# SAC-KG: Exploiting Large Language Models as Skilled Automatic Constructors for Domain Knowledge Graphs

**Hanzhu Chen[1,2], Xu Shen[2], Qitan Lv[1], Jie Wang[1]*, Xiaoqi Ni[1], Jieping Ye[2]**
[1] CAS Key Laboratory of Technology in GIPAS & MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition, University of Science and Technology of China
[2] Alibaba Cloud
{chenhz, qitanlv, xiaoqi_ni}@mail.ustc.edu.cn, jiewangx@ustc.edu.cn
{shenxu.sx, yejieping.ye}@alibaba-inc.com



## Abstract

Knowledge graphs (KGs) play a pivotal role in knowledge-intensive tasks across specialized domains, where the acquisition of precise and dependable knowledge is crucial. However, existing KG construction methods heavily rely on human intervention to attain qualified KGs, which severely hinders the practical applicability in real-world scenarios. To address this challenge, we propose a general KG construction framework, named SAC-KG, to exploit large language models (LLMs) as **S**killed **A**utomatic **C**onstructors for domain **K**nowledge **G**raph. SAC-KG effectively involves LLMs as domain experts to generate specialized and precise multi-level KGs. Specifically, SAC-KG consists of three components: *Generator*, *Verifier*, and *Pruner*. For a given entity, *Generator* produces its relations and tails from raw domain corpora, to construct a specialized single-level KG. *Verifier* and *Pruner* then work together to ensure precision by correcting generation errors and determining whether newly produced tails require further iteration for the next-level KG. Experiments demonstrate that SAC-KG automatically constructs a domain KG at the scale of over one million nodes and achieves a precision of 89.32%, leading to a superior performance with over 20% increase in precision rate compared to existing state-of-the-art methods for the KG construction task.

## 1 Introduction

Knowledge graphs (KGs) are a collection of factual triples, which represent human knowledge in a structured way, i.e., (head entity, relation, tail entity). In recent years, KGs have been successfully applied in various domains, including medical science (Santos et al., 2022), biology (Zhang et al., 2022), and social networks (Qiu et al., 2018). However, constructing domain KG requires extensive expert knowledge and human intervention, which severely restricts the practical implementation of domain KG construction.

To address this challenge, extensive research efforts have been devoted to the KG construction task (Angeli et al., 2015; Etzioni et al., 2008a). Canonical KG construction methods mainly focus on learning logical rules based on semantic patterns. Rule-based methods extract subject-predicate-object triples by utilizing lexical and semantic role labels (Zhan and Zhao, 2020). Recently, some large language models (LLMs)-based methods have emerged as a new trend and achieved superior performances than rule-based methods (Wang et al., 2021; Han et al., 2023). LLM-based methods extract triples from raw corpora by harnessing the prior knowledge stored within the LLM. Extensive works demonstrate that LLM-based methods are more creative (Swanson et al., 2021) and more human-understandable (Chefer et al., 2021).

Albeit with multiple benefits of the LLM-based methods, they confront two significant challenges that severely hinder their performance and deployment. First, there is **contextual noise** in input. Existing LLM-based methods extract triples directly from the raw context. The raw context includes a substantial amount of domain-irrelevant information, which may potentially distract the LLM and consequently degrade its performance (Shi et al., 2023; Kumar et al., 2021). Second, there is **knowledge hallucination** in output. Knowledge hallucination is that the LLM may generate content that is nonsensical or unfaithful to the provided source content (Zhang et al., 2023; Ji et al., 2023).

Regarding the domain KG construction, the LLM might generate certain irrelevant or incorrect triples due to the contextual noise and knowledge hallucination. Moreover, these incorrect triples may further propagate their errors to the next itera-

* Corresponding author.
This work was done when Hanzhu Chen was an intern at Alibaba Cloud.

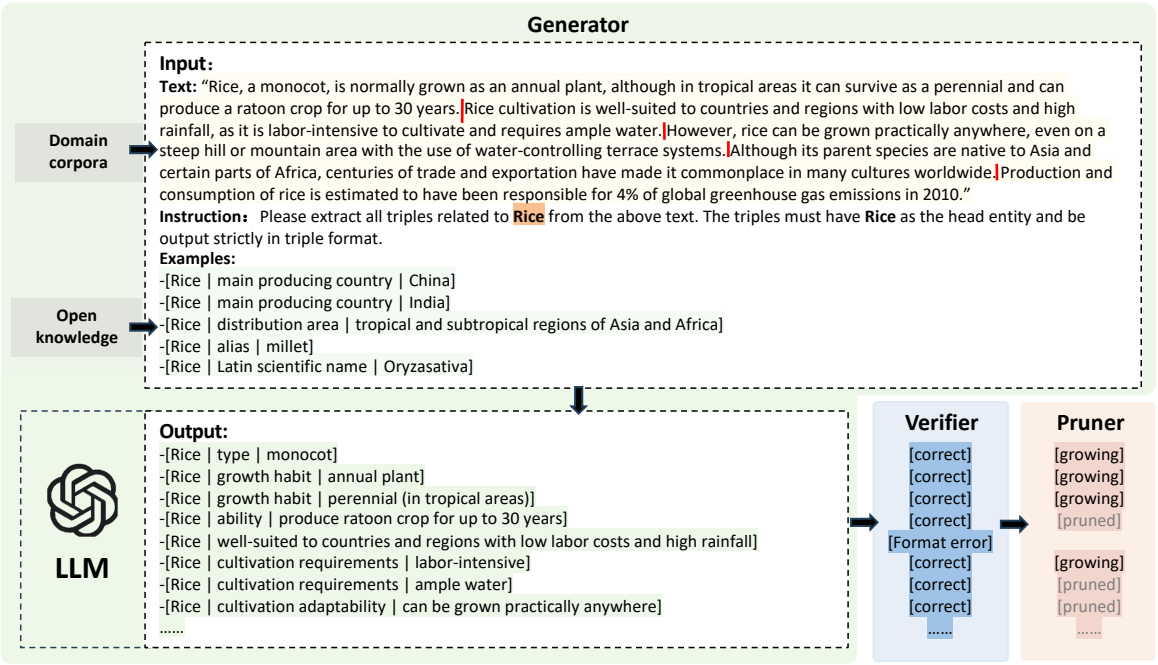



Figure 1: An example of input and output of the SAC-KG framework. Specifically, the input component consists of three segments: text, instruction, and examples. The text segment retrieves the most relevant corpora from a domain-specific corpora for a given entity. The instruction segment provides instructions to an LLM to generate corresponding triples. The example segment retrieves template triples from an open-source encyclopedia KG. The output includes generated correct triples and an indicator of "growing" or "pruned" by *pruner*.

tion, which significantly influences the credibility of the constructed domain KG.

Therefore, in this paper, we seek to answer the question: *Can we propose a general KG construction framework that is automatic, specialized, and precise?* With this consideration, we delve explicitly into the two significant challenges and propose a novel approach, namely SAC-KG, which involves the LLM as domain experts and constructs domain KGs by an entity-induced tree search algorithm automatically and iteratively. SAC-KG is a novel automatic KG construction framework and effectively addresses the issues mentioned above within LLM-based methods. Specifically, SAC-KG comprises three components:

(i) *Generator* applies a *domain corpora retriever* to retrieve the most relevant specialized context from raw domain corpora and an *open knowledge retriever* to retrieve the most relevant triples from an open-source encyclopedic KG, DBpedia, (Xu et al., 2017) for a given entity. Both of them are combined as input to the LLM. *Generator* can eliminate domain-irrelevant information and generate a specialized single-level entity-induced KG.

