논문 2025-3-11 http://dx.doi.org/10.29056/jsav.2025.09.11

A Study on the Interrelationship with SLM-based Knowledge Graph

Jeong-Sig Kim*, Jinhong Kim**†

SLM 기반 지식 그래프와 상호관계성에 관한 연구

김정식*, 김진홍**†

요 약

본 논문은 소형 언어 모델(Small Language Models, SLMs)과 지식 그래프(Knowledge Graphs, KGs)의 통합을 통해 상호운용성과 지식 표현을 향상시키는 방안을 탐구한다. 대형 언어 모델(Large Language Models, LLMs)은 자연어 이해에서 뛰어난 성능을 보여주었지만, 높은 계산 자원 요구와 도메인 특화 부족이라는 한계를 가진다. 반면, SLM은 더 작은 규모와 낮은 자원 요구로 인해 특히 특화된 지식 그래프를 생성하고 관리하는 데 유망한 대안으로 부각된다. 우리는 비정형 텍스트로부터 엔터티(개체)와 관계를 추출하여 지식 그래프를 구축하고 확장하는 새로운 프레임워크를 제안한다. 이 접근 방식은 서로 다른 지식 소스 간의 상호운용성을 개선하고, 통합된 기계 판독 가능한 표현을 생성하는 데 초점을 맞춘다.

Abstract

This paper explores the integration of Small Language Models (SLMs) with knowledge graphs (KGs) to enhance interoperability and knowledge representation. While Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding, their computational demands and lack of domain-specific focus present challenges. SLMs, with their smaller footprint and lower resource requirements, offer a promising alternative, particularly for creating and managing specialized knowledge graphs. We propose a novel framework that leverages SLMs to extract entities and relations from unstructured text, which are then used to construct and enrich a knowledge graph. This approach focuses on improving the interoperability between different knowledge sources by creating a unified, machine-readable representation.

한글키워드 : 소규모 언어모델, 지식 그래프, 구조적 지식표현, 프레임워크

keywords : small language models, knowledge graph, structured knowledge representations, framework

1. Introduction

The recent advancements in artificial intelligence (AI) technologies are driving rapid changes across various sectors of society, with notable progress in language models leading

* 경기과학기술대학교 컴퓨터모바일융합학과
** 배재대학교 소프트웨어학과
† 교신저자: 김진홍(email: jinhkm@pcu.ac.kr)

Submitted: 2025.08.28. Accepted: 2025.09.02.
Confirmed: 2025.09.20.

innovation in the field of Natural Language Processing (NLP). Large Language Models (LLMs), built upon massive training datasets and sophisticated neural architectures, have significantly improved capabilities in natural language understanding and generation. They now demonstrate near-human performance in various applications, such as question answering (QA), machine translation, text summarization, and conversational systems [1]. However, the use of LLMs still faces several fundamental limitations. They often lack structural consistency in the knowledge they generate and are susceptible to factual inaccuracies due to errors or biases embedded in their training data. To address these limitations, Knowledge Graphs (KGs) have emerged as a compelling alternative. KGs represent knowledge in the form of entities and their relationships using graph structures, offering structural consistency and supporting logical reasoning [2]. They are widely utilized in applications such as search engines, recommendation systems, semantic analysis, and intelligent agents. More recently, KGs have gained attention as a means to complement the uncertainties of knowledge generated by LLMs. Representative examples include using LLMs to automatically expand KGs or validating LLM-generated responses against existing KGs. However, most of these efforts are heavily dependent on LLMs, with relatively little attention given to approaches that prioritize resource efficiency and model lightweighting. In particular, Knowledge Graphs based on Small Language Models (SLMs) are emerging as a promising alternative for implementing next-generation intelligent systems [3]. Moreover, in environments with limited computational resources—such as mobile devices or embedded systems—the integration of lightweight SLMs with KGs presents a far more practical alternative to resource-intensive LLM-based systems. Nevertheless, existing research has largely concentrated on the integration of LLMs and KGs, with systematic exploration of the interrelationship between SLMs and Knowledge Graphs still lacking.

