KGDist: A Prompt-Based Distillation Attack against LMs Augmented with Knowledge Graphs

Hualong Ma[*]
SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences
China
mahualong54@live.com

Peizhuo Lv[*]
SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences
China
lvpeizhuo@gmail.com

Kai Chen[†]
Institute of Information Engineering, Chinese Academy of Sciences
China
chenkai@iie.ac.cn

Jiachen Zhou
SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences
China
zhoujiachen@iie.ac.cn

Abstract

With Knowledge Graph (KG) increasingly applied in various fields, the integration of KG has gained significant attention to augment the knowledge-specific task capabilities of language models (LMs). However, constructing and maintaining large KGs, much like LMs, can be expensive and challenging, often requiring extensive domain knowledge and human resources. This makes KG a valuable resource potentially vulnerable to theft threats from attackers. In this paper, we present KGDist, the first prompt-based KG distillation technique for extracting KG knowledge from KG+LM augmented models. Through iterations of prompt-based queries, we can steal a substitute KG containing task domain knowledge from the original KG. First of all, we initialize entities from a small scale task-specific corpus. Then, we construct specific task prompts for querying the victim LMs. According to the model outputs, we iteratively select entities showing strong correlation and reconstruct the relation edges for subsequent prompt crafting. We also propose a multi-granularity prompt construction method for reducing the querying cost. After acquiring the extracted KG, we launch a relation type-based pruning to cut off redundant edges forming cycles decreasing the performance of distilled KGs. We evaluate the effectiveness of KGDist on five benchmark KG+LM models designed for various tasks. Results demonstrate that our attack successfully extracts the distilled KGs with minimal performance degradation (under 2.4%) applied on LMs and less storage space. And also, the mechanism we apply greatly saves API queries compared to brute force method. In addition, further experiments demonstrate that we can split the KG knowledge from the LM noises effectively, and the distilled KGs have similar properties in knowledge distribution and graph structures to the original ones. Our code is available at https://github.com/Haro-M/KGDist.

CCS Concepts

• **Computing methodologies** → **Natural language processing**; • **Information systems** → *Graph-based database models*; • **Security and privacy** → **Systems security**.

Keywords

knowledge distillation, language model, knowledge graph

ACM Reference Format:
Hualong Ma, Peizhuo Lv, Kai Chen, and Jiachen Zhou. 2024. KGDist: A Prompt-Based Distillation Attack against LMs Augmented with Knowledge Graphs. In *The 27th International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2024), September 30–October 02, 2024, Padua, Italy.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3678890.3678906

1 Introduction

Over the past few years, the landscape of natural language processing has been dramatically reshaped by the advent of pre-trained Language Models (LMs) such as BERT [14], RoBERTa [38], XLNet [77], and GPT-4 [45]. Their unparalleled performance spans a multitude of applications across various industries, educational landscapes, and professional spheres. However, their prowess, while impressive, is not without limitations. The presence of irrelevant or noisy information within the training data can dilute the model's focus, thus affecting its efficacy in generating accurate predictions [39]. Additionally, the insufficient coverage of domain-specific or expert knowledge restricts their applicability in specialized areas. Meanwhile, the static nature of their training datasets means that these traditional LMs are not designed to dynamically update new knowledge, which poses a challenge in ever-evolving fields.

To further address these limitations, one promising avenue that researchers are exploring is the development of Augmented Language Models (ALMs). As outlined by Mialon et al. [39], these models are designed to integrate traditional LMs with external knowledge sources or processing modules. By synergizing such components, the goal is to achieve heightened performance in tasks

[*] Equal contribution.
[†] Corresponding author

RAID 2024, September 30–October 02, 2024, Padua, Italy
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0959-3/24/09
https://doi.org/10.1145/3678890.3678906

that necessitate specialized knowledge or dynamic, real-time information. Examples of such augmentations include integration with Knowledge Graphs (KGs) [50, 71, 81, 87], using real-time data from search engine results [42, 78, 79], and incorporating symbolic reasoning through modules or code interpreters [20, 25], etc.

Among these augmentations, KG holds a unique position, widely serving as valuable knowledge repositories in numerous domains, from business insights [17], common sense reasoning [59, 66, 76] to medical diagnoses [5, 56, 67]. Particularly, since KG itself is a manually constructed structurally concise knowledge base containing high-density knowledge, the use of KG augmentation can introduce credible knowledge content more efficiently than the traditional way of using corpus to train LMs. Besides, in the event of time-sensitive changes in knowledge, KG can be more simply updated with new knowledge or cleared of outdated knowledge without the need to retrain or finetune, providing greater flexibility. Such features make KG particularly suited for enhancing LMs.

However, constructing and organizing large-scale KGs is no trivial task. Ensuring accuracy and breadth requires extensive domain expertise, complemented by sustained human effort. For example, ConceptNet, a widely-recognized common sense knowledge graph, is built on more than 8 million nodes and 21 million edges [74]. Such endeavours, while laudable, demand considerable resources. Furthermore, the value and effort behind these KGs expose them to unique threats. One possible threat is that an attacker may process malicious attempts to query such augmented models. With the obtained output, the attacker can adaptively reconstruct a substitute of KG behind the model by identifying the knowledge in the input content through the output vector. Such activities threaten the IP underpinning these KGs and underscore the need for robust security measures in the ALM ecosystem. The obtained KG contains the knowledge needed by the attacker, which enables the attacker to bypass the overheads associated with constructing the KG.

A similar threat looming over the deployment of neural network models pertains to knowledge distillation techniques [26, 46, 48]. Knowledge distillation is the process wherein a normally simpler, smaller student model learns from a more complex teacher model by minimizing the prediction error between them. Such a distillation approach bypasses most of the training overheads compared to training the model directly from scratch using raw datasets. Essentially, the distilled model could replicate the performance of the original model, thereby undermining its unique value proposition.

Specially, specialized knowledge distillation methods have been developed for LMs [27, 57]. These techniques are tailored to condense larger LMs into smaller, more manageable ones. Nevertheless, these methods are not suited for extracting knowledge from KGs, given multiple differences between LMs and KGs. On the one hand, neural network models are composed of parameters with continuity, so that gradient optimisation can be performed by minimising the error, whereas KGs, which are databases with logical structures consisting of nodes and edges, do not lend them to such an approach; on the other hand, when building KGs, the specific types of entity nodes and relation edges are naturally in need, such that information cannot be determined by the numerical values of the output vectors obtained from LMs directly. When it comes to the security and safety issues of KGs, existing research has primarily focused on topics such as data poisoning [3, 83, 84] and adversarial attacks [4, 89]. However, the specific domain of KG distillation attacks, wherein the knowledge within KGs could be distilled and potentially misused, remains largely unexplored.