(ii) *Verifier* applies an error detection process to detect and output error types by employing rule criteria in RuleHub, a repository comprising over 7000 criteria mined from open KGs (Ahmadi et al., 2020) and an error correction process by reprompting the LLM corresponding to the detected error types. *Verifier* alleviates the propagation of error triples and promotes the precision of current-level KG.

(iii) *Pruner* employs a T5 model (Roberts et al., 2019) finetuned on DBpedia, an open-source encyclopedic KG (Xu et al., 2017), as a binary classifier. Pruner takes tail entities of each generated triple as input and determines whether the tail entities need the next-level generation. *Pruner* decides the generating direction, which further improves the precision of constructing the next-level KG.

SAC-KG is a general framework for KG construction with great automation, specialization, and precision. Experiments demonstrate that SAC-KG automatically constructs a domain KG at a scale of over one million nodes and achieves a precision of 89.32% and significantly outperforms existing state-of-the-art methods for the KG construction task, achieving over 20% in precision metric.

## 2 Related Works

**Open Information Extraction.** Open information extraction (OIE) facilitates domain-independent discovery of relational facts from large corpora

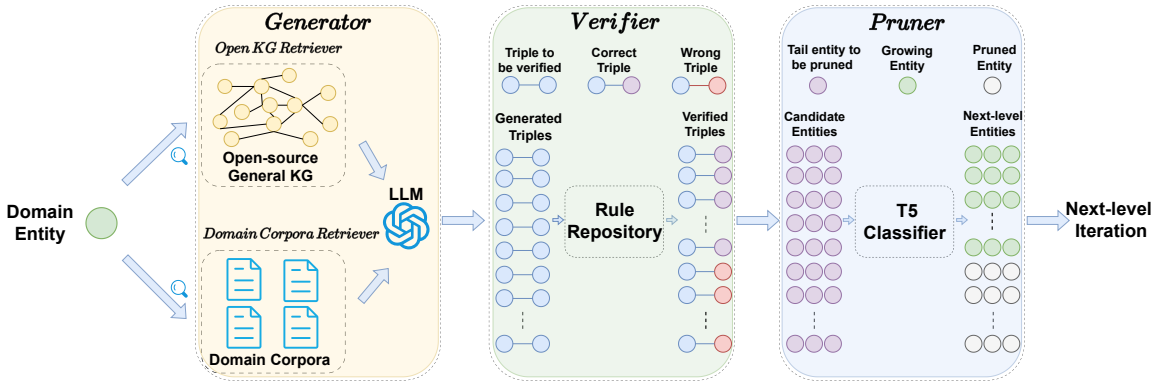



Figure 2: An overview of SAC-KG. SAC-KG organically integrates *Generator*, *Verifier*, and *Pruner* into a unified framework to construct the domain KG automatically. Specifically, for a given entity, SAC-KG iteratively generates a single-level entity-induced knowledge graph (KG). For each iteration, the set of entities designated as "growing" (see green entities in *Pruner*) forms the input for the next-level generation process to the *Generator*.

(Zhou et al., 2022). TextRunner (Etzioni et al., 2008b) is the first OIE model, which merges tuples with identical entities and normalizes relationships based on predefined rules. Following TextRunner, Stanford OIE (Angeli et al., 2015), a popular method for extracting general knowledge from texts, proposes an effective and novel approach to open information extraction, utilizing a classifier to extract self-contained clauses and natural logic inference to determine specific arguments. OIE6 proposes (Kolluru et al., 2020) a novel iterative grid labeling architecture to further improve the extraction quality. More recently, some methods employ LLMs to generate triples directly from input context (Wang et al., 2020; Cohen et al., 2023). Deepex (Wang et al., 2021) leverages the attention matrix of a finetuned pretrained language model to extract triples. PIVE (Han et al., 2023) prompts the LLM and complements additional triples iteratively. However, existing LLM-based methods suffer from both contextual noise and knowledge hallucination to generate high-qualified triples.

**In-context Learning.** In-context learning (ICL), where LLMs make predictions only based on contexts with a few examples, has become a new paradigm for natural language processing (Liu et al., 2021). With the scaling of both model size and training corpora size (Brown et al., 2020), LLMs demonstrate the ability of learning from a few prompts that contain some training examples (Dong et al., 2022a; Kojima et al., 2022). Different from supervised learning requiring a training stage that uses backward gradients to update model parameters, ICL does not need parameter update and directly performs predictions. ICL aims to learn from analogy, which directly LLMs to make predictions (Dong et al., 2022b) with these examples. By concatenating both context and prompt, LLMs learn patterns hidden in examples and perform well on downstream tasks (Kojima et al., 2022).

## 3 Method

We develop a general framework, named SAC-KG, to exploit LLMs (see Appendix A for related works) as skilled automatic constructors for domain KGs. Given domain corpora, the overall task is to extract triples with automation, precision, and controllability. SAC-KG organically integrates *generator*, *verifier*, and *pruner* in a unified framework to perform KG construction. An overview of SAC-KG is shown in Figure 2.

### 3.1 Generator

For a specified entity, which is typically a domain name or a randomly selected set of nouns within that domain, *Generator* employs a *domain corpora retriever* to retrieve the most relevant context from raw domain corpora and an *open knowledge retriever* to retrieve the most relevant triples from an open-source encyclopedic KG, DBpedia (Xu et al., 2017). *Generator* adopts in-context learning, rendering it parameter-free, unsupervised, and fully automatic. Retrievers also contribute to ensuring the quality of generated triples.

#### 3.1.1 Domain corpora Retriever

LLMs are frequently constrained by substantial knowledge hallucinations, where the contents produced by LLMs often diverge from factual knowledge (Dziri et al., 2022; Shuster et al., 2021). The hallucinations may potentially impact the reliability and practical applications of the constructed domain KG. To facilitate accurate knowledge augmentation for the LLM, we propose a *domain cor-*

Table 1: Domain KG evaluation of precision, recall, and domain specificity metrics on the same domain corpora.

| Backbone | Model | Number of recalls | Precision | Domain Specificity |
|---|---|---|---|---|
| Rule-based | OpenIE 6 (2020) | 1.94 | 42.05 | 31.96 |
| Rule-based | Stanford OIE (2023) | 2.12 | 45.94 | 31.24 |
| Bert | DeepEx (2021) | 1.64 | 48.28 | 34.76 |
| ChatGPT | PIVE (2023) | 5.08 | 64.48 | 51.58 |
| Qwen 7B | SAC-KG$_{\text{Qwen}}$ | 3.85 | 69.89 | 57.90 |
| Llama2 7B | SAC-KG$_{\text{Llama2-7B}}$ | 2.73 | 54.39 | 40.59 |
| Llama2 13B | SAC-KG$_{\text{Llama2-13B}}$ | 4.15 | 69.40 | 65.13 |
| ChatGPT | SAC-KG$_{\text{ChatGPT}}$ | **8.09** | **89.32** | **81.25** |

*pora retriever*. For a given entity, it initially segments the domain corpora into sentences and then ranks the relevant sentences based on the frequency of occurrence of that entity. Then, these sentences are concatenated into a text list. Finally, we rank them in descending order of relevance to the given entity, and concatenate them into a fixed-length text as input to the LLM.