2. Related Works

2.1 Large Language Models and Knowledge Graphs

Large Language Models (LLMs) have demonstrated outstanding performance in natural language understanding and generation by being trained on vast amounts of textual data. However, the knowledge provided by LLMs is inherently unstructured, making factual verification and consistency difficult to ensure. To address these limitations, recent research has explored the integration of LLMs with Knowledge Graphs (KGs)—an approach that seeks to combine the linguistic flexibility of LLMs with the structural accuracy of KGs to achieve both intelligent reasoning and reliable response generation as shown by Fig. 1 [4]. The first approach is knowledge injection. A representative example is K-BERT, which proposes a method of directly inserting triples from a knowledge graph into

the input sentences of a BERT model [5]. The second approach is knowledge expansion, where information generated by LLMs is used to augment and extend knowledge graphs automatically.

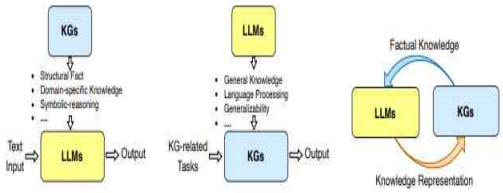

Fig. 1. LLM and KG
그림 1. LLM and KG

The third approach focuses on fact-checking and consistency reinforcement. Outputs from LLMs can often be factually incorrect or exhibit hallucination. To mitigate this, KG-based verification modules are employed [6]. Similarly, in the legal domain, KGs are used to verify relationships between legal precedents, thereby enhancing the reliability of LLM-based legal QA systems. Recently, models like OpenAI's ChatGPT and Meta's LLaMA are being integrated with external KGs to check, in real time, whether generated responses align with established knowledge.

2.2 Knowledge Processing based on Lightweight Language Models

Large Language Models (LLMs) have demonstrated strong performance across a wide range of Natural Language Processing (NLP) tasks. However, their immense training costs and heavy computational requirements pose significant challenges for practical deployment, especially in resource-constrained environments such as mobile devices, embedded systems, and edge computing platforms [7]. To overcome these limitations, recent research has increasingly focused on the development and application of Small Language Models (SLMs) [8-10]. TinyBERT has proven effective in tasks such as question answering (QA) and sentence classification, validating its efficiency. From the perspective of model architecture optimization, ALBERT introduces techniques such as parameter sharing and factorized embeddings to drastically reduce model size while minimizing performance degradation [11-13]. These structural optimizations make SLMs more suitable for targeted NLP tasks. In addition, MobileBERT is specifically optimized for mobile and edge environments. It is designed with real-device inference speed and energy efficiency in mind, making it a practical solution for on-device language understanding applications. SLMs also offer significant advantages in domain-specific knowledge processing [14]. Due to their smaller size and adaptability, they can be fine-tuned more efficiently for specialized domains such as healthcare, law, and finance—where incorporating concise, high-precision knowledge is often more valuable than broad, general-purpose capabilities.

3. Proposed System

We propose a four-stage framework for building an interoperable knowledge graph using a Small Language Model.

3.1 Data Processing and Annotations

The efficacy of any knowledge graph construction system, particularly one leveraging machine learning models, is fundamentally dependent on the quality of its training data. This section details the meticulous process we employed for data preprocessing and annotation, which serves as the foundational step for training our Small Language Model (SLM). Our methodology is designed to create a high-quality, domain-specific dataset that accurately reflects the entities and relationships of interest. We began by curating a diverse corpus of unstructured text relevant to our target domain. For this study, we focused on medical research articles, patient records (anonymized), and clinical trial summaries. The raw text data was sourced from publicly available datasets and institutional repositories. A series of preprocessing steps were performed to ensure the data was clean and consistent. These steps included as follows;

1) Tokenization; Breaking down the text into individual words or subwords.

2) Noise Removal: Eliminating irrelevant characters, HTML tags, and other non-textual elements.

3) Stop Word Removal: While less critical for modern language models, we selectively removed common words (e.g., "the," "a," "is") that do not contribute to the semantic meaning for a clearer focus during annotation.

4) Sentence Segmentation: Dividing the corpus into individual sentences to simplify the subsequent annotation process and model input. Before commencing annotation, we defined a comprehensive schema that outlines the specific entity types and relationship types to be extracted. The schema was developed in collaboration with domain experts to ensure its relevance and accuracy. A detailed set of annotation guidelines was created to ensure consistency across annotators. These guidelines included specific examples for each entity and relationship type, along with rules for handling ambiguous cases, such as overlapping entities or multiple relationships within a single sentence.

