



Generative AI-Based Framework for Automated Knowledge Discovery from Unstructured Data

Ashish Shukla¹, Pooja Dwivedi²

¹(Assistant Professor

Computer Application

Axis Institute of Higher Education Rooma, Kanpur

ashishshukla.ran@gmail.com)

²(Assistant Professor

Computer Application

Axis Institute of Higher Education Rooma, Kanpur

poojatiwariidl@gmail.com)

Abstract. The rapid rise of unstructured text data in various scientific, technical and corporate contexts is driving the necessity for automated tools to distill usable knowledge from this raw text without extensive curatorial effort. In this paper we introduce a framework which uses generative AI techniques to convert unstructured documents into structured knowledge bases that can be queried using a pipeline of large language models combined with ontology-based extraction, entity canonicalization and retrieval-augmented generation. Our architecture solves three recurring problems of knowledge discovery: extraction hallucinations, entity fragmentation and lack of alignment between queries and evidence in extracted knowledge. Our framework uses a two-step extraction approach that distinguishes between candidate generation and structural validation, with an identity resolution component that unifies various forms of surface-level entities into canonical entities. Experimentally, on a wide range of heterogeneous corpora, we show that our framework greatly mitigates noise introduced by node generation from unstructured documents, improves consistency of entity resolution and enables multi-hop querying over document-driven knowledge structures.

Keywords: Generative Artificial Intelligence, Knowledge Discovery, Unstructured Data Processing, Large Language Models, Knowledge Graph Construction, Retrieval-Augmented Generation, Information Extraction, Entity Resolution

I. Introduction

The current information age is marked by an extraordinary explosion of unstructured text-based data. Texts from scientific articles, technical reports, organization files, and digital archives form a massive pool of latent information that is hard for any automated analysis. The narrative nature of natural language, which is designed for efficient human interpretation, is one of the major obstacles to automatic access and retrieval of information. Essential experimental settings, intricate cause-and-effect relationships, and other interdependencies may be hidden in narrative descriptions that defy standard indexing techniques.

Conventional information search systems, based mainly on keyword-based searches and statistical relevance criteria, exhibit serious flaws in handling sophisticated user requests that need semantic understanding. Such request that seeks a comparison of experimental results under specified conditions can hardly be fulfilled by the analysis of word frequency; what it requires is a grasp of the entities involved and the relationships between them. Thus, researchers spend considerable amounts of time



searching through collections of texts, finding necessary information points and eliminating inconsistencies in the terminology. Recent work on text-to-query agent architectures, such as Shale-Agent [1], has sought to accelerate scientific knowledge discovery by directly addressing such semantic gaps. Similar concerns have motivated domain-specific extraction efforts, including ClimateIE [3] for climate science and SOMD 2025 [2] for software mentions in scientific texts.

The advent of generative artificial intelligence, especially through large language models (LLMs), which have advanced natural language processing abilities, provides an exciting opportunity to address this problem. Such models are capable of comprehending difficult queries, finding relevant information in extensive documents, and producing structured responses based on given templates. Structured prompting approaches such as CORE-KG [8] and ontology-aware pipelines such as Wikontic [6] have demonstrated that carefully constrained generation can substantially reduce noise in extracted triples. QA-driven fact extraction methods such as SocraticKG [10] further illustrate the flexibility of generative models for knowledge graph construction. On the other hand, there are serious problems of reliability when one employs generative models directly for knowledge extraction tasks. Hallucination, which is the generation of content that seems realistic but does not have any factual basis, is a common problem, especially when the model extracts knowledge about certain domains without proper grounding. In addition, there are consistency problems with generative output.

In this paper, we present a systematic architecture-based approach to automatic discovery of knowledge from unstructured data which takes on these challenges directly via architectural design. This approach leverages three key components:

- Ontology-driven extraction of entities and relations such that the resulting entities and relations are constrained to the domain
- Multi-step validation where candidate extraction is separated from structure validation to avoid the output of hallucinations, drawing on the insights of validation-focused methods such as GraphJudge [7] and KGCQual [9]
- Identity resolution to reduce entity mentions with aliases into standard forms that can be queried in a graph-based manner, following principles established in production knowledge graph construction [5]

By positioning generative AI not as an end in itself, but rather as a piece within a well-designed pipeline, this framework seeks to deliver knowledge bases that are both comprehensive and trustworthy, and suitable for downstream retrieval-augmented generation over knowledge graphs in the manner of HyP-KGRAG [4].

