational context of a

procedure by modeling entities such as machines, devices, energy sources, and equipment. This makes it especially effective for safety-related tasks and procedures.

2.3 Graph Generation Approaches

(Ren *et al*. 2023) propose a method for automatically constructing procedural graphs that captures not only sequential but also non-sequential dependency relations between steps, using syntactic and discourse features for node classification and edge prediction. While their approach relies on supervised classification over an annotated dataset, the approaches presented in this work use LLM-based extraction with ontology constraints, requiring no task-specific annotations. (Pan *et al.*, 2023) show that LLMs simplify key tasks in knowledge engineering, enabling knowledge graph construction at scale and with improved quality. A key advantage is that LLMs enable information extraction without extensive domain-specific training data through their zero-shot and few-shot capabilities (Y. Zhu *et al.*, 2023). This allows core subtasks of the text-to-knowledge-graph pipeline, such as entity recognition and relation extraction, to be streamlined through prompting rather than task-specific models. With LLM-based approaches, the quality of information extraction and graph creation is claimed to predominantly be a matter of model selection and prompt-engineering. Recent studies have begun to empirically compare these LLM-based extraction strategies. For example, Mo et al. (2025) proposed KGGen, a multi-stage pipeline that separates entity and relation extraction, cross-source aggregation, and entity clustering into distinct LLM-driven phases, demonstrating improved graph quality over single-pass extraction methods such as GraphRAG. While these works advance general-purpose knowledge extraction, they do not address ontology-conformant extraction from procedural text.

2.4 Research gap

The research community has made substantial progress in both defining procedural knowledge through ontological frameworks and automating knowledge graph construction using LLMs (Pan *et al.*, 2023). However, the intersection of these two fields, specifically the automated construction of ontology-conformant knowledge graphs from procedural text, remains underexplored. This work addresses this gap by comparing three LLM-based extraction approaches across six procedural texts, evaluating both intrinsic graph quality and downstream QA performance.

3. Methodology

Based on the findings of the literature review, this study aims to systematically evaluate the impact of model size, out-of-the-box vs. fine-tuned LLMs, and ontology distillation on knowledge graph quality. The overall methodology for assessing these parameters can be seen in Figure 1.

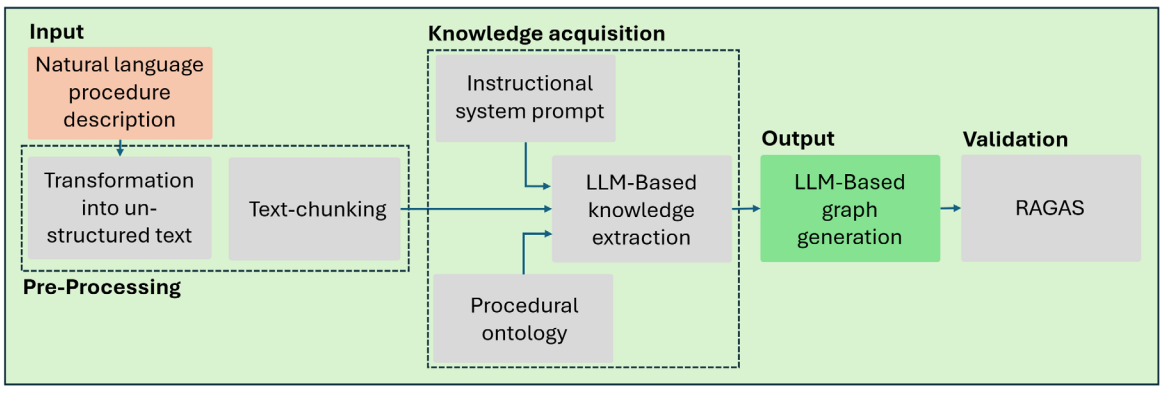

Figure 1: General pipeline for the three approaches addressed in this work.