检索策略

文本被分割成句子

#### 3.1.2 Open KG Retriever

When the input consists solely of domain-specific corpora and straightforward instructions, the output generated by large language models is often challenging to control and may even exhibit incorrect triple formats. To address this issue, we propose an *open KG retriever*, which adopts the in-context learning (Shin et al., 2022) and retrieves the most related triples associated with the entity from DBpedia (Xu et al., 2017) as examples. These examples encourage the model to generate content in the correct format, which enhances the controllability. We present our retrieval strategy as follows:

主要是为了固定三元组的格式

(i) For entities presented in the open-source KG, we provide related triples wherein the entity serves as the head entity, offering up to 10 cases as examples.

检索策略

(ii) For entities not presented in the open-source KG, we tokenize them and retrieve the most related set of triples. For instance, given the entity "micropropagation", if it is not found within the open-source KG, it will be tokenized into two subentities, "micro" and "propagation", to perform a subsequent retrieval from the open KG again.

没有就拆开，再没有就随机选择

(iii) For entities that remain unmatched even after tokenization, we randomly select ten triples in the KG as prompts.

We then concatenate the related context, the triple prompts, and corresponding instructions as input to the LLM and obtain the extracted triples as output for the *generator*.

### 3.2 Verifier

While the *generator* contributes to enhancing the output quality of the LLM, errors in generated triples still exist. To further enhance the quality of the final generated domain KG, we introduce *verifier*, which is responsible for identifying and filtering out erroneous triples generated by the LLM. *Verifier* is rule-based and parameter-free, enabling efficient execute error detection and correction. Specifically, the *verifier* consists of two steps: **error detection** step and **error correction** step.

For error correction, we use existing criteria mined from open KGs within RuleHub (Ahmadi et al., 2020) to identify errors and output error types. The workflow is as follows.

(i) **Quantity check.** If the number of triples is less than the threshold (default is 3), it will be categorized as "Quantity insufficient".

(ii) **Format Check.** If the triple does not conform to the example format, it will be categorized as "Format error". If head entity does not match the predefined entity, it will be categorized as "Head entity error". If head entity and tail entity are identical, it will be categorized as "Contradiction between head and tail".

(iii) **Conflict Check.** *Verifier* conducts comprehensive conflict detection for each triple in RuleHub (Ahmadi et al., 2020). For instance, ensuring that a person's birth time precedes their time of death and a person's age is not a negative number.

We sequentially conduct quantity, format, and conflict check for generated triples and output information about the error types.

For error correction, we first determine the error type using the error detection step and offer corresponding prompts. Then, we reprompt the LLM to regenerate a corrected output. For instance, if the error type is "format error", we prompt the model with: "Please generate it again strictly according

Table 2: Ablation study for SAC-KG in the first three-level constructed KG. Each iteration implies generating an additional layer of the KG. The symbol "-" denotes that in iteration 1, *pruner* has not been applied before.

| Iteration rounds | Model | Number of recalls | Precision | Domain Specificity |
|---|---|---|---|---|
| **Iteration 1** | SAC-KG$_{\text{w/o prompt}}$ | 10.15 | 80.64 | 74.19 |
| | SAC-KG$_{\text{w/o text}}$ | 11.27 | 81.48 | 71.80 |
| | SAC-KG$_{\text{w/o verifier}}$ | 13.05 | 76.47 | 69.88 |
| | SAC-KG$_{\text{w/o pruner}}$ | - | - | - |
| | SAC-KG | **13.50** | **88.81** | **80.50** |
| **Iteration 2** | SAC-KG$_{\text{w/o prompt}}$ | 4.13 | 77.35 | 71.16 |
| | SAC-KG$_{\text{w/o text}}$ | 7.52 | 63.24 | 52.23 |
| | SAC-KG$_{\text{w/o verifier}}$ | 8.43 | 77.06 | 70.41 |
| | SAC-KG$_{\text{w/o pruner}}$ | 2.30 | 73.33 | 70.42 |
| | SAC-KG | **9.94** | **84.61** | **76.27** |
| **Iteration 3** | SAC-KG$_{\text{w/o prompt}}$ | 3.22 | 61.53 | 56.61 |
| | SAC-KG$_{\text{w/o text}}$ | 5.84 | 38.41 | 31.59 |
| | SAC-KG$_{\text{w/o verifier}}$ | 4.83 | 58.53 | 52.29 |
| | SAC-KG$_{\text{w/o pruner}}$ | 1.32 | 44.82 | 39.65 |
| | SAC-KG | **6.63** | **76.74** | **68.60** |

to the format requirements, paying attention to the format of the example triples." Details of error types and according prompts are in Appendix B.

### 3.3 Pruner

After passing through the *verifier*, we obtain all the correct triples for this level, and then proceed to generate the next-level triples.

However, not all triples need the next-level generation. For instance, the triple "(rice, optimal growth temperature, 20-25 degrees Celsius)" is a correct triple, while its tail entity "20-25 degrees Celsius" does not need to be further generated as the head entity for the next-level triple generation.

Therefore, to enhance the controllability of the constructed KG, we propose *pruner*, a T5 binary classifier model finetuned on an open-source KG, DBpedia. Its input consists of the tail entities from each correct triple. Its output is "growing" or "pruned", indicating whether the entity should proceed to generate the next-level KG or cease further generation. Specifically, we input the text of entities to T5 and it generates "growing" or "pruned" as output. To train the *pruner*, we gather training data from DBpedia and select a subset of head entities to represent the "growing" category. We also gather an equivalent subset of tail entities, excluding those that overlap with the head entity list, to constitute the "pruned" category. We then use these entities text as input and the corresponding labels "growing" or "pruned" as output targets during fine-tuning.

Finally, leveraging domain corpora, we can produce a single-level KG for the input entity, which will subsequently be incorporated into a new level of generated KG. Hence, SAC-KG generates multiple triples with the entity and proceed to iterate, creating a KG subtree rooted in head entities of the generated triples. This process resembles the incremental growth of a tree layer by layer, akin to retrieving and accessing domain knowledge from shallow to deep. Furthermore, SAC-KG is an unsupervised approach that can be applied to any domain with significant volumes of unstructured text corpora, without the need for labeled data.

## 4 Experiments

We design experiments to evaluate the effectiveness of the proposed SAC-KG and provide more insights of the constructed domain KG. With this desiderata, we divide the experiments into five parts:

(i) To evaluate the effectiveness of SAC-KG, we compare SAC-KG with existing state-of-the-art methods for domain KG construction task.

(ii) To offer a more comprehensive evaluation of constructed KG, we conduct agreement evaluation between GPT4 and humans.

(iii) To provide more insight into SAC-KG, we conduct the ablation study of each component.

(iv) To analyze the constructed KG, we conduct case study of the constructed domain KG.

(v) To further demonstrate the effectiveness of SAC-KG, we evaluate SAC-KG on existing

Table 3: Case study results for different categories. For entities in the categories, we evaluate their single-level KGs and report the mean results.

| Entity category | Model | Number of recalls | Precision | Domain Specificity |
|---|---|---|---|---|
| **Rice variety** | OpenIE 6 (2020) | 1.90 | 31.65 | 24.05 |
| | Stanford OIE (2023) | 2.33 | 39.28 | 26.71 |
| | DeepEx (2021) | 3.04 | 60.37 | 43.47 |
| | PIVE (2023) | 2.57 | 54.48 | 43.58 |
| | SAC-KG$_{\text{ChatGPT}}$ | **13.11** | **84.28** | **76.88** |
| **Rice expert** | OpenIE 6 (2020) | **7.75** | 50.40 | 38.30 |
| | Stanford OIE (2023) | 4.25 | 43.03 | 29.26 |
| | DeepEx (2021) | 1.50 | 47.36 | 34.01 |
| | PIVE (2023) | 2.00 | 55.17 | 44.14 |
| | SAC-KG$_{\text{ChatGPT}}$ | 3.88 | **93.33** | **84.43** |

open-source OIE benchmarks.