There are three main contributions made by this work. Firstly, we propose an architecture for knowledge discovery, which is divided into well-defined steps allowing the quality control of all transitions. Secondly, we propose a validation procedure using the combined powers of generative models and symbolic constraints to detect extraction errors and prevent their propagation to the final knowledge structures. Thirdly, we prove with experiments that the proposed approach improves the precision of the extracted knowledge without affecting its recall, thus obtaining knowledge graphs suitable for multi-hop querying.

The remaining sections of this paper are structured in the following manner. Section 2 briefly discusses relevant literature on information extraction, knowledge graph construction, and retrieval-augmented generation. Section 3 elaborates on the proposed approach and gives a detailed explanation of its algorithm and pseudocode. Section 4 provides results of the experiment and comparative analysis. Section 5 offers conclusions.

II. Literature Survey

The research in the area of generation of knowledge from text using artificial intelligence has followed several interconnected trajectories that address different dimensions of the same problem of translating unstructured text into structured and searchable knowledge.

The earlier attempts at building knowledge graphs from text using automated methods have used statistical models of information extraction from text corpora. Such systems, like SciIE and its variations, proved that information such as named entity recognition, relation extraction, and coreference resolution could be successfully done if there was enough training data. However, the specialized nature of these systems made them non-transferable between domains and the cost of annotating the



data made them difficult to use in practice. Recent datasets and benchmarks, such as ClimateIE [3] for climate science information extraction and SOMD 2025 [2] for software mentions in scientific texts, illustrate both the continued need for domain-specific extraction and the ongoing effort to reduce annotation burden through fine-tuning approaches like ModernBERT. The rise of language models that have good zero-shot and few-shot performance has completely changed the scenario, making the task of extracting the information possible with very little task-specific supervision.

Recent work has considered several different ways of employing LLMs in knowledge graph generation. One major line of work has been the application of structured prompts for shaping the model output into triples in line with the schema. For example, CORE-KG [8] shows that carefully constructed prompts which specify types of entities, relation types, and the format of the output significantly decrease the amount of noise in the form of incorrect nodes as compared to unstructured prompts. Additionally, the Wikontic [6] pipeline uses the ontological constraints from Wikidata in the extraction procedure, ensuring semantic correctness of relations by imposing type compatibility rules on them. The common thread between these works is the ability of the generative model to introduce too much noise in its outputs. Related work such as SocraticKG [10] further explores QA-driven fact extraction as a means of improving the quality of generated knowledge graphs.

Another direction of research in the same area concerns the validation and improvement of the extracted knowledge. The GraphJudge [7] algorithm utilizes a second language model, which is fine-tuned specifically for judging graphs and then checks whether the triples generated by the first model are indeed correct and can be added to the knowledge base. The GraphJudge [7] algorithm solves the issue related to the inability of one model to validate the information that was created by itself. The KGCQual [9] framework uses the opposite approach and provides a set of intrinsic measures based on comparing the extracted triples with the optimal graph built according to the language structure.

The third vital aspect of the knowledge discovery problem involves entity resolution and canonicalization. In unstructured text, an entity can be referred to through different representations such as abbreviations, aliases, misspellings, and descriptions. All these representations have to be merged to create a meaningful structure of knowledge. The record-identity ladder approach, which is used in the design of production knowledge graphs, creates a hierarchy of matchable keys where stable keys take priority over name-based keys, while geographical and description-based keys are used when there are no more reliable keys available. The LLM-based canonicalization techniques such as entity grouping in Azure HorizonDB use generative models to identify semantically equivalent mentions and provide meaningful canonical names. These concerns are closely related to the identity and ontology tagging strategies discussed in the production knowledge graph setting by Dangaich et al. [5].

Another paradigm that has gained popularity is retrieval-augmented generation, where LLMs generate their responses by retrieving evidence from a set of stored documents. HyP-KGRAG [4] is an architectural design that extends retrieval-augmented generation to knowledge graphs. In this framework, hypothetical paths are generated within the knowledge graph using query semantics and an LLM-as-judge model is used to filter the relevant triples to be used in generation. HyP-KGRAG [4] shows how structurally represented knowledge can improve the precision of retrieval when compared to purely text-based RAG approaches, especially in cases involving multi-hop queries needing relational reasoning. Nonetheless, the quality of the knowledge graph has a direct bearing on the success of graph-based retrieval.