In this paper, we propose KGDist, the first prompt-based KG distillation attack designed for KGs in KG+LM augmented models. The goal of our approach is to extract domain knowledge from the original KGs by an acceptable number of queries, through blackbox access only, to obtain alternative distilled KGs. Based on the paradigm of traditional model distillation, we expect the distilled KGs and the original KGs to have similar accuracy and physical properties. The approach consists of three steps: First, in the Entity Selection step, we initialize a list of core entities to prioritize their queries and update it in the subsequent rounds based on the relevance of the entities. This helps in focusing on essential information related to the target KGs and avoids introducing the unnecessary entities. Then, in the Prompt-based Distillation step, we design a multi-granularity task prompt construction to query the model and apply a confidence-based threshold filtering against the model output. It allows us to rebuild the distilled KG and select entities for future queries effectively. Eventually, in the Distilled Graph Pruning step, we design an cycle pruning algorithm based on relation types. It helps prevent the accumulation of excessive relation cycles that could hinder KG utilization.

In terms of the performance of the distilled KG, vitally we evaluate on multiple KG+LM models and downstream tasks, showing that distilled KG obtained maintains performance with no more than 2.4% variation to the original ones, with also certain similarities in graph properties. Moreover, we use the same settings to distill on both KG+LM and base LM itself, and compare the resulting distilled KGs, demonstrating that merely less than 1% of the knowledge in the distilled KGs we obtain is from the LM. Results show that our approach can effectively strip and extract KG knowledge from the original KG+LM model. Besides, the proposed mechanisms significantly reduce required query numbers (only 6%-11% queries compared to the brute force method) and the storage space of the KGs (less than 3.3% storage compared to the original ones), highlighting the effectiveness and efficiency of our approach.

We present the following contributions:

- We propose a novel distillation attack on KGs in KG+LMs, which, to the best of our knowledge, is the first approach to effectively extract knowledge in graph structures from such models.
- We introduce a new prompt engineering and query paradigm that ensures the distilled KGs effectively and efficiently capture essential information, while the pruning mechanism we apply trims redundant edges and cycles.
- Evaluations demonstrate our attack's efficacy and resilience against multiple KG+LM strategies, maintaining comparable task performance and properties to the original KGs.

Overall, this work offers valuable insights for future research on the security and efficiency of KG techniques.

2 Related Work

KG augmented LM. A Knowledge Graph (KG) is a type of knowledge base that utilizes a graph-based data structure to combine various kinds of data. This structure facilitates the modeling of intricate relations between entities, such as people, places, and concepts, via nodes and edges [44]. Given its nature, KG is well-suited for encapsulating the complexity of human knowledge. Nevertheless, the vastness and intricacy of most KGs bring forth challenges. They typically feature numerous entity nodes and relation edges, making their manual curation and management not only resource-intensive but also reliant on significant human intervention, e.g., ConceptNet has more than 21 million nodes, over 8 million relation edges and more than 15,000 contributors [59]; WikiData has over 107 million nodes and 16.7 billion relation edges [66], completed with 1.98 billion edits made by over 23,400 active contributors [73].

In recent times, advancements in Language Models (LMs) have paved the way for integrating LMs with KGs through diverse methods. When it comes to task-specific augmentation, there are several stand-out techniques. For instance, in the realm of multiple-choice tasks, QA-GNN by Yasunaga et al. [81] assesses the importance of each node in the KG and synergizes LM representations with KG employing Graph Neural Network (GNN) message passing. Following this, GreaseLM by [87] expertly combines token and node representations to achieve a more detailed, cross-modal reflection. Additionally, DRAGON, as introduced by [80], leverages a cross-modal encoder to form a deeply integrated foundation model from both textual and KG data. For form-based Q&A task, the work by Knoblach et al. [30] stands out, where KG is employed to understand varied form structures, effectively translating user queries into answers within the graph structure.

In addition to task-specific models, some strategies strive for more general applicability by integrating knowledge from KG during the LM training phase. A notable example is THU-ERNIE by Zhang et al. [88], which integrates both corpus-based word vectors and KG entity representations. This model also introduces an additional KG-aligned task during its pre-training phase. KnowBERT [50] is another significant model that fuses pre-trained models, KG-based entities, and native word representations to offer a comprehensive pre-training experience. In a similar vein, Wang et al. [71] encode textual entity descriptions with an LM, optimizing the system for both KG and LM tasks. In conclusion, while a myriad of KG+LM combination methods exists, the field has not settled on a standard approach, and remains in a state of rapid evolution. Because of this, the extraction of KG information for such models is also difficult to be carried out in conjunction with specific mechanism details. To design attack methods with generalization, we need to start our scheme design from higher level concepts, such as directly from the relations between input and output information.

Knowledge distillation. Knowledge distillation, originally conceptualized by Hinton et al. [26], serves as a technique for training a student model to mimic the behavior of a normally more complex teacher model. The general methodology involves comparing the output vectors produced by both models for the same set of input data, and then minimizing the difference between these vectors during training. The goal is to imitate the teacher model's decision boundaries as closely as possible.

Efficiency in generating suitable query datasets for the teacher model can be a significant factor in the distillation process. Papernot et al. [48] have suggested to leverage the Jacobian matrix to create an augmented query dataset based on initial samples, making the distillation more effective. Further, Orekondy et al. [46] have explored the use of reinforcement learning algorithms to create specialized datasets that are interpretable and adaptive, further enhancing the efficiency and performance of the distillation process. In a Single-Teacher Multi-Student scenario, You et al. [82] introduces gated support vector machines (gSVMs), which help guide multiple student models. The gSVMs can adapt to training examples of different levels of difficulty, enabling the production of an array of specialized binary classifiers apt for various task domains.

Be that as it may, for models of huge size, it is often difficult to get good results or require huge overheads to distill the whole model directly. Therefore, there are also works on distilling a fraction of the original model capabilities using specific domain knowledge. As proposed by Tang et al. in [62], a portion of the task-related knowledge of a larger model (e.g., BERT, ELMo, etc.) is migrated to a simple LSTM model using a task-related distillation dataset. However, it is worth noting that while knowledge distillation offers numerous advantages, it is not devoid of concerns. The process could unintentionally compromise the intellectual property of the original model. Moreover, attackers might exploit distilled models for advanced attacks, as discussed by Papernot et al. [48] and Shumailov et al. [58]. A less explored realm in knowledge distillation is its application to KG. Unlike typical models with continuous parameters, KG consists of discrete entities and relationships. These characteristics make conventional model distillation techniques ill-suited for KG. Additionally, the potential security vulnerabilities associated with KG remain a topic yet to be thoroughly investigated. As well, considering the large storage space of KG databases and the requirements for adaption when used to enhance the LM (e.g., fine-tuning [50, 88] and adapter training [80, 81, 87]), a smaller distilled version of the original large KG, similar to distilled models, would also be of better utility and efficiency.