It covers the entire systematic pipeline, from natural language procedure inputs and preprocessing to information extraction or knowledge acquisition, LLM-based graph generation, and validation. Following the text-to-graph workflow described by (Zhong *et al.*, 2023), the pipeline is structured into the subtasks of data preprocessing, knowledge acquisition, and knowledge refinement. Each subtask is implemented as a modular component, facilitating the transition from natural language instructions into a machine-readable network of entities and relationships.

3.1 Pre-processing: Transformation Into Unstructured Text and Text Chunking

To ensure the focus on semantic structure during Knowledge Acquisition instead of formatting cues, the decision was made to transform the originally structured texts of the procedures into unstructured texts by removing the step marker heading, such as ”Step 1:”. This also better approximates the characteristics of spoken instructions. Furthermore, chunk sizes of 5, 8, and 12 sentences were used to systematically examine the balance between local precision and contextual completeness in triplet extraction. This balance is especially important in procedural texts, where meaningful relationships can extend over varying lengths, sometimes even across longer paragraphs. By evaluating multiple levels of context granularity, we can observe how chunk size affects whether these semantically complete procedural relationships remain intact or become fragmented.

3.2 Knowledge Acquisition

3.2.1 Extraction approaches

To investigate the relative influence of model capacity and extraction strategy on Knowledge Graph quality, three approaches were designed that vary along these two dimensions. A *Large-Single-Stage* approach used Qwen3-32B to extract all ontology elements in a single pass per text chunk. A *Small-Single-Stage* approach applied the same strategy with the smaller Llama2-13B to isolate the effect of model size. A *Small-Multi-Stage* approach decomposed the extraction into separate phases, each targeting a subset of the ontology, using a fine-tuned Llama2-13B to test whether a carefully designed extraction pipeline can compensate for reduced model capacity. The *Large-Single-Stage* approach accessed Qwen3-32B via the Groq cloud inference API, with a context length of 32,768 tokens. The *Small-Single-Stage* approach loaded Llama2-13B-chat locally in 4-bit quantization, with a context length of 4,096 tokens. The *Small-Multi-Stage* approach used the same small base model but split the extraction into two phases using three separately fine-tuned models, each trained on a separate training subset with a different expected output so that it learned only the narrow extraction scope of its respective phase. In the first phase, one model extracted the central Procedure node and initial Step nodes with hasStep and nextStep relationships from the first text chunk, while a second model extracted further Step nodes with the same relationship types from all subsequent chunks. In the second phase, a third model received each text chunk together with the Steps previously extracted from it and generated Execution Instruction nodes that capture detailed guidance for each step.

3.2.2 Supervised fine-tuning

To move beyond general-purpose prompting and produce a model specialized in mapping procedural descriptions onto a formal ontology, supervised fine-tuning was applied to the Llama2-13B model. The training data was generated synthetically by prompting Qwen3-32B to produce 59 short, unstructured procedural descriptions spanning diverse tasks. For each description, the model also generated the expected extraction output, including Procedure, Step and Execution Instruction nodes with their corresponding relationships (hasStep, nextStep, advises). These outputs were formatted as JSONL and normalized into the prompt-response format required by Llama2. The data was then split into three subsets, all containing the same 59 procedural descriptions but with different expected outputs. The outputs in the first subset contain only the extracted Procedure and Step nodes, the second only the Step nodes, and the third contains the Step nodes as part of the system prompt and Execution Instruction nodes as part of the expected answer. For training, parameter-efficient fine-tuning (PEFT) with BitsAndBytes 4-bit quantization was used. The model was loaded in 4-bit precision with float32 compute dtype and gradient checkpointing enabled to reduce VRAM usage. Low-rank adapters were applied with rank r=16, alpha=32, and a dropout of 0.05. Training ran for 16 epochs with a batch size of one, gradient accumulation over eight steps, a learning rate of 5e-5 and fp16 mixed precision.