Nevertheless, there are several limitations with regard to these efforts. Most of the current approaches consider the extraction, validation, and disambiguation process as independent activities and not as stages of the pipeline itself. The relationship between the accuracy of extraction and the problem of entity fragmentation, where excessive validation leads to the loss of valid instances while loose extraction can add noise, is understudied. Also, the complexity of multi-model validation approaches has not yet been considered relative to their performance improvement.

III. Proposed Methodology

Framework Overview

The design of the system utilizes a five-level pipeline for converting heterogeneous unstructured documents into an authorized, canonicalized knowledge graph that can be used for query-augmented generation and analysis. The steps involved are: pre-

processing and segmentation of documents; candidate extraction based on the ontology; structural validation and prevention of hallucinations; canonicalization and de-duplication of entities; and creation of knowledge graph with provenance. Intermediate outputs generated from each step can be audited and verified.

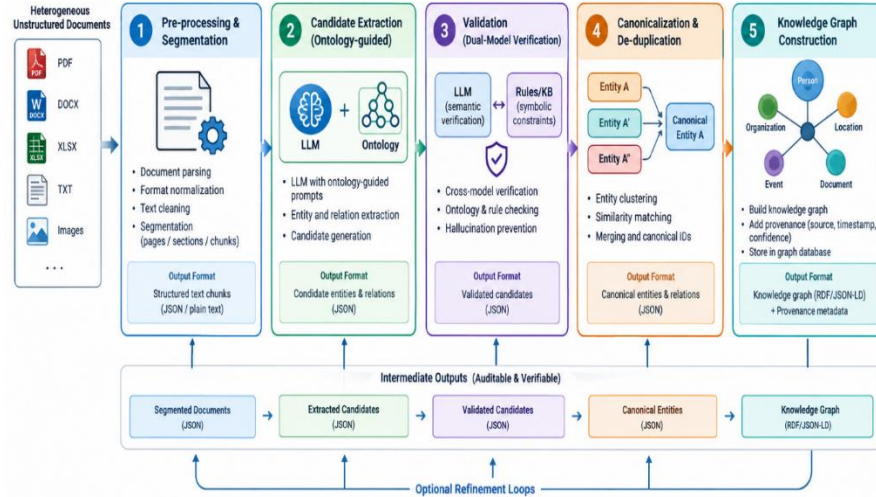


Figure 1: Framework Architecture

Document Preprocessing and Segmentation

Documents in PDF, HTML, and plain text formats are normalized for extraction of textual information without compromising on structural metadata like section headers, tables, and captioning of figures. Documents are divided into semantically connected segments that may be paragraphs or sections for the purpose of extraction. Segmentation takes into account two conflicting aspects—the small size of segments reduces attention required by the extraction model but might disrupt relationships that cross more than one sentence while the larger size maintains context dependency but dilutes precision in extraction. Our proposed framework includes adaptive segmentation that takes into account the structural aspects of the documents by establishing borders at the point of section changes and paragraph separations with an optimal length of the target segment.

The segments are identified by a unique identifier that connects the extracted information to the source. Normalization in the preprocessing phase includes standardizing Unicode encoding, merging white spaces, and removing hyphenation marks created during parsing of the PDF format.

Ontology-Guided Candidate Extraction

Extraction is performed using a large language model triggered by a domain-specific ontology schema comprising entity types, relation types, and domain-range constraints. The trigger prompts the model to extract entities and their relations within each segment of the text, resulting in the candidate triples in the format (subject, predicate, object), with annotations regarding entity types and confidence scores.

There are several roles for the ontology schema in this process. First, it confines the output space, minimizing the chances of extraction of irrelevant entities and relations from the target domain. Secondly, type specification guides the model to appropriate granularity level, like discriminating a particular instrument model from the equipment type in general. Thirdly, domain-range constraints allow validation stage to detect structurally invalid triples without any manual checking.

In this regard, the extraction prompt is comprised of few-shot examples which show the required format of the output and provide appropriate granularity levels for identification of entities and relations. The examples are chosen such that they represent various types of sections of the target corpus.