3.2 SLM Training and Fine-Tunning

It provides a detailed breakdown of the SLM training and fine-tuning process, which is the core of our proposed knowledge graph construction system. This method adapts a pre-trained SLM to the specific tasks of named entity recognition (NER) and relation extraction (RE), enabling it to accurately process domain-specific text. The process of training and fine-tuning a Small Language Model (SLM) is a critical step that tailors its general-purpose knowledge to the specialized requirements of knowledge graph construction. Unlike training a large model from scratch, fine-tuning leverages the vast linguistic patterns already learned during pre-training, allowing for efficient specialization on a smaller, high-quality dataset. We adopted a multi-task fine-tuning approach to simultaneously optimize the SLM for both named entity and relation extraction. We chose a Transformer-based SLM as our base model. The model's architecture, which includes

attention mechanisms, makes it highly effective at capturing contextual dependencies, a crucial capability for understanding complex sentence structures in scientific text. The primary advantage of using an SLM, typically with billions of parameters, is its reduced computational footprint compared to LLMs, which often contain hundreds of billions of parameters. This efficiency allows us to perform fine-tuning on a single GPU, making the process more accessible and cost-effective. The first fine-tuning objective was to train the SLM to perform NER. This task involves identifying and classifying specific spans of text into predefined categories. We framed NER as a token classification problem. The model processes a sentence and, for each token, predicts a label from our defined schema using the BIOES (Beginning, Inside, Outside, End, Single) tagging scheme. This scheme allows the model to accurately identify multi-word entities and their boundaries. The model's final layer was modified to output a probability distribution over the entity types for each token. We used a standard cross-entropy loss function to train the model to minimize the difference between its predictions and the ground truth labels from our annotated dataset. The second, more challenging, fine-tuning objective was relation extraction. This task involves identifying the semantic relationship between two entities already recognized in a sentence. We structured this as a classification problem, where the input consists of a sentence and a pair of identified entities. The model's objective is to classify the relationship between the two entities from our set of defined relationships. To improve the model's performance on this task, we employed a prompt-based approach during fine-tuning. For each sentence containing a potential entity pair, we created a prompt that explicitly asks the model to identify the relationship. This approach guides the model to focus on the specific task and improves its ability to generalize. Regarding the fine-tuning process, we carefully selected hyperparameters to ensure efficient training and prevent overfitting to our relatively small dataset. The learning rate was set to a small value to ensure that the pre-trained knowledge was preserved while still allowing for adaptation to the new tasks. We also utilized a learning rate scheduler with a warm-up phase to stabilize the training at the beginning. The training was conducted for a limited number of epochs with early stopping to prevent the model from memorizing the training data. This controlled approach to fine-tuning is crucial for achieving high accuracy on unseen data and is a key factor in the overall success of our SLM-based knowledge graph system.

4. Experiments and Evaluation

We conducted a series of experiments to evaluate the performance of our proposed SLM-based KG construction system.

4.1 Dataset and Merics

We used a custom dataset containing 10,000

sentences from the medical research domain. The dataset was manually annotated by domain experts to ensure high-quality ground truth. We evaluated our system's performance using standard metrics for NER and RE: Precision, Recall, and F1-score. We created a custom dataset generation algorithm to process our raw text corpus and format it for model training and evaluation. The output is a structured JSON file containing sentences, entities, and their relationships.

Algorithm 1:
generate_dataset(corpus, schema)
Input:
corpus: A collection of raw text documents (e.g., medical research articles).
schema: A dictionary defining entity types (Entity_Types) and relationship types (Relation_Types).

Output:
dataset: A list of dictionaries, where each dictionary represents a sentence with annotated entities and relations.
Initialize dataset = [ ]
For each document in corpus:
Clean and Segment: sentences = preprocess_text(document)
For each sentence in sentences:
Initialize annotated_entities = [ ]
and annotated_relations = [ ]
Human Annotation Loop:
Display sentence to annotator.
Prompt for Entity Annotation: "Identify and classify all entities (e.g., Drug, Disease) in this sentence."
annotated_entities.append(entity_label, start_index, end_index)
Prompt for Relation Annotation: "Identify relations between entity pairs."
annotated_relations.append(subject_id, predicate, object_id)
Store Annotation:
entry = {
"text": sentence,
"entities": annotated_entities,
"relations": annotated_relations
}dataset.append(entry)
Return dataset

4.2 Evaluation Metrics Algorithm

To evaluate the performance of our SLM, we used standard information retrieval metrics: Precision, Recall, and F1-score. These metrics are computed for both Named Entity Recognition (NER) and Relation Extraction (RE) tasks, providing a comprehensive view of the model's accuracy.