KG pruning. Knowledge graph pruning is a crucial process aimed at streamlining large-scale KGs. Its primary goal is to eliminate redundant or less informative elements, resulting in a more concise, efficient, and accurate representation. KG pruning techniques vary, each offering unique approaches to enhancing efficiency and accuracy. For instance, in a study by Mondal and Mukherjee [41], a straightforward multi-source sequential BFS (Breadth-First Search) algorithm is employed. This approach generates a subgraph or forest from a Disease-Symptom graph database, effectively speeding up queries. Similarly, Faralli et al. [18] devises a method that simplifies complex graphs layer by layer, starting from the bottom. Unnecessary nodes and edges are systematically removed while ensuring connectivity, transforming noisy graphs into coherent structures. An innovative approach, known as KGPruning, is proposed by Wu et al. [75]. This method utilizes a graph hierarchy inference technique based on the Agony model. It excels in eliminating noise from the KG while preserving its underlying semantic structure. Furthermore, it establishes a tree-like classification system that harmoniously combines meaning and structure. This dual-pronged approach successfully eliminates irrelevant information and addresses the challenges posed by multiple inheritance problems. However, the most essential difference with our work is that KG pruning can only be applied to already acquired white-box KG, aiming to optimizing the original KG. Our method is more importantly applicable to black-box scenarios that can only be accessed via API to steal KG knowledge hidden behind the KG+LM model, with the reduction of its size as a by-product.

Represent knowledge from LM with KG. KG serves as a type of highly structured data representations that intuitively model relations between entities, and LM gains a broad understanding of semantic relationships and factual knowledge through their training on extensive text corpora. Several studies utilize such properties by using KG structures to characterise the intrinsic knowledge of LMs. For instance, a study by Swamy et al. [61] investigates how LMs acquire knowledge by monitoring changes in KG structures during tasks probing specific relations. This approach provides insights into how LMs perceive various types of relational data. Alivanistos et al. [1] introduces ProP, a technique for completing relations in KG edges using GPT-3. They achieve this by crafting prompts tailored to facilitate the extraction of LM knowledge. Additionally, the Language Model Analysis (LAMA) framework, as proposed by Petroni et al. [51], serves as a standardized benchmark for evaluating the knowledge competencies of LMs. LAMA employs questions from multiple KG sources to assess how well an LM can answer queries requiring factual or relational understanding, albeit limited to simple single-entity token completions.

However, these works focus on small-scale KG generation merely for plain LMs without any KG. In contrast, our approach is designed to be able to extract and separate the knowledge from KG and LM in augmented models. Moreover, these studies view the extracted KGs purely as tools for assessment, and do not consider those as the knowledge that can be extracted by the attacker. With a few straightforward queries, they can only get small KG segments for validation, which are not of further use. Meanwhile, our work treats the KG as an asset containing practical knowledge. We take a more systematic approach to extract the original KG knowledge as a whole, so that the generated KG itself can be utilized for knowledge-related tasks as the original KG, with a similar performance.

Prompt-based knowledge theft from LLMs. As more prompt engineering techniques are applied to popular LLMs, it is natural that attacks that use specially constructed prompts to access the model and steal private knowledge from it to be proposed. For instance, Carlini et al. [7] introduce a new attack on black-box language models to recover the complete embedding projection layer of a transformer model. The attack works top-down to extract the model's last layer. By exploiting the low-rank nature of the final projection layer, which maps from the hidden dimension to a higher-dimensional logit vector, they can extract the embedding dimension or final weight matrix using targeted API queries. Li et al. [35] explore imitation attacks on LLMs to extract their specialized code abilities. They investigate the effectiveness of these extraction attacks with multiple query schemes: zero-shot, in-context, and Chain-of-Thought and design response checks to enhance the imitation training process. However, these studies only steal the attribute parameters or specific capabilities of the model itself, which is still similar to traditional model distillation. Whereas for attacks on the augmented language models, we target the knowledge base mounted behind them rather than the model itself, hoping to get a replica of the knowledge base. Traditional methods cannot be applied to this scenario.

3 Overview

3.1 A Motivation Example

On the one hand, some kinds of KGs contain valuable private information themselves, but can be utilized indirectly through LM, such as Bing and Google's AI chatbots use KG containing user preferences and records for personalized recommendations, yet such KGs is private data of the service providers [55, 64]. On the other hand, due to the requirement for structure and accuracy, KGs of all kinds are generally challenging to build and manage, often requiring massive human or monetary costs. Generalized KGs tend to be of tremendous scale, e.g., ConceptNet has more than 21 million nodes and over 8 million relation edges [59], involved over 15,000 contributors[59] for its maintainence; WikiData has over 107 million nodes and 16.7 billion relation edges [66], and its initial development was funded by a €1.3 million donation, half of which came from the Allen Institute for Artificial Intelligence[49]. While on another side, specialized KGs often require guidance from domain experts with relevant professionalism, which is also costly in expense. According to [2], the long-term cost of a true enterprise KG is around $10-20 million.

All of these examples demonstrate that constructed KGs are vital and valuable, calling for protection. Meanwhile, artificially constructed KGs tend to have larger number of complex relation types due to the fact that it is mostly based on rules considering conceptual coverage. When such complex KGs are utilized to accomplish specific downstream tasks, it is not necessary that the entirety of the entities or relations are required to play a role. Studies such as [33, 81, 87] etc., all opt to compress the original complex relation types when utilizing KGs. This is applied based on redundancy in both entities and relations. For entities, it is obvious that a particular task often requires only relevant entity information; for relations, the relation types are often based on a semantic rule rather than task-oriented when they are initially determined, which makes them unnecessary for specific tasks.

Therefore, we may consider a motivation example where Alice distributes a language model API with a private knowledge graph KG_{Alice} as an augmentation component behind. During API access, there is no direct perception of the KG content itself. Now consider an adversary Eve who has access to the API. Eve's expectation is to accomplish a specific task, and he/she knows only the required relation types and part of entities for this task domain. Using the output of the API, he/she can automatically extract a knowledge subgraph on the needed task domain from KG_{Alice} to obtain a replacement graph KG_{Eve}, thus having a consistent performance with KG_{Alice} in the task domain when augmenting LMs. This bypasses the cost paid by Alice in constructing the KG, thus infringing on the property rights of KG_{Alice}. Moreover, as KG_{Eve} only focuses on the task-related part and is naturally slim in size, for Eve, it has better utilization efficiency and space efficiency than KG_{Alice}.