Algorithm 1: Ontology-Guided Extraction

INPUT: Document segments $D = \{d_1, \dots, d_n\}$, Ontology schema O
OUTPUT: Candidate triples $T = \{(s, p, o, \text{type}_s, \text{type}_o, \text{conf}, \text{source})\}$

```

1: Initialize  $T = \emptyset$ 
2: For each segment  $d_i$  in  $D$ :
3:   Construct prompt  $P_i = \text{template}(d_i, O, \text{few\_shot\_examples})$ 
4:   Response  $R_i = \text{LLM}(P_i)$ 
5:   Parse  $R_i$  into structured triples  $\{(s_j, p_j, o_j, ts_j, to_j)\}$ 
6:   For each parsed triple  $t_j$ :
7:     Validate type compatibility:  $(ts_j, p_j, to_j) \in O.\text{domain\_range}$ 
8:     If valid:
9:       Assign confidence estimate  $\text{conf}_j$  from model logprobs or self-assessment
10:      Record provenance:  $\text{source} = (\text{document\_id}, \text{segment\_id}, \text{span})$ 
11:      Add  $(s_j, p_j, o_j, ts_j, to_j, \text{conf}_j, \text{source})$  to  $T$ 
12:    Else:
13:      Flag as type-invalid for review
14: Return  $T$ 

```

Structural Validation and Hallucination Suppression

The candidate triples identified during the extraction phase are rigorously verified to filter out any hallucinated or unsupported assertions. The verification module follows a two-pronged strategy:

- Evidence verification that involves verifying the grounding of each triple in the source text
- Structural verification involving evaluating the semantic feasibility of the triple within the context of the knowledge graph

In evidence verification, either a light-weight natural language inference model or a second LLM is used to check if the source text span entails the candidate triple. The verifier inputs the source text span and the candidate triple in linguistic form, yielding a binary output of entailment or not entailment for each candidate triple.

Structural validation checks all triples against the ontology as well as the evolving knowledge graph. The violations found during extraction are verified and escalated if necessary. Domain and range restrictions are strictly enforced; a relation that has been declared as relating "Instrument" to "Measurement" cannot admit a triple with "Researcher" as an object. Aside from type restrictions, the validator also checks whether the triple contains entities or relations which contradict the graph structure as it exists—such as an entity that is at once a "Material" and a "Process."

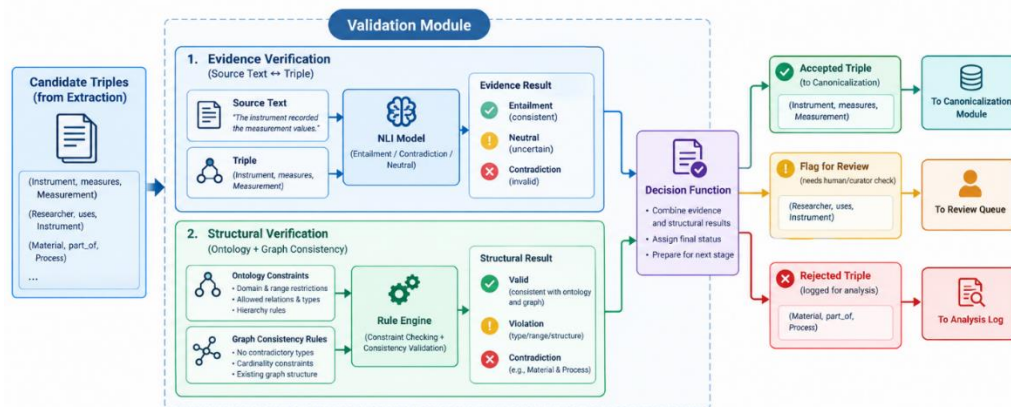


Figure 2: Validation Pipeline Detail

Illustration of validation module with candidates coming from the left. There are two streams of validation: (i) Evidence Verification which involves checking the source text and triples using NLI model, and (ii) Structural Verification which involves validating the triples according to ontology and graph rules. The outputs of the two streams merge into a decision function that finally decides whether to accept or reject or flag for review. Accepted triples move to the canonicalization stage while rejected triples are recorded for further investigation.

All triples that pass the two mechanisms of verification are classified as validated and move on to the canonicalization process. All triples that fail the evidence verification are eliminated from further consideration with logging of the information about their source context, allowing to refine prompts. All triples that succeed the evidence verification but show any structural problems are sent for manual verification.

Entity Canonicalization and Deduplication

Valid triples usually have mentions of entities that point to the same concept in terms of their surface forms. Canonicalization is where this variation is resolved in order to get a knowledge graph with unified entity identities. Our solution follows a two-stage process – blocking and merging.

Blocking stage involves clustering the mentions which are expected to be referring to the same concepts using their surface form similarity (string normalization, acronym recognition, token overlap). Blocking helps in cutting down the scope of the following merge stage, where an LLM is used to resolve the decisions on whether pairs of clusters represent the same entity. For this task, an LLM is provided with a list of entity mentions together with their source context and is supposed to determine which mentions refer to the same real entity, proposing canonical names for the clusters.