3.2.3 Prompting strategy

Each extraction pass was guided by a system prompt consisting of two components: an instructional part defining the extraction task and output format, and the target ontology specifying the entity classes and relationship types to extract. All prompts instructed the model to take on the role of an expert in Knowledge Graph construction for procedural tasks while discouraging it from using any prior knowledge. Output-format constraints required the model to respond only in JSON format without surrounding commentary, and imperative language was used to increase compliance for all critical constraints. Across all approaches, two prompt variants were used to handle multi-chunk documents. A first-chunk prompt instructed the model to identify the central Procedure node and connect it to the first extracted Step. For all subsequent chunks, an

after-chunk prompt provided the name of the previously extracted Procedure and the last Step node as carry-over context, with the explicit instruction to connect the first step of the current chunk to the last step of the previous chunk and not to create additional Procedure nodes. The key difference between the *single-stage* and *multi-stage* approaches lay in the scope of each prompt. In the *large* and *small-single-stage* approaches, each prompt contained the full or distilled ontology and instructed the model to extract all entity types and relationship types in a single pass. In the *Small-Multi-Stage* approach, each prompt targeted only a narrow subset of the ontology. The first-phase prompts were restricted to Procedure and Step nodes with hasStep and nextStep relationships. The second-phase prompt received the previously extracted Step nodes as part of the input and was instructed to extract only Execution Instruction nodes and their advised relationships.

3.2.4 Procedural ontology variants

To evaluate the trade-off between task-specific efficiency and level of detail, two distinct versions of the PKO by (Carriero et al., 2025) were used across the large and small-single-stage approaches. The full PKO included both the Core and Industry modules. The Core module modeled the specification and execution of a procedure, capturing complex relations such as action and tool requirements, step verifications, issue occurrences, and user questions. The industry module extended this with entities representing the physical and operational context, such as machines, energy sources, safety equipment and lockout-tagout procedures, broadening applicability to industrial domains. The goal of using the full ontology was to enable complex reasoning and interoperability across different industrial systems. The distilled variant focused on the core entities and relationships essential for procedural flows, reducing the input context length, and allowing the model to concentrate on the most retrieval-relevant subset of the ontology. The *Small-Multi-Stage* approach was not tested with different ontology variants, as its decomposed extraction architecture already targeted a minimal subset of the ontology in each phase.

3.3 LLM-based Knowledge Graph Generation

In the final step, we used the Neo4j Graph Builder module to automatically generate knowledge graphs from the JSON format triplets generated in the previous step.

3.4 Evaluation Strategy

The evaluation strategy consisted of intrinsic and extrinsic quality metrics evaluated using a set of natural language procedural knowledge texts.

3.4.1 Validation data

The source for validation was six publicly available "how-to" or DIY (do-it-yourself) procedural descriptions, which can be seen in Table 1. These procedures were chosen because they cover different domains, enabling the pipeline to be evaluated against various linguistic and structural challenges.

Table 1: Set of texts used as inputs for evaluation.

ID	Manual	Character	Word	Sentences
1	How to replace a laptop Battery	1964	255	21
2	How to replace a mobile phone Battery	14254	2623	164
3	How to replace a 3d printer nozzle	3122	512	34
4	How to change filament during printing	1661	335	11
5	How to build and assemble a Cat Treat Timer	2689	457	32
6	How to build a Pan-Tilt Head for DIY Motion Control	6793	1208	55

3.4.2 Intrinsic quality measures

Ontology conformance was evaluated using metrics that focus on correctness and graph structure rather than instantiation ratios. Specifically, the assessment considered class violations (when generated object classes are not part of the ontology), relationship violations (when generated relationships are not defined in the ontology), the number of disconnected subgraphs (which hinder graph traversal), and edge duplicates (redundant connections). Unlike instantiation ratios, where high values don't guarantee superior extraction, these metrics directly address the accuracy and coherence of the extracted knowledge, accounting for cases

where low instantiation may result from either the model's limitations or the absence of ontology concepts in the source text.