3.2 Threat Model

Suppose the new attackers, armed with only black-box API access to a victim model that is equipped with the valuable knowledge graph, can query the model through prompts, extracting knowledge from it while remaining hidden in the shadows. Like classic model distillation and attacks for LMs [9, 15, 70], with a small amount

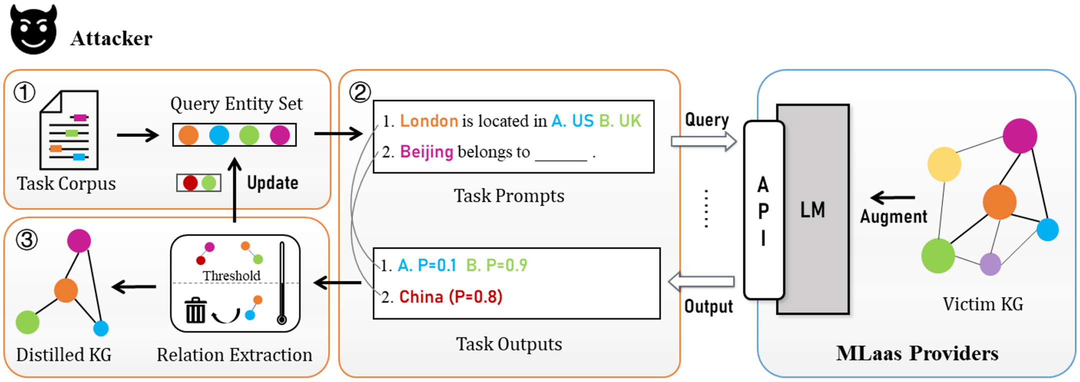

Figure 1: Workflow of KGDist, where P means confidence scores from LMs.

of domain knowledge on the part of the attacker, or by collecting corpus from related domains for keyword extraction, the attacker is able to access entities (normally 1%-2%) in the original KG that are most relevant to the desired task, with specific wordings that may vary with the original KG entities and no structural information. With such knowledge in hand, it is available to construct input prompts tailored to specific task relation types, thereby querying the model and obtaining the output confidence values (i.e., knowledge). Then the attack can collect knowledge leveraged to construct an alternative graph iteratively. This substitute contains private information from the victim and can be integrated into their local models and achieve performance akin to original KG+LM.

3.3 Challenges

Despite the similarities between our approach and traditional model distillation in terms of ideas, reconstructing the knowledge graph itself has unique challenges, mainly due to the structural properties of the knowledge graph itself. First, as can be seen from the actual knowledge graph example above, the spatial volume of a knowledge graph applied to an LLM is usually big. Reconstructing the knowledge graph requires rebuilding the relations between entities, making it difficult in terms of time. Second, unlike continuous parameters of a model in the traditional distillation attack, the knowledge graph is diverse in terms of logical and semantics structure, which makes it challenging to automatically optimize by methods utilized by the attacker and those in the original knowledge graph cannot be fully aligned. Thus, it is also challenging to use semantically different relation types to maximize the knowledge migration from original knowledge graph to the distilled one.

3.4 Approach Overview

Figure 1 presents an overview of KGDist, comprising three main phases: Entity Selection, Prompt-Based Distillation, and Distilled Graph Pruning.

In Entity Selection, we initialize an entity set that serves as the starting point for the distillation attack, which is then dynamically updated in iterative rounds based on a confidence score designed to evaluate each entity pair in the prompts queried.

In Prompt-Based Distillation, we employ a multi-granularity method for construction of different model input prompts to balance between computational efficiency and effectiveness. According to the model output, we identify and select entities that exhibit high levels of correlation with the query and reconstruct the relation edges for subsequent prompt crafting.

In Distilled Graph Pruning, with the draft KG obtained, we perform optimization to eliminate redundant or noisy information within, thus improving the efficiency of utilization and making them structurally closer to manually constructed KGs. Given that cycles within the graph can introduce complexity and potential errors, we focus on the types of relations that produce cycles and apply a relation type-based pruning algorithm.

4 Methodology

4.1 Entity Selection

As we can only access the model through API query, it is vital to select the entities for each round (which are defined as "focal entities") reasonably. Inappropriate selection of entities may lead to redundancies in the number of queries, thus drastically increasing query costs. This involves two primary aspects: determining the starting point of the distillation by selecting a suitable initial set of query entities, and dynamically updating the query entities for subsequent rounds based on the results of each iteration.

For the initial phase, according to [22, 32, 68], traversing dense subgraph parts first during graph traversal can indeed improve efficiency. Combined with the distributional characteristics of the knowledge graph, we use a localisation-first traversal method based on knowledge density to determine the starting point for distillation thereby improving efficiency. Intuitively, the most important part of the concepts in a given knowledge domain tends to be the most associated with other concepts, which means that the relation subgraphs centred on this part of the nodes in the original KG are more densely structured in terms of edges, as such an idea has already been widely accepted in work such as [6, 23, 47]. Furthermore, for attackers, even in a black case scenario where the original KG is completely inaccessible, they can simply access a small set of entities in the desired domain by collecting related domain corpus (e.g., from the Internet or a relevant task corpus) for keyword extraction. Since the LM itself has the ability to understand semantics, this part of the entities does not need to be strictly aligned with the entity names of the original KG (e.g., case, singular or plural, etc.). They can then determine the order of the initial entity set through the same priori knowledge source, e.g., by quantifying the word frequencies. This improves the efficiency of graph distillation on the one hand, and on the other hand enables the attacker to retain the parts of knowledge that are most relevant to the domain concepts by early stopping, for example, when the attacker merely need a small scale distilled KG.

In subsequent rounds of iteration, for the purpose of efficiently probing all relevant entities and edges to reconstruct the distilled KG, we utilize the selected entities from prior rounds as the basis for generating new query prompts (See Section 4.2). Leveraging these selected entities allows for a more focused and precise interrogation of the model, thus optimizing the query process.

Since the essence of KG-based LM augmentation is to utilize the existing knowledge in KG to complement the LM trained with the corpus, it is natural that for the input prompt, the output content of the KG+LM combination will have a higher confidence level if KG-related entity and relation information exists. For the output confidence values, we can access them through the model's API. In particular, many language models give developers/users information such as the probability or expected accuracy of the relevant prediction results ([11, 28, 29]), which we can utilize as our confidence values. Intuitively, it seems that we can directly reconstruct the KG edges through the model's top predictions of the inputs, but in reality, doing so directly may introduce noises. During training, LM, as a probabilistic model, learns not only knowledge information, but also grammar rules (e.g., insubstantial vocabulary such as articles, prepositions, etc.), which does not play a big role in the reconstruction of KG. Such noises may dilute the extracted knowledge and cause a surge in subsequent queries, thus affecting the efficiency of our method. On the other hand, similar to the hidden vocabulary phenomenon [13] of LMs, some combinations of words, although not immediately conforming to semantic logic, could still reflect the underlying knowledge encapsulated and reinterpreted from the KGs. Setting a hard threshold to separate out merely logical prompt entries would lead to a considerable loss in the KG knowledge. Therefore, for entity selection here, we use a soft threshold mechanism. We retain elements that meet the threshold condition, along with a few that don't, to ensure the threshold elements comprise a specific percentage of the total. For example, in the dataset $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$ with a threshold setting of under 0.4, hard threshold filtering retains only $\{0.1, 0.2, 0.3, 0.4\}$. Soft threshold filtering with an 80% compromising rate retains $\{0.1, 0.2, 0.3, 0.4, 0.5\}$ to ensure 80% of the elements are below 0.4. We tune the threshold to ensure most but not all of the prompts[1] exceeding it are coherent and semantically logical, keeping part of the knowledge from such results.