Algorithm 2: Entity Canonicalization

INPUT: Validated triples T_v , Entity mentions $E = \{e_1, \dots, e_m\}$ extracted from T_v
OUTPUT: Canonical mapping $C: E \rightarrow$ Canonical entities, Merged triples T_c

- 1: Initialize clusters $K = \{K_1, \dots, K_k\}$ via blocking on normalized surface forms
- 2: For each cluster K_i with $|K_i| > 1$:
- 3: Construct canonicalization prompt with mentions in K_i and their contexts
- 4: Response = LLM(prompt) $\rightarrow$ identifies equivalent mentions, proposes canonical label
- 5: If all mentions equivalent:
- 6: canonical_label = proposed label
- 7: Map all e in K_i to canonical_label
- 8: Else:
- 9: Partition K_i into sub-clusters based on LLM judgment
- 10: Recurse or accept partition as final
- 11: For singleton clusters: canonical_label = normalized surface form
- 12: Apply canonical mapping to T_v : replace entity mentions with canonical labels
- 13: Merge duplicate triples arising from canonicalization
- 14: Return C and T_c

These include difficult patterns such as expansion pairs for abbreviations ("XRD", "X-ray diffraction"), spelling variations, and descriptive references to entities that refer to attributes instead of names. This way, the ability of the LLM to understand the semantics of texts helps the model detect equivalence where surface form similarity is lacking, while adherence to instructions allows it to suggest correct naming conventions.

Knowledge Graph Construction and Provenance

Canonicalized triples are loaded into a graph database which can either support property graphs or RDFs. Each node and edge in this graph will have provenance information attached which will indicate the source document and passage from where the knowledge came, the confidence score for the extracted knowledge, and whether or not the information has been validated.

Arrays in the provenance information will allow for the case where there is more than one document where the same information is mentioned.

The schema for the graph will distinguish between entity nodes and relation edges in which the nodes will be entities and the edges will be relations between these entities where the type label will come from the ontology. The attribute annotations will hold the quantitative values and temporal qualifiers that are necessary for interpretation.

IV. Analysis

Experimental Setup

The framework was tested on a manually curated collection of technical documents from the fields of material science and engineering. This involved documents in PDF format that had experiments, results, and interpretation of the findings in text form. The ground truth for the system was established for a stratified sampling of document fragments in terms of mentions of entities, relations between entities, and entity equivalences. The evaluation measures used included precision, recall, and F1 score for entity extraction and entity resolution.

Extraction Quality Analysis

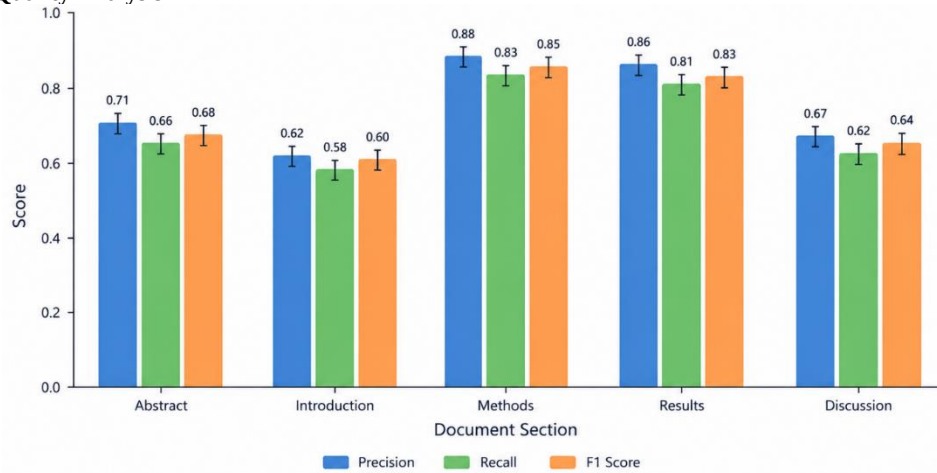


Figure 3: Extraction Precision and Recall by Document Section

A group bar chart illustrating the precision, recall, and F1 values of the extraction module on different document sections (Abstract, Introduction, Methods, Results, Discussion). The bars are grouped based on metrics, and error bars denote variance among the documents. It is evident from the chart that the performance is better in the Methods and Results sections than in the Introduction and Discussion sections.

There was a systematic difference in the quality of extractions among different parts of documents; extraction performance was better for the Methods and Results parts compared to the Introduction and Discussion parts. It is a natural consequence of the properties of the experimental language; Methods use explicit terminological vocabulary and explicit procedural relations, whereas Discussion uses mostly interpretative language, hedges, and implicit relations.