### 4.1 Datasets and Experiment Setup

We initially collect raw textual data from specialized books, web pages, and genealogical data to the rice domain. In total, we collect 70 specialized books, 1522 web pages, and 24000 genealogical records related to rice (see Appendix H for details). These domain corpora exhibit varying degrees of structural diversity and different levels of textual quality, which can also effectively emulate the conditions encountered in the majority of original corpora within other domains.

We retrieve domain entities as root node from the open-source KG and obtain their domain texts from domain corpora. We retrieve up to 500 tokens of domain text for each node. We then compare the extraction of triples based on the same input text by different baselines. We assess performance by using the following metrics.

**Precision:** To assess precision, we conduct evaluations through both manual and automatic manners, with the latter being more scalable in nature. Following Vicuna (Zheng et al., 2023), we employ GPT-4 (OpenAI, 2023) as an automatic judge. Specifically, we take extracted triples with their corresponding text as input to the GPT-4 for assessing the correctness of each triple.

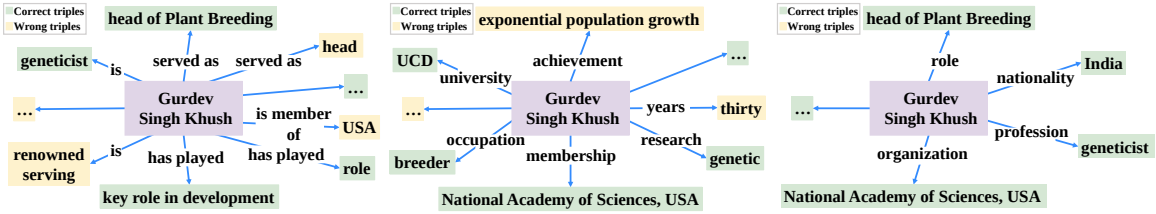
**Recall:** Estimating recall is infeasible due to the inability to access the ground truth triples for each domain text. Therefore, we report the average count of verified triples for each domain text. That is, we report recall without providing the denominator. We refer to this metric by the **number of recalls**. As in (Vo and Bagheri, 2016; Kolluru et al., 2020), this metric aligns with the real-world scenarios, where it is impractical to obtain the entire set of accurate facts. Consequently, the convention is to report only the count of generated facts. This metric serves as an indicator of the effective extraction and utilization of domain corpora.

**Domain Specificity:** We aim to generate triples that are correct, domain-related, and distinct from those triples in the open-source encyclopedic KG. Specifically, inspired by the survey (Wang et al., 2023), we aim to ensure the construction of a large-scale domain KG with higher domain expertise. To this end, we introduce a domain-specific metric that quantifies the proportion of generated triples meeting three criteria: correctness, domain-related, and not presented in the open-source encyclopedic KG. This metric is computed as |set of generated domain-related and correct triples $-$ set of triples in the open-source KG| / |set of generated triples|, where "$-$" denotes the set difference operation, and "||" represents the cardinality of a set. The primary objective of domain specificity is encouraging the LLM to extract knowledge not solely reliant on the open-source KG but also capable of summarizing and condensing domain knowledge from the domain corpora.

For the parameter set up of *generator*, we set temperature value of the LLM as 0.1 and a maximum sequence length of 2000 tokens. For *pruner*, we use the low-rank adaptation (Hu et al., 2021) to efficiently finetune a T5 (Roberts et al., 2019) model. We train the model with 2 epochs and use batch size of 64. We set the learning rate as 0.001. More details can be found in Appendix E.

### 4.2 Main Results

We employ four baseline models for our study, namely OpenIE 6 (Kolluru et al., 2020), Stanford OIE (Angeli et al., 2015), DeepEx (Wang et al.,




(a) Case study for OIE6. (b) Case study for PIVE. (c) Case study for SAC-KG.

Figure 3: Visualization results of rice expert case of OIE6, PIVE, and SAC-KG. Entities marked in green denote the correct triples and entities marked in yellow denote the wrong triples.

Table 4: Statistical indicators of GPT4 evaluation and human evaluation.

| Precision | Recall | F1 score | Cohen's Kappa coefficient |
|---|---|---|---|
| 0.906 | 0.951 | 0.928 | 0.613 |

2021), and PIVE (Han et al., 2023). OpenIE 6 and Stanford OIE represent state-of-the-art methods for rule-based triple extraction, with the Stanford OIE method being based on the updated version released in September 2023. DeepEx is a representative approach that combines Bert (Devlin et al., 2018) with rule-based techniques for triple extraction, while PIVE directly utilizes ChatGPT (OpenAI, 2020) to construct KGs.

To comprehensively evaluate the performance of our approach, we utilize multiple LLMs as backbones. We mainly use ChatGPT (OpenAI, 2020), which is widely recognized for its superior performance and serves as the foundation for many research endeavors. Furthermore, to demonstrate the effectiveness of our method on models with smaller parameter sizes, we also select Qwen 7B, Llama2 7B, and Llama2 13B (Bai et al., 2023; Touvron et al., 2023) as our backbones.

As shown in Table 1, SAC-KG outperforms previous methods in KG construction consistently. Rule-based approaches like OIE6 and Stanford OIE, which extract triples through lexical and semantic role labels, exhibit poor performance for precision and domain specificity metrics. We also observe that rule-based approaches tend to extract uninformative triples, leading to a falsely inflated recall rate (see Section 4.5 for details). DeepEx and PIVE, which utilise LMs as backbones, show some improvement but still also perform suboptimally. When utilizing the ChatGPT as the backbone, we achieve an precision rate of $89.32\%$ and domain specificity of $81.25\%$. This further demonstrates the effectiveness of our approach in the direct construction of KGs from open domain corpora.

### 4.3 Agreement Evaluation

We use GPT-4 for automatic and efficient evaluation. To demonstrate the validity of this approach, we conduct a human evaluation. Specifically, we engage 20 volunteers, comprising 5 PhDs, 7 PhD candidates, and 8 master students with KGs and rice backgrounds. We make evaluation questionnaires, each with 100 "text-triple list" pairs (input texts from the model and corresponding output triple lists). Volunteers assess triple correctness and error reasons (e.g., text inconsistency, formatting, or others) when marking a triple as incorrect.

We provide key statistical indicators for more trustworthy results. We use human evaluation results as the ground truth for precision, recall, F1 score, and average precision. As shown in Table 4, the results indicate a close alignment between GPT4 evaluation and human evaluation. With a precision value of $0.906$, GPT4 identifies most positive samples. The recall value of $0.951$ shows that it can capture most true positives. The F1 score of $0.928$ also shows the solidity of GPT4 evaluation. Moreover, Cohen's Kappa coefficient above $0.6$ suggests a medium to high consistency between GPT4 evaluation and human evaluation. Overall, these statistics demonstrate the effectiveness and dependability of GPT4 evaluation. More details of GPT and human evaluation are in Appendix C.