4.2 Prompt-Based Distillation

With the filtered entities as the focal points for each round, we need to construct prompts for querying the model. As demonstrated in several recent related studies [37, 60, 85, 91], model access through sensible prompt engineering can give rise to effectiveness and efficiency in scenarios involving knowledge extraction. Therefore, the model query phase in our distillation attack scheme places special emphasis on the careful design of the query prompt, satisfying our need to focus on the entities and relations originating from KG more accurately and efficiently.

Query Strategy. The most straightforward approach to formulate a query strategy might entail the amalgamation of all entities and relations for a comprehensive traversal. Nevertheless, this approach invites two predominant challenges. Firstly, the inherent mechanism of LMs signifies that the word order and different modifiers within the input prompts may disrupt output results, as discussed in [8, 31, 54], both of these elements may introduce uncontrollable noise into the distillation knowledge. In order to minimize this uncertainty, we utilize concise assertions and reverse the positions of the subject and object entities to perform queries in both directions, and preserve only outputs with the highest confidence.

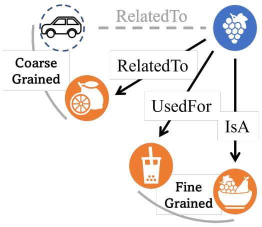

Figure 2: Example for Multi-Grained Relation Query.

The second challenge lies in the voluminous number of entities and the uneven distribution of relations present in KGs. This makes an exhaustive traversal of every entity-relation pair not only resource-consuming but also inefficient. To tackle this, we propose a two-tiered query approach according to the types of relations, categorizing relations as either coarse-grained or fine-grained. Coarse-grained relations, such as *RelatedTo* and *About*, serve as indicators that simply signal the existence of relations between two entities, without offering specific detail. Conversely, fine-grained relations like *IsA* and *UsedFor* offer precise, contextual information about the interplay between entities. As is illustrated in Figure 2, When *Grapes* is selected as the focal entity, we first query its coarse-grained relation with other nodes to be queried, and stop the query if the output confidence fail to surpass the predetermined threshold (e.g., *Automobile*). Subsequently, we delve into fine-grained relations for deeper analysis and contextual understanding. If such edges exist, we may further replace the coarse-grained edges with the fine-grained relations (e.g., *Beverage* and *Salad*).

Prompt Construction. With the aforementioned entity selection and query strategy, we can determine the query pattern and relation granularity. Accordingly, we proceed with the construction of the prompts based on the specific scenario. We divide the mainstream knowledge graph language model applications into two separate scenarios and focus our attack independently. The first scenario, referred to as **Case 1**, deals with users inputting pre-established entities and relations into the system. The LMs then generate and return confidence scores that quantify the plausibility of these specified entity-relation pairs. Notably, this scenario does not introduce new entities and is typically utilized in applications like multiple-choice Question and Answer (Q&A) systems. The second scenario,

[1] To balance the number of queries and knowledge coverage, we use small batch sampling for testing and choose 90% here as the threshold.

i.e., **Case 2**, is more dynamic in nature. Here, users provide entities and relations they have at hand, and the LM proceeds to suggest correlated new entities. Applications of this case can be observed in Q&A systems and recommender systems, among others.

For both scenario, firstly, we initialize entity set E_0 and the relation set R from task corpus c_0 for distillation:

$$E_0, R = initialize(c_0) \tag{1}$$

Case 1. In the first scenario, our goal is to construct the relation edges between entities at hand. An example of multiple choice Q&A task is demonstrated in Equation 2, where we choose entities for creating prompts in the task-corresponding format:

$$\textbf{Prompt: } \textit{"Which } \underbrace{\textit{is related to}}_{r_i} e_i \textit{? (a) } e_1\textit{, (b) } e_2\textit{, } \cdots \textit{, (N) } e_N\textit{"} \tag{2}$$

where e_i is a focal entity, $e_j (j \in 1, 2, ..., N)$ is a query entity, r_i is a focal relation (*RelatedTo* here as an example), and N be the amount of options, determined by the victim LM capacity. With the prompt construction, we can formulate distillation in such case as below:

$$\begin{aligned} edge_n &= \{(e, e', r) \in E_n \times E_{unc} \times R \mid C(e, e', r) > \tau_c\} \\ E_{n+1} &= \begin{cases} random\_select(E_{unc}, q_c), & edge_n = \emptyset \\ \{e' \mid (e, e', r) \in edge_n\}, & \text{otherwise} \end{cases} \end{aligned} \tag{3}$$

where the prompt (e, e', r) constructed from the elements in the current round of focal entities E_n, the set of unchecked entities E_{unc}, and the relation set R is input into the model M. The model subsequently yields the corresponding output confidence $C(e, e', r)$ for the prompt, and prompts that surpass τ_c are kept in the distilled KG. Concurrently, the tail entity e' is appended to the focal entities E_{n+1} for the subsequent round. In instances where no prompt has a confidence level above τ_c, a random selection of q_c entity from E_{unc} are designated as the focal entities for the next round. The entities that have been checked are removed from E_{unc}.

Case 2. For the second scenario, we aim to reconstruct the relation edges and extract unknown entities. Since the mask language modeling (MLM) task is widely used in natural language model training [14, 38, 45], and experiments [1, 51] have shown the ability of LMs to complete the content of the unmask. We employ the MLM format by designing prompts with the queried entity as the predicted [MASK] content, as illustrated in Equation 4,

$$\textbf{Prompt: } \underbrace{\textit{"[MASK] ... [MASK]}}_{n \ tokens} \ \underbrace{\textit{belongs to}}_{r_i} e_i\textit{."} \tag{4}$$

where e_i be a focal entity, and r_i be a focal relation (*BelongsTo* here as an example), $n \in \mathbb{Z}$ be the length of the distilled entity.