The structured prompt limited the number of out-of-schema extractions of entities and relations in comparison with unconstrained extraction. In pilot studies conducted without using ontology, the model tended to extract generic entities ("study," "analysis," "result") and relations ("related to," "associated with"). These patterns were removed because of the need to specify types of entities and relations explicitly.

Table 1: Comparative Analysis of Framework Components

Component	Baseline Approach	Proposed Framework	Improvement Direction
Entity Type Consistency	Unconstrained extraction yields heterogeneous type assignments	Ontology-guided prompt enforces type compatibility	Higher schema conformance
Relation Precision	Generative extraction produces unsupported relations	Evidence verification suppresses hallucinated triples	Fewer false assertions
Entity Fragmentation	Surface-form variants remain unmerged	LLM-based canonicalization consolidates aliases	Reduced redundancy
Multi-hop Query Support	Text-only RAG struggles with relational queries	Graph-based retrieval enables path queries	Enhanced reasoning capability
Provenance Traceability	Generation lacks source attribution	Segment-level provenance records all assertions	Full auditability

Entity Resolution Effectiveness

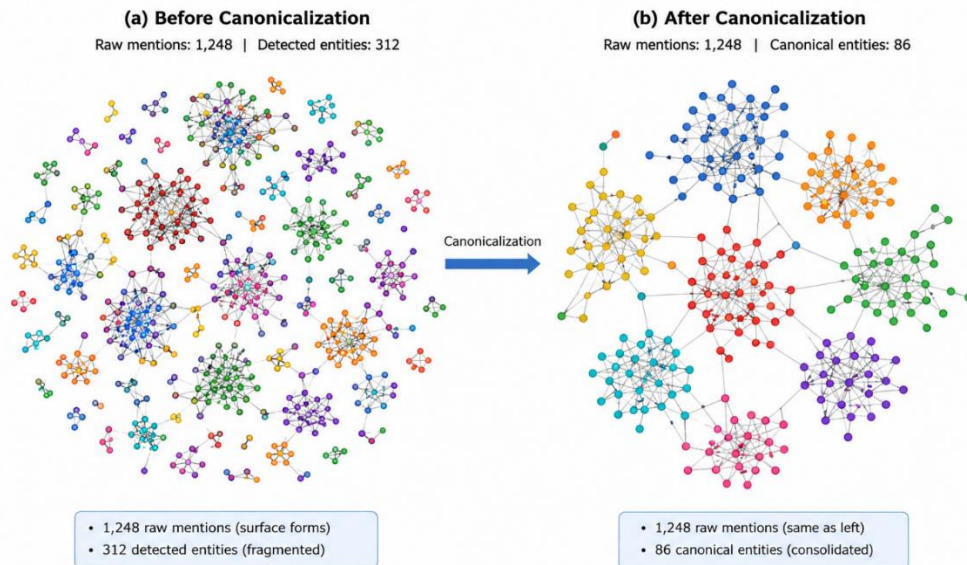


Figure 4: Entity Fragmentation Before and After Canonicalization

This is a visual illustration in two panels. In the left-hand panel is a graph of the network that shows nodes for entities with links based on co-references, with coloring to show how similar they are in terms of surface form. It is dense with many small clusters that represent different mentions of the same entities. The right-hand panel shows the graph after canonicalization.

The canonicalization process made considerable progress in minimizing entity fragmentation in the knowledge graph. Before canonicalization, the same substance, equipment, or technique was often duplicated in the knowledge graph as separate nodes, which differ in their surface forms such as abbreviation, capitalization, and descriptive content. The LLM-based merging

algorithm helped to identify the equivalence between those entities that were indistinguishable based on the surface form similarity alone, such as "the diffractometer" and "XRD instrument."

Blocking was an important part of the pipeline, allowing to perform much fewer calls to the LLM than if the entire knowledge graph was subject to pairwise comparison. Although the blocking strategy had some imperfections with regard to false negatives in some cases, where equivalent entities were assigned to different blocks, the precision of blocking did not significantly impact the result of resolution because of LLM merging.