3.4.3 Extrinsic quality measures

To evaluate the extrinsic quality, or the "fitness for use" of the generated knowledge graphs we employed a Graph-RAG approach. The quality of any RAG-based application depends on the truthfulness of the generated response to the user's query. Evaluating RAG quality is challenging, as it involves multiple dimensions including retrieval quality, the LLM's ability to exploit retrieved information, and overall generation accuracy (Es *et al.*, 2023). To address this, (Es *et al.*, 2023) developed RAGAS (Retrieval Augmented Generation Assessment), a framework for evaluating RAG-based QA applications without requiring human ground-truth annotations. RAGAS assesses three quality dimensions: faithfulness, measuring whether the answer is grounded in the retrieved context; context relevance, measuring the absence of irrelevant information in the retrieved context; and answer relevance, measuring whether the answer addresses the posed question. For each of the six procedural source texts, 12 question-answer pairs were constructed as ground truth. Each generated knowledge graph was then used in a Graph-RAG pipeline to answer all questions, and an LLM-as-a-judge evaluated whether the generated answers matched the ground truth using a binary pass/fail judgment.

> Consider the response correct if it:
>
> (1.) Contains the key information from the expected answer
>
> (2.) Is factually accurate based on the provided context
>
> (3.) Adequately addresses the question asked.
>
> Return 'pass' if the response is correct, 'fail' if it's incorrect.

This evaluation process was systematically applied to all three approaches across our different parameters, *Chunk Size, Temperature*, and *Ontology Type*. The temperature was varied between 0.2 and 0.6 to test whether introducing moderate stochasticity during the generation process would improve extraction diversity without compromising compliance with the output format. Higher values were excluded to avoid parsing errors in tasks requiring strict JSON formatting.

4. Results

In total, 180 knowledge graphs were generated across the three approaches, six source texts and the full configuration space of chunk size, temperature, and ontology detail. Graph size, measured in nodes and edges, was strongly influenced by source text complexity, with longer procedural descriptions consistently producing larger graphs across all approaches. The *Large-Single-Stage* approach, based on Qwen3-32B, tended to produce the largest graphs in most configurations, while the *Small-Single-Stage* approach, using the base Llama2-13B model, generated the sparsest graphs with the highest rate of parsing errors. The *Small-Multi-Stage* approach, which combined fine-tuning with a decomposed extraction strategy, consistently achieved the highest average node degree while producing fewer unique nodes, suggesting that it captures relational structure more effectively rather than simply listing more entities.

4.1 Intrinsic Quality

Table 2 provides a comparative evaluation of ontology conformance approaches, focusing on three key intrinsic metrics: class violations, relationship violations, and edge duplicates. The *Large-Single-Stage* approach demonstrated a low level of hallucination, with only five class violations and one relationship violation, producing an average of 7.69 edge duplicates. These results suggest a high level of accuracy and coherence in the extracted knowledge, with minimal errors in class and relationship adherence. In contrast, the *Small-Single-Stage* (Llama-13B-4Q) approach exhibited significantly more violations: 179 class, and 60 relationships. However, its 7.31 edge duplicate rate is comparable to that of the *Large-Single-Stage* approach. This indicates a struggle in maintaining ontology conformance, which is likely due to limitations in accurately recognizing or extracting ontology elements. Finally, the *Small-Multi-Stage* approach (fine-tuned Llama-13B-4Q) performed almost as well as the *Large-Single-Stage* approach on class violations, producing 13 relationship violations, which is significantly more than the *Large-Single-Stage* approach but significantly less than the *Small-Single-Stage* approach. In terms of edge duplicates, this approach performed best with an average of 4.72.

Table 2: Intrinsic metrics across the three approaches.