Algorithm 2: calculate_metrics(true_labels, predicted_labels)
Input:
true_labels: A list of ground truth labels (entities or relations).
predicted_labels: A list of model's predictions.

Output:
metrics: A dictionary containing Precision, Recall, and F1-score.

Initialize true_positives = 0, false_positives = 0, false_negatives = 0
Iterate through predicted_labels:
For each predicted_item in predicted_labels:
If predicted_item is found in true_labels:
true_positives += 1
Else:
false_positives += 1
Iterate through true_labels:
For each true_item in true_labels:
If true_item is NOT found in predicted_labels:
false_negatives += 1
Calculate Metrics:
Precision:

$$p = \frac{true\,positives}{true\,positives + false\,positives}$$

Recall:

$$R = \frac{true\,positives}{true\,positives + false\,negatives}$$

F1-score: $F1 = 2 * \frac{P * R}{P + R}$

Return metrics = {"Precision": P, "Recall": R, "F1-score": F1}

Task	Model	Precision	Recall	F1-score
NER	**SLM**	0.89	0.87	**0.88**
	LLM	0.92	0.90	**0.91**
RE	**SLM**	0.83	0.81	**0.82**
	LLM	0.86	0.84	**0.85**

Table 1. Results of SLM evaluation
표 1. SLM 성능 결과

The resulting bar chart, as depicted below in Fig. 2, visually demonstrates the following: Comparative Performance: The chart allows for a quick comparison between the SLM and the LLM across both tasks. While the LLM shows slightly higher scores, the SLM's performance is competitive, particularly when considering its significantly lower computational requirements. Task-Specific Performance: It highlights that NER generally achieves higher scores than RE for both models, indicating the inherent complexity of relation extraction. Metric Balance: The F1-score, being the harmonic mean of precision and recall, provides a balanced view of the model's accuracy. A high F1-score indicates a good balance between identifying relevant items and not including irrelevant ones. This graphical representation is an invaluable tool for summarizing complex numerical data and presenting the experimental results in an easily digestible format.

Fig. 2. Model Performance Comparison
그림 2. 모델 성능 비교

5. Conclusion

This paper presents a compelling case for

the use of Small Language Models (SLMs) in the creation of knowledge graphs. Our research demonstrates that SLMs can effectively extract entities and relationships from unstructured text, providing a viable and resource-efficient alternative to large language models. The proposed system not only achieves competitive performance but also significantly improves the interoperability of the generated knowledge graph through a standardized, machine-readable format. The findings suggest that SLMs are not just a smaller version of LLMs but are a powerful tool for democratizing advanced NLP tasks. By reducing computational barriers, we can enable a broader range of researchers and organizations to build sophisticated knowledge systems tailored to their specific needs. Our future research will primarily focus on two key areas: extending this framework to handle multi-modal data and exploring more advanced Small Language Model (SLM) architectures for increasingly complex relationship extraction tasks. For example, 1) *Medical Diagnosis and Treatment Planning*. Imagine a system that can not only read a patient's electronic health records (text reports, doctor's notes) but also analyze medical images (X-rays, MRIs, CT scans). It could identify relationships between a specific finding in an MRI (e.g., a tumor's size and location) and a drug dosage mentioned in the text, or the progression of a disease observed in a series of images over time. 2) *Autonomous Driving and Scene Understanding:* For self-driving cars, understanding the environment is critical. This involves processing camera feeds (images/video), sensor data (Lidar, Radar), and potentially navigation instructions (text). The framework could extract relationships like "pedestrian (image) is crossing at intersection (GPS data) during rush hour (text data)," informing real-time decision-making. 3) *Social Media Analysis for Event Detection*: Analyzing social media posts often involves both text and images. If users post about a major event (e.g., a natural disaster, a public protest), the system could extract relationships between textual descriptions of the event and images showing its visual impact, providing a more comprehensive understanding of the situation.