### 4.4 Ablation Study

To further investigate the contribution of each component within SAC-KG to the KG construction, we conduct a series of ablation experiments on the entire framework. We compute these metrics in each iteration to obtain a fine-grained result. Specifically, we denote SAC-KG without the *open KG retriever* as SAC-KG$_{w/o\ prompt}$, SAC-KG without the *domain corpora retriever* as SAC-KG$_{w/o\ text}$, SAC-KG without the *verifier* as SAC-KG$_{w/o\ verifier}$, and SAC-KG without the *pruner* as SAC-KG$_{w/o\ pruner}$, respectively.

Table 5: F1 score and AUC results on OIE2016, WEB, NYT, and PENN datasets.

| Model | OIE2016 | | WEB | | NYT | | PENN | |
|---|---|---|---|---|---|---|---|---|
| | F1 | AUC | F1 | AUC | F1 | AUC | F1 | AUC |
| OpenIE 6 (2020) | 55.3 | 61.1 | 61.1 | 64.9 | 30.7 | 55.2 | 54.2 | 63.1 |
| Stanford OIE (2023) | 59.3 | 65.1 | 63.3 | 69.3 | 31.1 | 56.9 | 56.8 | 67.1 |
| DeepEx (2021) | 72.6 | 58.6 | 91.2 | 82.4 | 85.5 | 72.5 | 88.5 | 81.5 |
| PIVE (2023) | 70.4 | 71.1 | 89.8 | 86.0 | 83.5 | 77.9 | 86.0 | 81.2 |
| SAC-KG$_{\text{ChatGPT}}$ | **74.7** | **73.2** | **96.6** | **95.7** | **88.8** | **87.3** | **91.1** | **90.1** |

As shown in Table 2, the absence of any component within SAC-KG results in a performance degradation of the entire framework. Notably, the *pruner* and the *open KG retriever* have a more pronounced impact on the performance of SAC-KG. These two components control generating direction and adding examples, respectively. This implies the importance of enhancing controllability in the KG construction process.

### 4.5 Case Study

We conduct a case study on the KGs constructed by SAC-KG and the baselines. Specifically, we select rice varieties and rice experts as two cases to analyze the distinctions between different constructed KGs. More cases are in Appendix F. As illustrated in Table 3, each iteration of SAC-KG demonstrates favorable results in terms of precision and domain specificity. While in rice expert case, rule-based approaches achieve higher recall rates but exhibit suboptimal precision and domain specificity. This may be attributed to the heightened sensitivity to personal name entities of rule-based methods (Kolluru et al., 2020). However, their precision and domain specificity do not demonstrate satisfactory performance. On the contrary, SAC-KG exhibits higher precision and domain specificity in this case, albeit with lower recall rates.

We visualize three single-level KGs in the rice expert case to gain a further insight. As Figure 3 shows, the rule-based approaches (OIE6) tend to generate redundant triples simply through lexical and syntactical analysis. These triples often contain limited specific information. PIVE extracts more informative triples, while it is still affected by irrelevant textual noise and extracted incorrect triples such as "(Gurdev Singh Khush, achievement, exponential population growth)". SAC-KG, while extracting a reduced number of triples, produces triples that possess a higher degree of human interpretability and domain information. Therefore, improving recall rates in specific cases of SAC-KG to increase the utilization of domain corpora information will be a focus of our future research.

### 4.6 Results on OIE benchmarks

To further demonstrate the effectiveness and generality of our SAC-KG, we conduct experiments on open-source benchmarks for traditional OIE tasks. Following the setting and the evaluation method of DeepEx (Wang et al., 2021), we evaluate the OIE2016 (Stanovsky and Dagan, 2016), NYT, WEB (Mesquita et al., 2013), and PENN (Radford et al., 2021) datasets and use traditional AUC and F1 score as metrics. Details of the datasets are summarized in Appendix G.

As shown in Table 5, SAC-KG also outperforms existing state-of-the-art methods across traditional OIE benchmarks, which demonstrate its effectiveness and generalization. Specifically, SAC-KG outperforms rule-based methods (OpenIE 6 and Stanford OIE) by a large margin. And compared with LLM-based methods (DeepEx and PIVE) SAC-KG also attain the optimal results consistently, which demonstrates the effectiveness and robustness of SAC-KG. These results also show the effectiveness of SAC-KG in the traditional OIE task.

## 5 Conclusion

In this paper, we propose a novel automatic domain KG construction framework named SAC-KG, which effectively constructs KG directly from domain corpora. SAC-KG incorporates LLMs as domain experts and iteratively employs an entity-induced tree search algorithm for the construction of a multi-level KG. Specifically, we propose *Generator*, *Verifier*, and *Pruner* to form a general KG construction framework with automation, precision, and controllability. SAC-KG constructs a domain KG at the scale of over a million nodes with an precision of 89.32%, achieving over 20% increase in precision metric. This superior performance of SAC-KG over existing state-of-the-art methods demonstrates effectiveness of our SAC-KG.

## Acknowledgements

This work was supported in part by National Key R&D Program of China under contract 2022ZD0119801, National Nature Science Foundations of China grants U23A20388, 62021001, U19B2026, and U19B2044. We would like to thank all the anonymous reviewers for their insightful comments.

## Limitations

While SAC-KG can construct domain-specific KGs, it cannot inject or update the domain knowledge into LLMs. Exploring low-cost methods to inject domain knowledge into LLMs for the creation of a domain-specific LLMs will be the focus of our future work. We will also focus on employing this approach as a means to explicitly interpret the learned knowledge of LLMs.

## References

Naser Ahmadi, Thi-Thuy-Duyen Truong, Le-Hong-Mai Dao, Stefano Ortona, and Paolo Papotti. 2020. Rulehub: A public corpus of rules for knowledge graphs. *Journal of Data and Information Quality (JDIQ)*, 12(4):1–22.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. 2021 ieee. In *CVF International Conference on Computer Vision (ICCV)*, pages 387–396.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Roi Cohen, Mor Geva, Jonathan Berant, and Amir Globerson. 2023. Crawling the internal knowledge-base of language models. *arXiv preprint arXiv:2301.12810*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022a. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022b. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Nouha Dziri, Sivan Milton, Mo Yu, Osmar Zaiane, and Siva Reddy. 2022. On the origin of hallucinations in conversational models: Is it the datasets or the models?

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008a. Open information extraction from the web. *Commun. ACM*, 51(12):68–74.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S Weld. 2008b. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training.

Jiuzhou Han, Nigel Collier, Wray Buntine, and Ehsan Shareghi. 2023. Pive: Prompting with iterative verification improving graph-based generative capability of llms. *arXiv preprint arXiv:2305.12392*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model.

Shibo Hao, Bowen Tan, Kaiwen Tang, Bin Ni, Xiyan Shao, Hengzhe Zhang, Eric P. Xing, and Zhiting Hu. 2023. Bertnet: Harvesting knowledge graphs with arbitrary relations from pretrained language models.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 643–653.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Keshav Kolluru, Vaibhav Adlakha, Samarth Aggarwal, Soumen Chakrabarti, et al. 2020. Openie6: Iterative grid labeling and coordination analysis for open information extraction. *arXiv preprint arXiv:2010.03147*.

Vivek Kumar, Rishabh Maheshwary, and Vikram Pudi. 2021. Adversarial examples for evaluating math word problem solvers.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 447–457.

OpenAI. 2020. Chatgpt: A large-scale generative model for conversation.

OpenAI. 2023. Gpt-4 technical report.

Jiezhong Qiu, Jian Tang, Hao Ma, Yuxiao Dong, Kuansan Wang, and Jie Tang. 2018. Deepinf. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery; Data Mining*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training.