Approach (distilled ontology)	Class violations	Relationship violations	Edge duplicates
Large-Single-Stage ***Qwen3-32B***	**5**	**1**	7.69
Small-Single-Stage ***Llama-13B-4Q***	179	60	7.31
SFT- Small-muti-stage ***Fine-tuned Llama-13B-4Q***	8	13	**4.72**

4.1.1 Extrinsic quality

Figure 2 shows the average performance across the six different guides presented in Table 1. The fine-tuned *SFT-Small-Multi-Stage* approach achieved the highest overall QA accuracy with an average pass rate of 48.4% and a peak pass rate of 55.3% when employing a chunk size of 5 and 0.2 temperature setting. The *Large-Single-Stage* approach averaged 40.3% and peaked at 46.8% with the same settings. The *Small-Single-Stage* approach performed the worst, with an average of 26.4% and a peak of 33%, using the same chunk and temperature settings. Chunk size had a notable influence on performance, while temperature showed a slight but inconsistent tendency toward better results at lower values.

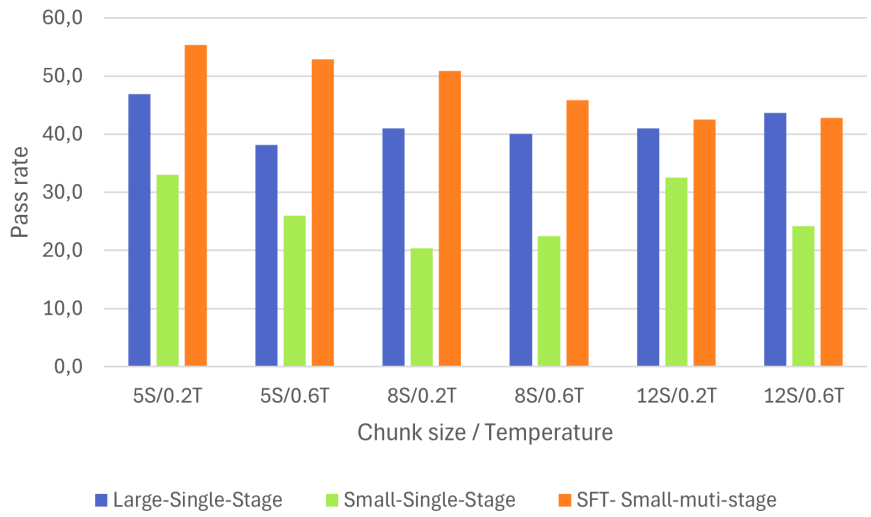

Figure 2: The three approaches showing average pass rate across 6 different chunk size+temperature extraction conditions

5. Discussion

The most notable finding is that a fine-tuned 13-billion parameter model scored higher than the 32-billion parameter general-purpose model on both intrinsic quality metrics and downstream QA accuracy. This pattern held across six procedural texts spanning device repair, 3D printing, and DIY assembly. For practitioners with limited computational resources or API budgets, this means comparable results at a fraction of the cost. This result can be attributed to two factors: supervised fine-tuning on synthetic extraction data and a decomposed extraction strategy that targeted only one or two ontology elements per phase. The decomposed extraction strategy reduced the complexity of each individual phase, allowing a smaller model to outperform a larger one that had to extract all ontology elements in a single pass. Among the intrinsic quality metrics, average node degree and duplication rate showed the strongest correlation with QA accuracy. Graphs with a higher average node degree provided a richer context during retrieval, since each seed node connected to more neighboring information. Duplicate triples had the opposite effect, filling the retrieval context with redundant information, leaving less room for relevant facts. Raw graph size, by contrast, did not predict QA performance. More triples did not uniformly translate to more useful information. Structural connectivity and ontological conformance acted as minimum thresholds rather than performance predictors. Graphs that violated them showed