Specifically, we utilize multiple connected [MASK] tokens for predicting longer entity names. We also employ the cumulative confidence of each [MASK] prediction as the filtering threshold. For multi-word-length entity prediction, we examine the predicted words at each [MASK] position, disregard filtered words, and retain only the highest-confidence combination. The MLM task may introduce irrelevant vocabularies such as conjunctions and pronouns, the probability of these terms appearing in knowledge graph entity entries is low, but they may generate a large number of redundant entities in the distilled knowledge graph, leading to an increased number of unnecessary queries and reduced query efficiency. To address this issue, we construct a filtered word list to exclude such words (Table 3). With the prompt construction, we can formulate distillation attack in such case as below:

$$\begin{aligned} edge_n &= \{(e, e', r) \mid (e, r) \in E_n \times R,\ C(e, e', r) > \tau_c\} \\ E_{n+1} &= \{e' | (e, e', r') \in edge_n\} \end{aligned} \tag{5}$$

where the prompt (e, r), constructed from the elements in the current round of focal entities E_n and the relation set R, is the input for the model M. The model outputs the predicted entities e' and the corresponding confidence $C(e, e', r)$, and the statements (e, e', r) with confidence above the threshold τ_c are added to the distilled KG, and the entities are added to focal entities E_{n+1} of the next round.

After multiple iterations, when no additional new entities are to be investigated or the iteration number exceeds max epoch, we ultimately obtain the set of relation triplets containing distilled entity-relations, which can be formulated as follows:

$$G_{distilled} = (\bigcup_{i=0}^{n} E_i, \bigcup_{i=0}^{n} edge_i) \tag{6}$$

Moreover, to display a clearer picture of paradigms above, we showcase practical examples for both scenarios in Section 5.6.

4.3 Distilled Graph Pruning

With the triplets extracted, we can reconstruct our distilled KG. Next, it is necessary to prune the obtained distilled relation edges. Due to the noise sourced from the training corpora of LMs, distilled KGs can indeed exhibit a plethora of edges. As evidenced by classic works on KG utilization like [18, 63, 65], it has been observed that small cycles in KGs can diminish the efficiency of the utilization, consequently calling for rigorous pruning in preprocessing. On the other hand, distilled KGs contain some redundant edges due to their automated generation, quite divergent in pattern from the artificially constructed KG. Meanwhile, it is challenging to design a pruning technique in our scenario. The most important reason is that the distilled graph may also be large in terms of volume and complex in relation types, making it difficult to be pruned in terms of time cost. Also, since our distillation relation types vary a lot in different situations, traditional rule-based pruning approaches may fail to cope with knowledge graphs with different relation types. Therefore, with the relation types predefined, we design a simple yet efficient type-based approach for distilled KG pruning.

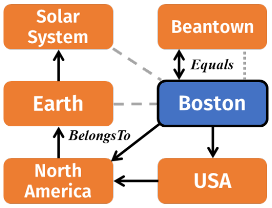

Figure 3: Example for KG Pruning.

We apply a BFS-based cycle detection on distilled KG and identify the following two types of relations that would form most of small cycles: One category includes nested relation types (e.g., *IsA* and

PartOf). Starting from a focus entity, such relation edges can form chains with long distances, while edges between initial and distal nodes, although semantically correct, reflect unnecessary knowledge and form redundant cycles. As shown in Figure 3, focusing on *Boston*, although all the relation edges may be valid, not all of them are needed. Meanwhile, classical KG+LM methods [36, 81, 87, 90] tend to utilize knowledge paths in KGs with limited hops.

Taking these factors into account, we introduce a pruning approach that employs a hyperparameter, HOP_{max}, to limit the edge number retained for any nestable relations with the same starting point. For instance, in Figure 3, when we set $HOP_{max} = 2$, only edges connecting *Boston* to *USA* and *North America* would be retained, thus reducing redundancy. The second category pertains to symmetric relation types (e.g., *Equals* and *NextTo*). Although edges in KGs are defined with the concepts of head and tail, the actual distinction between these two is often not considered in KG augmentation approaches. We find that interchanging the head and tail entities has a negligible impact on the semantic of such symmetric relations. To optimize for this, we propose a pruning strategy that retains only the edge with the higher confidence score for each symmetric relation.

As depicted in Figure 3, overall, our pruning strategy has the following benefits: first, we eliminate redundant edges from the knowledge graph, making it more time-efficient for subsequent loading and utilization sessions; second, since the attacker's goal is to obtain a functionally equivalent subgraph to be used only on the desired domain, pruning further reduces the spatial costs of the subgraphs; and finally, pruning also aligns the distilled more closely with the original KG, ensuring that the pruned KGs maintain performance with minimal degradation.

It is worth noting that although we demonstrate our approach of querying and pruning using specific scenarios and relation types for instance, since the criteria we use (e.g., whether a query returns a new entity, whether a relation exists or can be nested) are based on high-level logic, our approach applies to all scenarios or specific relation choices with transferability. Moreover, we can distinguish these scenarios into two most dominate scenarios according to whether the victim model returns new entities during the attack: scenarios in which no new entities are returned as in [30, 33, 80, 81] ; and cases in which new entities are returned including [50, 71, 88]. Deisgned at a high level, this category criterion can mostly cover mainstream model application scenarios.

5 Experiment

5.1 Experimental Setup

Our experimental setup is shown in Table 1, as described below.

LM and KG architectures. We employ various representative KG+LM models, categorized into two cases:

- **Case 1:**
  - *QA-GNN* [81] and *DRAGON* [33] are both based upon the *RoBERTa*$_{large}$ [38], augmented with *ConceptNet* [59].
  - *GreaseLM* [87] relies on *AristoRoBERTa* [12], finetuned from *RoBERTa*$_{large}$ and augmented with *ConceptNet*.
- **Case 2:**
  - *KnowBERT* [50] utilizes *BERT*$_{base}$ [14], and is augmented with a self-constructed *Wikipedia* KG.
  - *KEPLER* [71] employs *RoBERTa*$_{base}$ and is augmented with a self-constructed *WikiData5M* KG.

Evaluation Tasks. We use the tasks where KGs contribute more to the performance in the original model evaluation. For all tasks that are not explicitly stated, the assessment metric is the task accuracy. For **Case 1**, we select the following task:

- ***OpenbookQA (OBQA)*** [40]. A 4-way multiple-choice Q&A resource supplemented by an open book of scientific facts.

For KnowBERT in **Case 2**, we select the following tasks:

- ***KG Probing***. Following the original work, we generate tuples from KGs and create two testing instances with the subject or the object masked. The evaluated metric is the Mean Reciprocal Rank (MRR) of the masked word predictions, averaging the reciprocals of ranks for first correct answers.
- ***Words in Context (WiC)*** [52] provides two sentences with a same word, and requires the LMs to discern whether both share the same meaning in the contexts, evaluating the accuracy of contextual word representations.
- ***OpenEntity*** [10] assesses the performance of entity typing by classifying entity mentioned into pre-established types.

For KEPLER in **Case 2**, we select the following tasks:

- ***FewRel*** [24]. A dataset for relation classification. Gao et al. [21] proposed FewRel 2.0, adding data from the medical field. During each round, N relations, K supporting instances per relation, and several queries are randomly selected and the models assign them to one of the relations, relying solely on these instances. [2]
- ***OpenEntity***. Same as in the KnowBERT evaluation.
- ***LAMA* [50] & *LAMA-UHN* [53]**. A popular knowledge probing method that features cloze-style questions. LAMA-UHN is the enhanced version with dubious templates elimated.