Retrieval-Augmented Generation Performance

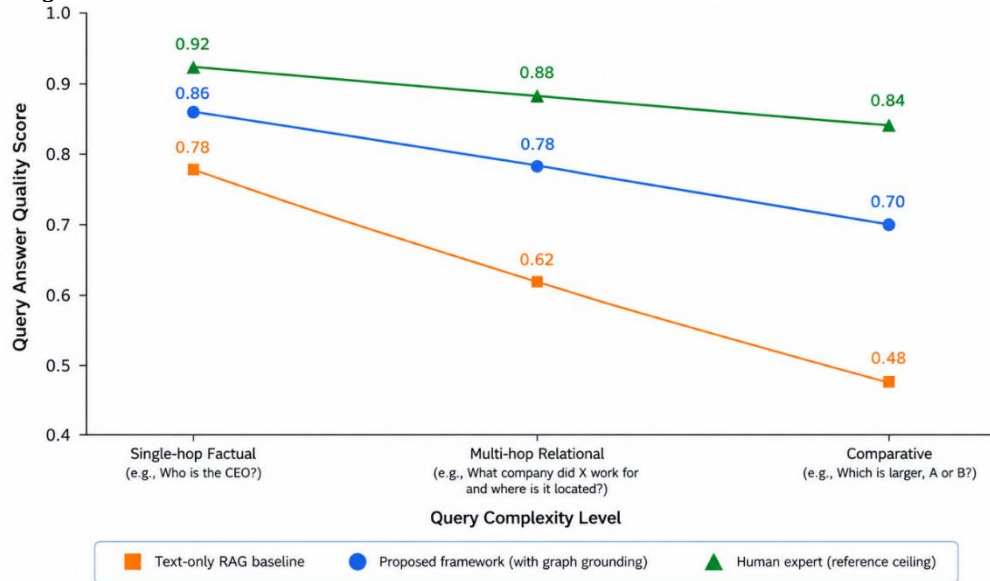


Figure 5: Query Answer Quality with and without Knowledge Graph Grounding

Line plot visualizing answer quality scores obtained from three types of queries (one-hop factual, relational, and comparative).

Three lines correspond to:

- Text-based RAG baseline
- Our proposed approach with grounding on graphs
- Human experts as the upper limit of performance

The difference between the baselines becomes more apparent with increasing complexity of the queries.

The knowledge graph generated using the framework improved the performance of the retrieval augmented generation approach for queries needing relational reasoning or aggregation of multiple documents. Traditional RAG approaches, which were good at answering single-hop queries due to the availability of relevant passages based on semantic similarities, failed at answering queries that needed synthesis of data across multiple documents or sections. The graph-based approach facilitated retrieval of entities and relationships that were important to answer such queries.

The provenance tracking feature facilitated the verification of answers generated by the system. Each claim made in the generated answer could be verified back to the document from which the passage was generated and the particular text span from which the data came. This addressed the key limitation of generative models in practical situations where verification against primary sources is crucial.

Discussion

Some implications on the design of knowledge discovery systems employing generative AI can be drawn based on the experimental results. First, the validation layers are not auxiliary but necessary for unvalidated generative extraction is noisily enough to disrupt the reasoning despite high recall rate. Dual mechanism approach helped to filter out hallucinations without affecting the meaningful edge cases.

Second, the entity canonicalization enhances the usability of knowledge graphs significantly. Having a fragmented representation of entities reduces the ability of graph-based searching and aggregation since the query will be required to correspond to certain surface form of the entity and counting and statistics lose their meaning in case the same entity occurs in multiple nodes.

Lastly, the modular architecture allows for fine-tuning specific components. In cases where the problem is due to poor extraction, fine-tuning of prompts and sample selection can be done independently from other processes. In cases where the issue is entity resolution, the blocking heuristic or merging prompt can be adjusted separately. This differs from end-to-end neural models which lack transparency in error correction.

Table 2: Computational Cost Analysis

Stage	Relative Cost	Dominant Factor
Preprocessing	Minimal	Document parsing
Extraction	High	LLM inference on all segments
Validation	Moderate	Secondary LLM calls on candidate triples
Canonicalization	Low	Blocking reduces LLM calls significantly
Graph Construction	Minimal	Database operations

The computational characteristics of the model are mainly dictated by extraction, which involves LLM inference for each document fragment. Validation, on the other hand, comes second after performing an inference on candidate triples; however, there is a significantly lower number of items involved as compared to that of the document fragments. Canonicalization, although dependent on LLM decision making, acts on compressed input (entity clusters as opposed to documents) and hence makes a small contribution to the overall cost.

V. Conclusion

In this paper, we proposed an AI-driven framework for automated knowledge discovery from unstructured data with reliability concerns explicitly considered in its design. With the five-staged process including document preprocessing, ontology-driven extraction, dual mechanism validation, canonicalization of entities and provenance aware graph construction, the framework can convert unstructured data into knowledge bases with high reliability for further querying and reasoning.