Adam Roberts, Colin Raffel, Katherine Lee, Michael Matena, Noam Shazeer, Peter J Liu, Sharan Narang, Wei Li, and Yanqi Zhou. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Alberto Santos, Ana R Colaço, Annelaura B Nielsen, Lili Niu, Maximilian Strauss, Philipp E Geyer, Fabian Coscia, Nicolai J Wewer Albrechtsen, Filip Mundt, Lars Juhl Jensen, et al. 2022. A knowledge graph to interpret clinical proteomics data. *Nature biotechnology*, 40(5):692–702.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H Chi, Nathanael Schärli, and Denny Zhou. 2023. Large language models can be easily distracted by irrelevant context. In *International Conference on Machine Learning*, pages 31210–31227. PMLR.

Seongjin Shin, Sang-Woo Lee, Hwijeen Ahn, Sungdong Kim, HyoungSeok Kim, Boseop Kim, Kyunghyun Cho, Gichang Lee, Woomyoung Park, Jung-Woo Ha, and Nako Sung. 2022. On the effect of pretraining corpora on in-context learning by a large-scale language model.

Mohammad Shoeybi, Md.MostofaAli Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. 2019. Megatron-lm: Training multi-billion parameter language models using model parallelism. *Cornell University - arXiv,Cornell University - arXiv*.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation.

Gabriel Stanovsky and Ido Dagan. 2016. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2300–2305.

Ben Swanson, Kory Mathewson, Ben Pietrzak, Sherol Chen, and Monica Dinalescu. 2021. Story centaur: Large language model few shot learning as a creative writing tool. In *Proceedings of the 16th Conference of the European Chapter of the Association for*

*Computational Linguistics: System Demonstrations*, pages 244–256.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Duc-Thuan Vo and Ebrahim Bagheri. 2016. Open information extraction. *World Scientific Encyclopedia with semantic computing and robotic intelligence,World Scientific Encyclopedia with semantic computing and robotic intelligence*.

Chenguang Wang, Xiao Liu, Zui Chen, Haoyun Hong, Jie Tang, and Dawn Song. 2021. Zero-shot information extraction as a unified text-to-triple translation. *arXiv preprint arXiv:2109.11171*.

Chenguang Wang, Xiao Liu, and Dawn Song. 2020. Language models are open knowledge graphs. *arXiv preprint arXiv:2010.11967*.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, et al. 2023. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *arXiv preprint arXiv:2310.07521*.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small.

Bo Xu, Yong Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Wanyun Cui, and Yanghua Xiao. 2017. Cndbpedia: A never-ending chinese knowledge extraction system. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 428–438. Springer.

Junlang Zhan and Hai Zhao. 2020. Span model for open information extraction on accurate corpus. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9523–9530.

Ningyu Zhang, Zhen Bi, Xiaozhuan Liang, Siyuan Cheng, Haosen Hong, Shumin Deng, Jiazhang Lian, Qiang Zhang, and Huajun Chen. 2022. Ontoprotein: Protein pretraining with gene ontology embedding. *arXiv preprint arXiv:2201.11147*.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*.

Shaowen Zhou, Bowen Yu, Aixin Sun, Cheng Long, Jingyang Li, Haiyang Yu, Jian Sun, and Yongbin Li. 2022. A survey on neural open information extraction: Current status and future directions. *arXiv preprint arXiv:2205.11725*.

## A More Related Works

**Language Models.** Language models including GPT (Radford et al.), BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), and Megatron-LM (Shoeybi et al., 2019) have led to a learning paradigm shift in natural language processing (NLP). Models are first pre-trained on extensive volumes of unlabeled text corpora with language modeling objectives, and then fine-tuned on downstream tasks. Recently, large language models (LLMs) including ChatGPT (OpenAI, 2020) and PaLM (Chowdhery et al., 2022) have shown great performance in both few-shot and even zero-shot scenarios (Brown et al., 2020). To further enhance the interpretability of these LLMs, some research endeavors explain LLMs through attribution analysis (Wang et al., 2022; Hanna et al., 2023; Gurnee et al., 2023). Another line of work aims to retrieve the knowledge explicitly from LLMs as the basis for interpreting them, including the reasoning task (Shi et al., 2023) and the QA task (Hao et al., 2023; Dhingra et al., 2020; Guu et al., 2020).

## B Details of Error Types and Prompts

Our approach involves performing error correction based on the types of errors output by the error detection module. To further enhance inference efficiency, if the number of categorized triples exceeds a predefined threshold (the default being 3), the *verifier* will amalgamate the corresponding prompts from Table 6 with the original input to the LLMs for regeneration. Conversely, if the number of categorized triples is below this threshold, the *verifier* will directly eliminate the marked triples, obviating the need for regeneration.

## C More Details of Agreement Evaluation

We conduct a human evaluation by engaging several doctoral candidates with expertise in rice research. Specifically, we involve the validation of each generated fact by manual examination of highly reliable web sources such as Wikipedia to verify the precision of each fact. Moreover, we provide initial correct/incorrect triple examples for guidance. We set a 5-second minimum evaluation time per triple. For a 10-triple pair, volunteers need spend 50 seconds before the next evaluation. We also highlight entities and relations in text when they appear in triples.

Following Vicuna(Zheng et al., 2023), we report agreement evaluation to demonstrate the rationality of our automatic evaluation. In Table 7, we compare the results of different evaluation methods for the same generated KG, where 'GPT-4' denotes the automatic evaluation in our main text, 'Author' denotes the results of evaluation by the authors, 'Humen' denotes the results of evaluation by domain experts, and 'Humen-M' denotes the majority judgment of humen. Moreover, we report the agreement between two types of judges on GPT-4, Author, Human, and Human-M in Table 8. The agreement between two types of judges as the probability of each type agreeing on questions, i.e., whether a given triple is correct or incorrect.

We engage volunteers with background related to KGs and rice, because they possess basic domain knowledge, which enables them to more accurately assess the quality of the generated domain knowledge graphs. Moreover, these volunteers come from a variety of universities and research institutions to enhance objectivity in evaluation.

## D Visualization of Ablation Study

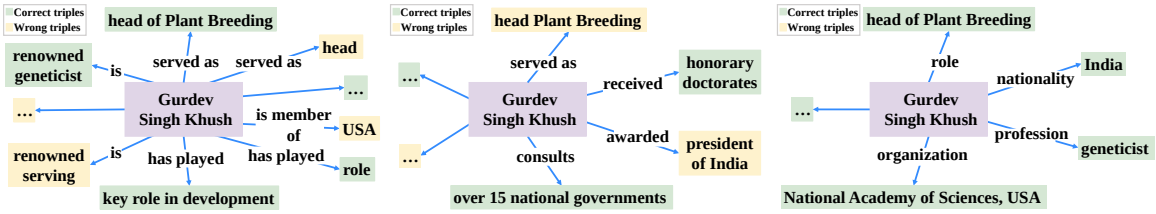
We further visualize the first three-level generated KG of each ablated version of SAC-KG. As Figure 5 shows, the full version of SAC-KG exhibits the overall best result, and the number of error triples in each level do not exhibit significant differences. This phenomenon reveals that error propagation is not notable in the iterative generation of the domain KG. On the contrary, SAC-KG$_{w/o\ text}$ and SAC-KG$_{w/o\ pruner}$ exhibit error propagation, which leads to a significant increase of error triples generated in the third layer. SAC-KG$_{w/o\ prompt}$ and SAC-KG$_{w/o\ verifier}$ only extract fewer triples, which means the LLM suffers from summarizing knowledge in domain corpora without examples and error correction process. These results further affirm that each component within the framework contributes significantly to the construction.