degraded QA accuracy, but among graphs that satisfied both, these metrics did not account for performance differences. Over-extraction of step nodes did not degrade retrieval quality. Instead, the additional nodes provided more seed entities for Graph-RAG traversal. These findings favor recall-oriented extraction over precision. Missing a step hurts retrieval more than extracting too many. Similarly, a richer ontology did not guarantee better results. Ontology design should be driven by the intended downstream use. In this evaluation, test questions focused on step-level information, and the distilled ontology that prioritized steps outperformed the full PKO. Including ontology elements that the evaluation does not query adds extraction overhead without measurable benefit. Of the configuration parameters tested, chunk size affected extraction quality while model temperature showed a slight but inconsistent tendency toward better results at lower values. The experimental design has several limitations. Fine-tuning and the decomposed extraction strategy were introduced together, so their individual contributions cannot be separated. A sample of just six procedural texts from technical domains is small. The fine-tuned, decomposed approach performed best across all six, suggesting that the finding is not domain specific. However, it is unclear whether this would also apply to medical protocols or legal procedures. Finally, QA accuracy depends on the retrieval implementation used, and different strategies may interact differently with graph structure.

6. Conclusion and Future Work

This study shows that a fine-tuned 13-billion parameter model with a decomposed extraction strategy can outperform a much bigger general-purpose model for automated KG construction from procedural text. For practitioners, this means effective pipelines are feasible without large-scale compute infrastructure. Future work could address four directions. First, testing multi-stage extraction with base models would separate the contributions of fine-tuning from extraction strategy, clarifying whether performance gains require synthetic training data or can be achieved via prompt engineering. Beyond this, exploring even smaller models and agentic architectures, where multiple specialized sub-models collaborate on different extraction phases, could further reduce computational requirements while maintaining extraction quality. Second, expanded evaluation frameworks including semantic consistency checks and structural redundancy measures would more fully assess intrinsic KG quality, potentially using LLMs as automated judges. Additionally, constructing gold-standard reference graphs for each source text would enable triple-level completeness evaluation, which the current proxy metrics cannot provide. Third, broader datasets covering varied domains, modalities, and formality levels, including audio-transcribed and written instructions, would test the generalizability of the findings. Fourth, extension beyond procedural text to user manuals, narratives, technical specifications, and scientific abstracts would explore domain-independent KG generation.

Acknowledgement

This research and development project is funded by the German Federal Ministry of Research, Technology and Space (BMFTR). The author is / The authors are responsible for the content of this publication.

Ethical Declaration: A self-evaluation document consisting of 13 questions, provided by the Ethics Committee at the Technical University of Berlin, was completed to assess the conducted research. If any of the questions had been answered with “yes,” a full ethics approval application would have been required. In this case, all 13 questions were answered with “no,” and therefore no further ethical approval was necessary. Questionnaire link: https://www.tu.berlin/en/ipa/ueber-uns/ipa/representatives-and-committees/ethics-committee

AI Declaration: AI tools were used for proofreading the entire text.

References

Carriero, V.A. *et al.* (2024) “Towards an Ontology for Procedural Knowledge in Industry 5.0.” *SOFLIM2KG-SemIIM@ ISWC*.

Carriero, V.A. *et al.* (2025) “Procedural knowledge ontology (pko).” *European Semantic Web Conference*, Springer, pp. 334–350.

Celino, I. *et al.* (2025) “Procedural knowledge management in Industry 5.0: Challenges and opportunities for knowledge graphs,” *Journal of Web Semantics*, 84, p. 100850.

Curry, E. *et al.* (eds.) (2025) *The Semantic Web: 22nd European Semantic Web Conference, ESWC 2025, Portoroz, Slovenia, June 1-5, 2025: proceedings, part II*. *European Semantic Web Conference*, Cham: Springer (Lecture notes in computer science, 15719). Available at: https://doi.org/10.1007/978-3-031-94578-6.

Deschermeier, P. and Schäfer, H. (2024) *Die Babyboomer gehen in Rente*. IW-Kurzbericht.

Eiriksdottir, E. and Catrambone, R. (2011) “Procedural Instructions, Principles, and Examples: How to Structure Instructions for Procedural Tasks to Enhance Performance, Learning, and Transfer,” *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 53(6), pp. 749–770. Available at: https://doi.org/10.1177/0018720811419154.