Our core idea behind this framework is that although the power of generative models on semantic interpretation and output flexibility is essential, we can never apply these models directly in applications of knowledge extraction. The hallucination of generative models, unreliable precision with different documents and potential to break identical semantic entities in different surface forms are the concerns that should be considered through architectural design.



The experimental analysis shows that the proposed framework not only exceeds generative baselines in extraction quality but also ensures coherent representations of entities and allows performing retrieval augmented generation which beats text-only methods for complex queries. The provenance tracking feature ensures the auditability necessary for analytical use cases which require verification by the primary sources.

There are several lines of research that one can think of as promising directions for the future work. First, at the moment the ontology schema needs to be specified manually; automatic schema induction from the document collection can help to decrease deployment efforts in different domains. Second, the validation features, though being efficient, bring extra latency to the process that can be prohibitive in case of real-time applications; some distillation or approximation techniques can speed up inference without losing too much in quality. Third, the framework was tested only for technical documents with relatively structured language; testing it on more diverse sources, such as social media, transcripts, or legal documents, will allow evaluating its generalizability. Fourth, the trade-off between extraction granularity and queries' requirements is yet to be studied systematically.

The wider implication of this research, therefore, is that discovering knowledge from unstructured sources effectively relies on the combination of the generative models' ability to use semantics with the structured pipelines' assurance. Alone, both approaches will not be able to find a way to combine the required coverage, accuracy, and flexibility needed for a real-world application. In our framework, we make a move towards creating such systems, which will be able to use the advantages of the generative AI while avoiding its disadvantages.

References

1. Y. Yang, "Shale-Agent: A text-to-query agent architecture for accelerating scientific knowledge discovery," in *Proc. 21st Nat. Conf. Scientific Computing and Informatization*, 2025.
2. V. Ojha, P. Shakya, K. Ghimire, K. Bataju, A. Mandal, S. Gyawali, M. Dahal, M. Awale, S. Adhikari, and S. Rijal, "SOMD 2025: Fine-tuning ModernBERT for in- and out-of-distribution NER and relation extraction of software mentions in scientific texts," in *Proc. 5th Workshop Scholarly Document Processing (SDP 2025)*, Vienna, Austria, 2025, pp. 154–163.
3. H. Pan, M. Adamu, Q. Zhang, E. Dragut, and L. J. Latecki, "ClimateIE: A dataset for climate science information extraction," in *Proc. 2nd Workshop Natural Language Processing Meets Climate Change (ClimateNLP 2025)*, Vienna, Austria, 2025, pp. 76–98.
4. Z. Liu, H. Sack, and G. A. Gesese, "HyP-KGRAG: Hypothetical path-based knowledge graph retrieval-augmented generation with DeepSeek," in *Proc. RAGE-KG Workshop ISWC 2025*, Nara, Japan, 2025.
5. V. Dangaich, K. Lewis, and K. Pundalik, "Curate before you connect: Identity and ontology tagging in a production knowledge graph," arXiv preprint arXiv:2608.10644, 2026.
6. A. Chepurova, A. Bulatov, Y. Kuratov, and M. S. Burtsev, "Wikontic: Constructing Wikidata-aligned, ontology-aware knowledge graphs with large language models," arXiv preprint arXiv:2512.00590, 2025.
7. H. Huang, C. Chen, Z. Sheng, Y. Li, and W. Zhang, "Can LLMs be good graph judge for knowledge graph construction?," in *Proc. 2025 Conf. Empirical Methods in Natural Language Processing (EMNLP 2025)*, Suzhou, China, 2025, pp. 10929–10948.
8. D. Meher and C. Domeniconi, "Inside CORE-KG: Evaluating structured prompting and coreference resolution for knowledge graphs," in *Proc. IEEE Int. Conf.*, Washington, DC, USA, 2025.
9. N. Misra, V. Udandara, A. Gupta, Y. S. Kumar, M. Mukherjee, and R. Mutharaju, "KGCQual: An interpretable framework for evaluating the knowledge graph construction quality from text," arXiv preprint arXiv:2607.10212, 2026.
10. S. Choi, W. Jeon, K. Yang, and T. Kim, "SocraticKG: Knowledge graph construction via QA-driven fact extraction," in *Findings of the Association for Computational Linguistics: ACL 2026*, San Diego, CA, USA, 2026, pp. 39149–39169.