## E Details of Experiment Setup

We provide parameter settings for the mentioned large language models. For all large language models, We set the temperature hyperparameter that controls the output stability of LLMs as 0.1. The lower the temperature setting, the more stable the model output is. We set the max input length as 500 tokens, which represents the maximum token length input to the model is 500 tokens per response. We set the max length of retrieved text as 2000, which represents the maximum length of

Table 6: Details of the error types and according prompts

| Error type | Prompt |
|---|---|
| General conflict | Please generate it again strictly according to the requirements. |
| Quantity too small | Please generate it again strictly according to the requirements, and pay attention to generating sufficient triples. |
| Head entity error | Please generate it again strictly according to the requirements, and note that the head entity must be xx |
| Format error | Please generate it again strictly according to the format requirements, paying attention to the format of the example triples. |
| Contradiction between head and tail | Please generate it again strictly according to the format and requirements, and note that the head and tail entities are generally inconsistent. |




(a) Case study for Stanford OIE. (b) Case study for Deepex. (c) Case study for SAC-KG.

Figure 4: Visualization results of rice expert case of Stanford OIE, Deepex, and SAC-KG. Entities marked in green denote the correct triples and entities marked in yellow denote the wrong triples.

Table 7: Different types of judges on GPT-4, Author, Human, and Human-M.

| Judge | Number of recalls | Precision | Domain Specificity |
|---|---|---|---|
| GPT-4 | 8.09 | 89.32 | 88.31 |
| Author | 7.88 | 87.09 | 86.85 |
| Human | 7.80 | 86.15 | 85.96 |
| Human-M | 7.93 | 87.55 | 87.35 |

Table 8: Agreement between two types of judges on GPT-4, Author, Human, and Human-M.

| Judge | GPT-4 | Author | Human | Human-M |
|---|---|---|---|---|
| GPT-4 | - | 86.65 | 84.78 | 87.11 |
| Author | - | - | 82.67 | 89.69 |
| Human | - | - | - | 92.97 |
| Human-M | - | - | - | - |

the domain corpus text returned by domain corpus retriever is 2000 tokens. For *pruner*, we apply the Low-Rank Adaptation (LORA) (Hu et al., 2021) to efficiently finetune a T5 (Roberts et al., 2019) model on the open-source KG. We train the model with 2 epochs and use batch size of 64. We set the learning rate as 0.001. Moreover, we randomly sampled 120 domain-related entities from the open source graph as root node input *generator*.

## F More Case Study Results

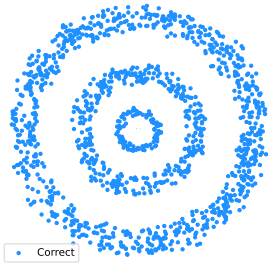
We visualize the rice experts case results (mentioned in Section 4.5) of Stanford OIE and Deepex in Figure 4. SAK-KG also performs better and produces more precise and domain-aware triples.

We also present more case study results in Table 10. We also observe that in rice disease, rice pest and rice variety cases, SAC-KG also outperforms all mentioned baselines in all three metrics by a large margin. This also demonstrates the effectiveness of our SAC-KG.

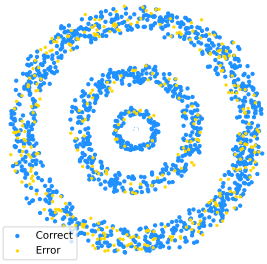
We further visualize the three case study results in Figure 6. Three single-level KGs generated by SAC-KG in these three cases are also precise and human understandable, which demonstrates the effectiveness of our method.

Table 9: Statistics of OIE benchmark datasets.

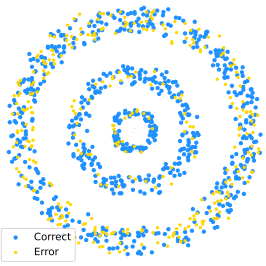
| Dataset | Domain | #Sents | #Triples | | |
|---|---|---|---|---|---|
| | | | Train | Dev | Test |
| OIE2016 | News, Wiki | 3,200 | 5,078 | 1,673 | 1,730 |
| WEB | News, Web | 500 | - | - | 461 |
| NYT | News, Wiki | 222 | - | - | 150 |
| PENN | Mixed | 100 | - | - | 52 |



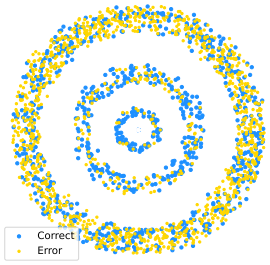

(a) All correct triples extracted from the full version of SAC-KG.



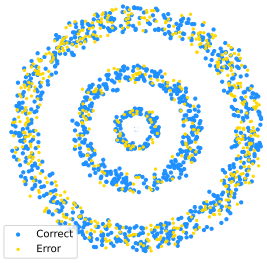

(b) Triples generated by the full version of SAC-KG.



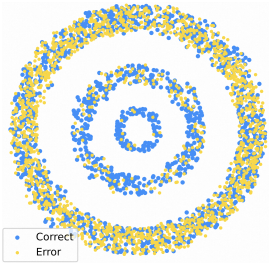

(c) Triples generated by SAC-KG$_{w/o\ prompt}$.




(d) Triples generated by SAC-KG$_{w/o\ text}$.




(e) Triples generated by SAC-KG$_{w/o\ verifier}$.




(f) Triples generated by SAC-KG$_{w/o\ pruner}$.

Figure 5: Visualization of the first three-level constructed KG by full version and ablated versions of SAC-KG. The radius of each concentric circles denotes levels of each generated levels. Nodes marked in blue denote the correct triples and nodes marked in yellow denote the wrong triples.

## G Details of OIE benchmarks

We assess the performance of open information extraction (OIE) systems using benchmark datasets that include OIE2016 (Stanovsky and Dagan, 2016), derived from Newswire and Wikipedia through automatic conversion from QA-SRL (He et al., 2015). Moreover, we also incorporate NYT, WEB (Mesquita et al., 2013), and PENN (Radford et al., 2021). A summary of the benchmark dataset statistics is presented in Table 9.

## H Scientific Artifacts

The data we collect in specialized domains is publicly available and viewable online. The data owners have indicated that the data can be used for scientific research or have not indicated that the data cannot be used for scientific research, and our collection process is also in compliance with regulations. Moreover, there is no unique identification of individuals or offensive content in these data.

We provide details on collecting data. First, we search on Google for relevant domain books using the keyword "rice" and downloaded a total of 70 books. Second, we search for relevant web pages on Google using the same keyword and gather text from a total of 1522 pages. Third, we collect 24000 genealogical data from relevant rice genealogical databases, which included information about each type of rice and its parent data. Subsequently, we perform simple data cleaning on this data, which involved removing HTML tags, images, tables, and special meaningless characters.

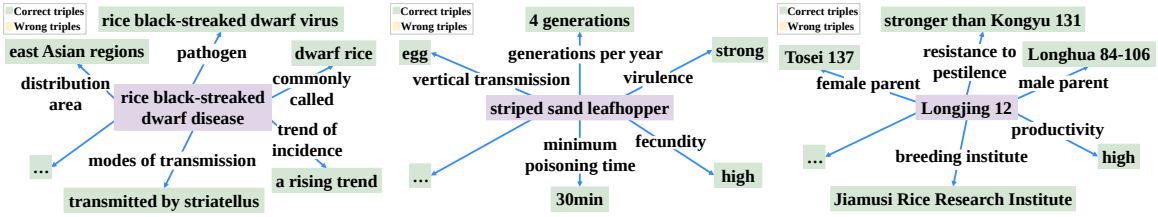
## I More discussions on SAC-KG

### I.1 Could the random return of a set of triples in cases where no relevant triples are retrieved potentially detrimentally affect performance or accuracy?