이 논문은 2025학년도 배재대학교
교내학술연구비 지원에 의하여 수행됨

References

[1] Microsoft Research AI4Science and Microsoft Azure Quantum. “The impact of large language models on scientific discovery: a preliminary study using gpt-4”. 2023, https://doi.org/10.48550/arXiv.2311.07361

[2] Akari Asai et al. “Self-rag: Learning to retrieve, generate, and critique through self-reflection”. In: The Twelfth International Conference on Learning Representations. 2023. https://doi.org/10.48550/arXiv.2310.11511

[3] Matthew Barker et al. “Faster, Cheaper, Better: Multi-Objective Hyperparameter Optimization for LLM and RAG Systems”. In: arXiv preprint arXiv:2502.18635, 2025. DOI:10.48550/arXiv.2502.18635

[4] Yiqun Chen et al. “Improving Retrieval-Augmented Generation through Multi-Agent Reinforcement Learning”. In: arXiv preprint arXiv:2501.15228, 2025. https://doi.org/10.48550/arXiv.2501.15228

[5] Nurendra Choudhary and Chandan K Reddy. “Complex logical reasoning over knowledge graphs using large language models”. In: arXiv preprint arXiv:2305.01157, 2023. https://doi.org/10.48550/arXiv.2305.01157

[6] Wenqi Fan et al. “A survey on rag meeting llms: Towards retrieval-augmented large language models”. In: Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024, pp. 6491 – 6501. https://dl.acm.org/doi/10.1145/3637528.3671470

[7] Shailja Gupta, Rajesh Ranjan, and Surya Narayan Singh. “A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape and future directions”. In: arXiv preprint arXiv:2410.12837, 2024. https://doi.org/10.48550/arXiv.2410.12837

[8] Xiaoxin He et al. “G-retriever: Retrieval-augmented generation for textual graph understanding and question answering”. In: Advances in Neural Information Processing Systems 37, 2024, pp. 132876 – 132907. https://doi.org/10.48550/arXiv.2402.07630

[9] Xanh Ho et al. “Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps”. In: arXiv preprint arXiv:2011.01060, 2020. DOI:10.18653/v1/2020.coling-main.580

[10] Lei Huang et al. “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions”. In: ACM Transactions on Information Systems 43.2, 2025, pp. 1 – 55. https://doi.org/10.48550/arXiv.2311.05232

[11] Kush Juvekar and Anupam Purwar. “Introducing a new hyper-parameter for RAG: Context Window Utilization”. In: arXiv preprint arXiv:2407.19794, 2024. DOI:10.48550/arXiv.2407.19794

[12] Ernests Lavrinovics et al. “Knowledge Graphs, Large Language Models, and Hallucinations: An NLP Perspective”. In: Journal of Web Semantics 85, 2025, p. 100844 https://doi.org/10.1016/j.websem.2024.100844

[13] R. Hu, J. Zhong, M. Ding, Z. Ma, and M. Chen, “Evaluation of Hallucination and Robustness for Large Language Models,” in 2023 IEEE 23rd International Conference on Software Quality, Reliability, and Security Companion(QRS-C), 2023, pp. 374 – 382. doi: 10.1109/QRS-C60940.2023.00089.

[14] Patrick Lewis et al. “Retrieval-augmented generation for knowledge-intensivenlp tasks”. In: Advances in neural information processing systems 33, 2020, pp. 9459 – 9474. https://dl.acm.org/doi/abs/10.5555/3495724.3496517

Authors

Jeong-Sig Kim(김정식)

2007.2 Ph.D. in Electrical, Electrical and Computer Engineering from Sungkyunkwan University
2012.3-Present : Professor in Gyeonggi University of Science and Technology
<Research Interests> Wireless Sensor Networks, Mobile Computing, Network Security Protocols, Proxy Caching Systems, Artificial Intelligent

Jin-Hong Kim(김진홍)

2006.2 Ph.D. in Electrical, Electrical and Computer Engineering from Sungkyunkwan University
2020.3-Present : Professor in Department of Software, Paichai University
<Research Interests> Big Data, Artificial Intelligent, Intelligent Software, Smart Vehicular Network, Smart Platform