Entities and Relations. For initial entities, We control for the 1-2% of entities in the original KG that are known by the attacker to be most relevant to the task domain. Specifically, as in [34, 43], we extract the public entities of the task domain corpus and the original KG, sort them according to their number of hits in the task corpus, and select the most relevant ones to initialize. For relations, due to the complexity of the relation types in the original KGs, according to [81, 87], we select commonly used relations based on coverage on logical relations, as shown in Table 2.

Besides, considering that there may be potential effects of different relation types on our attack, we also test with different types, as discussed in Section 6. For filtered entities, from the pre-experiments, we identify the most high frequency noise words as filtering entities, as shown in Table 3.

Hyperparameters. For the different hyperparameters, we use the following guidelines for the determination:

- τ & $\mathbf{WL_{max}}$. As τ discussed in Section 4.2, since MLM may also output illogical results for more [MASK] tokens, we set them to ensure 90% prompts above it are logical. We discuss the effect of different τ, shown in Section 6.
- $\mathbf{HOP_{max}}$. We set it preliminary experiments on sample batches, experiments of the effect are also provided in Section 6.

[2] We set N = 10, K = 5 to maximize the performance difference between the models with and without KGs.

Table 1: Experimental Setup

Case	Approach	Base Model	Knowledge Base	Task	Init Ent (Size %)[4]
Case 1	QA-GNN	RoBERTa$_{large}$	ConceptNet[1]	OpenBookQA	5,289 (0.66%)
	DRAGON				
	GreaseLM	AristoRoBERTa			
Case 2	KnowBERT	BERT$_{base}$	Wikipedia	KG Probing Word in Context OpenEntity	8,102 (1.72%)
	KEPLER	RoBERTa$_{base}$[2]	WikiData5M	FewRel OpenEntity LAMA[3] LAMA-UHN[3]	30,250 (0.66%)

[1] Following original settings, only the entity subset in English of ConceptNet is utilized.
[2] To control the variables, we used the same small-scale corpus for RoBERTa fine-tuning in original work.
[3] For LAMA we evaluate on T-REx and SQuAD, and for LAMA-UHN we choose Google-RE and T-REx.
[4] Denotes the size of initial entity set and its proportion in the original entity set.

Table 2: Distillation Relation Set

(a) Case 1

Type	Relation	Nestable
Abstract	*RelatedTo*	Yes
Causal	*Causes*	Yes
Belonging	*BelongsTo*	Yes
Category	*IsA*	Yes
Usage	*UsedFor*	No
Impact	*Affects*	No

(b) Case 2

Type	Relation	Nestable
Abstract	*RelatedTo*	Yes
Category	*IsA*	Yes
Influence	*Influences*	No
Belonging	*BelongsTo*	Yes
Causal	*Causes*	Yes
Usage	*UsedFor*	No
Location	*LocatedIn*	No
Similarity	*SimilarTo*	No

Table 3: Filtered Word Types

Type	Instances
Adverb	*almost, mostly, only*
Pronoun	*you, him, everyone, why*
Article	*a, an, the*
Conjunction	*because, until, but*
Symbol	*:, >, <,], [UNK]*

- $\mathbf{q_c}$. It limits the query count per round, and has no effect on the total query and KG performance. We select it based on our GPU resources and can change it as well.

Accordingly, the hyperparameters are set as follows, for Case 1:

- For both: HOP_{max} = 6, q_c = 10
- QA-GNN: τ_c = 18, GreaseLM: τ_c = 15

For Case 2:

- For both: HOP_{max} = 3, WL_{max} = 4, $epoch$ = 500
- KnowBERT: τ_c = -3.5 per token, KEPLER: τ_c = 18 per token

5.2 Task Performance

We evaluate our attack against various KG+LM models, as shown in Table 4. We find that the distilled KGs, like the original KGs, perform comparably well when augmenting the base models[3]. Specifically, in Case 1, all distilled KGs achieve task accuracy results that are nearly identical to the original ones, with a maximum difference of just 0.60%. In Case 2, KnowBERT's distilled KG outperforms the plain LM in two tasks with strong knowledge correlations, with only minor differences of 0.03% and 0.20%. The WiC task's performance decreases more, likely due to lower knowledge correlation of the task. Nevertheless, this performance decrease remains below 2.40%, maintaining an advantage over the baseline model. The performance reduction in KEPLER can be attributed to its reliance on the extensive WikiData 500M and the relatively small initial entity set. These results emphasize that our method successfully extract the task-related knowledge from the original KGs.

5.3 Efficiency

In the preceding sections, we have discussed various strategies aimed at reducing queries to improve the overall computational efficiency of our method. Alongside this, by focusing on task-relevant subgraphs, our approach not only maintains high performance but also results in more compact distilled KGs when juxtaposed with the original, more expansive KGs. Consequently, this section assesses the temporal and spatial efficiency of our proposed technique.

Temporal Efficiency. We introduce multi-granularity prompt construction and word filtering to enhance efficiency in KG distillation. A basic approach is to use brute force algorithm to query all combinations of entities and relations. For temporal efficiency, we use query counts rather than times as a measure because it is unfair to compare times between APIs of models with different time complexities. Since the lengths of the query statements we constructed do not fluctuate much, it is more objective and accurate to use query counts for the efficiency measure. Table 5 presents that our approach substantially improve the temporal efficiency, reduced by up to 93.17% from query numbers of the brute force method. On the other hand, since we initialize with the most important entities, which tends to be in the most densely informative part of the original KG, the beginning rounds will naturally extract the most important relations. This allows us to further reduce model queries by using mechanisms such as early-stopping, while sacrificing only a small fraction of the task accuracy. As is shown in Table 6, we revisit GreaseLM and QA-GNN in Case 1 and apply early stopping.

It can be shown that the distilled KG obtained from the first 25% and 50% of the queries already contains a large part of the KG knowledge, and only loses no more than 3% of the performance compared to the full queries, which is still higher than the base LM. It is also noticeable that when the edge number is just less than half of the distilled KG with full queries, there is still a 2.8% and 0.6% performance improvement compared to the plain LM, proving that the knowledge gained from the initial distillation rounds is the most important, proving the effectiveness of our entity set initialization.

[3] We also evaluate LAMA and LAMA-UHN tasks despite their low original accuracy, as they are widely used for evaluating LM knowledge performance [19, 24, 69, 86].

Table 4: Task accuracies of baseline LMs, original and distilled KGs.