**A1**: It does not detrimentally impact performance. In experiments, we observe that the more relevant the triple prompts, the stronger the prompting effect. In instances where relevant triples cannot be retrieved, randomly returning a set of triples or returning a fixed set (both perform equally) still enhances performance compared to not returning any triples.

### I.2 If there are still incorrect triples after passing through the *verifier*, does the *verifier* fail to detect them?

**A2**: To enhance reasoning efficiency, the *verifier* presently employs the rule-based techniques for




(a) Rice disease case study for SAC-KG.

(b) Rice pest case study for SAC-KG.

(c) Rice variety case study for SAC-KG.

Figure 6: Visualization results of rice disease, rice pest, and rice variety case study of SAC-KG. Entities marked in green denote the correct triples.

format and conflict assessments, resorting to re-prompting the LLM solely upon the detection of errors. Furthermore, extensive empirical findings reveal that the utilization of domain text and triple prompts diminish the likelihood of encountering factual inaccuracies (e.g., erroneous triples such as "United States, Capital, New York"), with the majority of errors being concentrated in formatting or conflicts. Nevertheless, it is worth noting that future research could explore the possibility of enabling the model to autonomously perform verification, which would substantially increase the computational cost of reasoning.

### I.3 Are all the triples provided as input to the *pruner* correct?

**A3**: Not necessarily all of them are correct. The preceding *verifier* strives to ensure the correctness of triples while maintaining high efficiency. However, the accuracy of triples does not significantly impact the subsequent generation process, as the quality of the tail entity primarily dictates the quality of the subsequent generation, irrespective of triple correctness. Consequently, the objective of the *pruner* is to eliminate low-quality tail entities.

### I.4 Why is the *pruner* trained with DBpedia able to yield favorable results in the domain of rice?

**A4**: Indeed, this is intuitive because determining whether an entity can function as a head entity does not necessitate an extensive domain-specific knowledge. It primarily involves learning the pattern distinctions between head and tail entities. Additionally, DBpedia encompasses fundamental nouns pertinent to the domain of rice. For instance, in the case of tail entities such as "**glutinous rice**", "**rice blast disease**", "33 acres", "3-4 days of concentrated irrigation", and "lack of nitrogen fertilizer", it is relatively straightforward to discern that those entities marked in **bold** can be utilized as head entities, implying their association with meaningful individual objects.

### I.5 Does the open-source KG incorporate domain entities?

**A5**: As previously stated, open-source KGs like DBpedia encompass fundamental nouns related to the domain of rice. They encompass basic rice varieties, rice-related diseases and pests, as well as planting methods. Nevertheless, more specialized terminology may not be encompassed. Consequently, one approach is to identify common domain entities from the open-source knowledge graph and employ them as a foundation for constructing the KG. Subsequently, SAC-KG can be employed to guide the incremental extraction of "specialized knowledge from the corpus, layer by layer".

Table 10: More case study results for different categories. For entities in the categories, we evaluate their single-level KGs and report the mean results.

| Entity category | Model | Number of recalls | Precision | Domain Specificity |
|---|---|---|---|---|
| **Rice variety** | OpenIE 6 (2020) | 1.90 | 31.65 | 24.05 |
| | Stanford OIE (2023) | 2.33 | 39.28 | 26.71 |
| | DeepEx (2021) | 3.04 | 60.37 | 43.47 |
| | PIVE (2023) | 2.57 | 54.48 | 43.58 |
| | SAC-KG$_{\mathrm{Qwen}}$ | 4.15 | 69.89 | 61.05 |
| | SAC-KG$_{\mathrm{Llama2-7B}}$ | 2.27 | 49.34 | 35.26 |
| | SAC-KG$_{\mathrm{Llama2-13B}}$ | 3.60 | 59.79 | 55.96 |
| | SAC-KG$_{\mathrm{ChatGPT}}$ | **13.11** | **84.28** | **76.88** |
| **Rice disease** | OpenIE 6 (2020) | 2.77 | 41.66 | 31.67 |
| | Stanford OpenIE (2023) | 3.44 | 57.40 | 39.03 |
| | DeepEx (2021) | 1.66 | 45.45 | 35.45 |
| | PIVE (2023) | 1.07 | 81.51 | 62.21 |
| | SAC-KG$_{\mathrm{Qwen}}$ | 3.55 | **94.11** | 82.82 |
| | SAC-KG$_{\mathrm{LLaMa-7B}}$ | 1.56 | 63.64 | 47.27 |
| | SAC-KG$_{\mathrm{LLaMa-13B}}$ | **5.44** | 80.32 | 75.53 |
| | SAC-KG$_{\mathrm{ChatGPT}}$ | 4.11 | 93.67 | **85.01** |
| **Rice pest** | OpenIE 6 (2020) | 1.26 | 62.26 | 47.32 |
| | Stanford OIE (2023) | 2.11 | 64.70 | 44.00 |
| | DeepEx (2021) | 2.91 | 68.62 | 49.41 |
| | PIVE (2023) | 8.65 | 64.84 | 51.87 |
| | SAC-KG$_{\mathrm{Qwen}}$ | 3.50 | 64.53 | 45.56 |
| | SAC-KG$_{\mathrm{Llama2-7B}}$ | 2.77 | 45.86 | 35.16 |
| | SAC-KG$_{\mathrm{Llama2-13B}}$ | 5.34 | 76.79 | 71.60 |
| | SAC-KG$_{\mathrm{ChatGPT}}$ | **14.44** | **89.96** | **80.54** |
| **Rice pesticide** | OpenIE 6 (2020) | 1.90 | 48.83 | 37.11 |
| | Stanford OpenIE (2023) | 2.09 | 60.52 | 41.15 |
| | DeepEx (2021) | 2.5 | 45.45 | 32.72 |
| | PIVE (2023) | 3.18 | 61.40 | 49.12 |
| | SAC-KG$_{\mathrm{Qwen}}$ | 3.0 | 80.48 | 70.82 |
| | SAC-KG$_{\mathrm{LLaMa-7B}}$ | 2.64 | 54.72 | 42.26 |
| | SAC-KG$_{\mathrm{LLaMa-13B}}$ | 2.54 | 62.22 | 55.46 |
| | SAC-KG$_{\mathrm{ChatGPT}}$ | **3.72** | **90.54** | **83.30** |
| **Rice expert** | OpenIE 6 (2020) | **7.75** | 50.40 | 38.30 |
| | Stanford OIE (2023) | 4.25 | 43.03 | 29.26 |
| | DeepEx (2021) | 1.50 | 47.36 | 34.01 |
| | PIVE (2023) | 2.00 | 55.17 | 44.14 |
| | SAC-KG$_{\mathrm{Qwen}}$ | 2.50 | 66.66 | 55.25 |
| | SAC-KG$_{\mathrm{Llama2-7B}}$ | 3.75 | 55.56 | 40.00 |
| | SAC-KG$_{\mathrm{Llama2-13B}}$ | 2.62 | 75.00 | 70.32 |
| | SAC-KG$_{\mathrm{ChatGPT}}$ | 3.88 | **93.33** | **84.43** |