Es, S. *et al.* (2023) “Ragas: Automated Evaluation of Retrieval Augmented Generation.” arXiv. Available at: https://doi.org/10.48550/ARXIV.2309.15217.

Georgeff, M.P. and Lansky, A.L. (1986) “Procedural knowledge,” *Proceedings of the IEEE*, 74(10), pp. 1383–1398.

Hogan, A. *et al.* (2022) “Knowledge Graphs,” *ACM Computing Surveys*, 54(4), pp. 1–37. Available at: https://doi.org/10.1145/3447772.

Issa, S. *et al.* (2021) “Knowledge Graph Completeness: A Systematic Literature Review,” *IEEE Access*, 9, pp. 31322–31339. Available at: https://doi.org/10.1109/ACCESS.2021.3056622.

Kurz, M. (2016) “BPMN Model Interchange: The Quest for Interoperability., 8th S-BPM.”

Mo, B. et al. (2025) "KGGen: Extracting Knowledge Graphs from Plain Text with Language Models." arXiv. Available at: https://doi.org/10.48550/ARXIV.2502.09956.

Pan, J.Z. *et al.* (2023) “Large Language Models and Knowledge Graphs: Opportunities and Challenges,” *Transactions on Graph Data and Knowledge (TGDK)*. Edited by A. Hogan et al., 1(1), p. 2:1-2:38. Available at: https://doi.org/10.4230/TGDK.1.1.2.

Poveda-Villalón, M. *et al.* (2022) “LOT: An industrial oriented ontology engineering framework,” *Engineering Applications of Artificial Intelligence*, 111, p. 104755. Available at: https://doi.org/10.1016/j.engappai.2022.104755.

Seo, S. *et al.* (2022) “Structural Quality Metrics to Evaluate Knowledge Graphs.” arXiv. Available at: https://doi.org/10.48550/ARXIV.2211.10011.

Sörqvist, E.F.H. *et al.* (2025) “Egocentric expert guidance for procedural activities using Smart Glasses, Computer Vision and RAG,” *Procedia CIRP*, 134, pp. 1047–1052.

Sternberg, R.J. (2013) *Thinking and Problem Solving*. Academic Press (Handbook Of Perception And Cognition, 2).

Surif, J., Ibrahim, N.H. and Mokhtar, M. (2012) “Conceptual and Procedural Knowledge in Problem Solving,” *Procedia - Social and Behavioral Sciences*, 56, pp. 416–425. Available at: https://doi.org/10.1016/j.sbspro.2012.09.671.

Zhong, L. *et al.* (2023) “A Comprehensive Survey on Automatic Knowledge Graph Construction.” arXiv. Available at: https://doi.org/10.48550/ARXIV.2302.05019.

Zhu, R. *et al.* (2023) “Assessing the Quality of a Knowledge Graph via Link Prediction Tasks,” *Proceedings of the 2023 7th International Conference on Natural Language Processing and Information Retrieval. NLPIR 2023: 2023 7th International Conference on Natural Language Processing and Information Retrieval*, Seoul Republic of Korea: ACM, pp. 124–129. Available at: https://doi.org/10.1145/3639233.3639357.

Zhu, Xiaoyan *et al.* (eds.) (2019) *Knowledge Graph and Semantic Computing: Knowledge Computing and Language Understanding: 4th China Conference, CCKS 2019, Hangzhou, China, August 24–27, 2019, Revised Selected Papers*. Singapore: Springer Singapore (Communications in Computer and Information Science). Available at: https://doi.org/10.1007/978-981-15-1956-7.

Zhu, Y. *et al.* (2023) “LLMs for Knowledge Graph Construction and Reasoning: Recent Capabilities and Future Opportunities.” arXiv. Available at: https://doi.org/10.48550/ARXIV.2305.13168.