Case	Model	Task	Augmented Graph		
			No KG	Original	Distilled
Case 1	QA-GNN	OpenBookQA	64.8	67.8	67.2
	DRAGON		64.8	72.0	71.4
	GreaseLM		78.4	83.9	83.1
Case 2	KnowBERT	KG Probing	0.09	0.31	0.28
		WiC	65.4	70.9	68.5
		OpenEntity	73.6	76.1	75.9
	KEPLER	FewRel	64.7	71.0	68.8
		OpenEntity	73.8	76.2	75.3
		LAMA$_{\text{T-REx/SQuAD}}$	23.2/8.0	24.6/14.3	24.0/12.8
		LAMA-UHN$_{\text{Google-RE/T-REx}}$	2.4/15.2	3.3/16.5	3.0/16.0

All tasks use the accuracy rate (%) except for the KG Probing task, which uses the MRR for assessment.

Table 5: Query counts of distillation.

Model	Brute Force	Our Method
QA-GNN	21.6M	1.5M
GreaseLM	21.6M	2.4M
KnowBERT	72.6M	5.3M (12.2M)[1]
KEPLER	44.1M	2.8M (5.9M)[1]

[1] Results out/in the brackets represent with/without word filtering.

Table 6: Early-stopping based on query counts.

Model	Query Limit	Edges (Rate %)	Acc (%)
GreaseLM	Original	2,487,810	83.9
	Non-Limit (2.2M)	26,052 (1.05%)	83.1
	50% (1.1M)	19,824 (1.05%)	82.5
	25% (0.6M)	10,418 (0.42%)	81.2
	Plain LM	-	78.4
QA-GNN	Original	2,487,810	67.8
	Non-Limit (1.5M)	11,084 (0.44%)	67.2
	50% (0.75M)	9,345 (0.38%)	66.4
	25% (0.375M)	6,842 (0.28%)	64.8
	Plain LM	-	64.2

Spatial Efficiency. Table 7 highlights the significant storage reductions comparing the original KGs to their distilled versions.

Table 7: Storage comparison of original and distilled KGs.

KG	Source	Entities (Rate %)	Edges (Rate %)
ConceptNet	Original	799,273	2,487,810
	QA-GNN	5,289 (0.66%)	11,084 (0.45%)
	DRAGON	5,289 (0.66%)	20,183 (0.81%)
	GreaseLM	5,289 (0.66%)	26,052 (1.05%)
Wikipedia	Original	470,113	446,300
	KnowBERT	15,218 (3.22%)	10,441 (2.34%)
WikiData500M	Original	4,594,485	20,614,279
	KEPLER	51,654 (1.12%)	58,188 (0.28%)

For entities, the distilled KGs hold at most 3.3% of the original, and more typically, less than 1%. Similarly, the number of relation edges in the distilled KGs seldom exceed 2.4% of the original ones, generally under 1%. This drastic reduction can be attributed to two factors. First, by prioritizing domain-specific entities and relations, our design manages to dramatically cut down on storage space, making our distilled KGs far more leaner than their original versions. Second, as elucidated in Section 6, results have shown that KG+LM augmentation methods is usually not able to harness the full potential of the original KG. In essence, our distilled KG captures the LM's genuine utilization of the original KG more effectively, which aligns with the core principles of distillation attacks and also provide insights on such KG+LM methods.

5.4 Comparison with Ground-Truth KGs.

In this subsection, we compare the distilled and the groud-truth KG. Since our interest is the task domain subgraph, we extract the entity nodes in the original KG and their neighboring edges within 2 hops as ground-truth subgraphs. Next, we evaluate them from two perspectives: graph properties and knowledge properties.

Graph Properties. Since distilled KGs are constructed automatically, there may be variability in physical properties from the original KGs. We assess the relationship between them, and a detailed comparison of their graph properties is presented in Table 8.

Table 8: Comparison of graph properties.

(a) Case 1

Metrics	QA-GNN		DRAGON		GreaseLM	
	Ori.	Dist.	Ori.	Dist.	Ori.	Dist.
Avg Degree	3.88	2.25	3.92	3.27	4.17	3.70
Avg Cycle Len	4.76	4.00	5.02	4.01	5.79	4.01
Cycle Num	2,368	145	3,104	2,514	3,957	3,795
Avg Path Len	2.18	2.01	2.09	2.21	2.16	2.20
Density	0.0015	0.0020	0.0013	0.0014	0.0011	0.0008

(b) Case 2

Metrics	KEPLER		KnowBERT	
	Ori.	Dist.	Ori.	Dist.
Avg Degree	4.58	2.10	1.27	2.00
Avg Cycle Len	64.05	5.22	16.85	4.86
Cycle Num	16,735	1,688	1,480	122
Avg Path Len	NC[1]	3.69	NC	NC
Density	0.0003	0.0001	0.0006	0.0002

[1] NC indicates that the graph is not connected so this metric can't be measured.

In Case 1, we notice that the distilled KGs closely resemble the original subgraphs in graph properties. This similarity is evident in metrics such as node average degree, graph density, and average path length, indicating that relevant knowledge of each node has been effectively extracted and that knowledge paths between nodes remain intact, indicating that our distilled KG is similar to the original ones in terms of the knowledge distribution. Our method for pruning also demonstrates its validity, as the distilled KG exhibits less and shorter cycles compared to the original subgraph.

Notably, when using the same KG in Case 1, GreaseLM achieves the highest task accuracy, suggesting that its augmentation efficiently harnesses KG potential, as reflected in the high similarity in graph properties between the distilled KG and the original. Essentially, this reaffirms that our method can reflect the utilization efficiency of KGs by LMs. In Case 2, the outcomes still fall within the typical metric boundaries for natural KGs. However, we observe reduced connectivity between the KGs, with both the distilled KGs and subgraphs displaying disjunctions and lower node degrees. This divergence may be due to the nature of the MLM tasks, which introduces randomly distributed knowledge, resulting in looser and more fragmented KGs compared to Case 1.

Knowledge Properties. In addition, we assessed the similarity between the distillation KG and the ground-truth KG in terms of knowledge attributes in three different dimensions. We propose the following two metrics to evaluate the similarity of knowledge properties: the Entity Hit Rate (abbreviated as EHR), which is computed to calculate the percentage of nodes in the distillation KG that appear in the ground-truth KG; and the Relationship Presence Rate (abbreviated as RPR), which measures whether the edges in the distill KG, with nodes at their ends, have paths with hops less than three in the original KG. The result is presented in Table 9.

Table 9: Comparison of knowledge properties.

Metrics	QA-GNN	DRAGON	GreaseLM	KEPLER	KnowBERT
EHR (%)	100.00	97.46	99.92	82.64	83.76
RPR (%)	25.74	27.98	34.12	12.89	14.71

As can be seen from the Table 9, results of EHR are all above 90% in Case 1, and slightly lower but also above 80% in Case 2. The reason for this and the experimental results of graph properties is also similar, due to the fact that the models in Case 1 have fewer mixed noise compared to the models in Case 2. For RPR, it has a lower value overall, which is due to a larger change in the structure of the KG after the LM interpretation. However, relatively speaking, the trend of RPR also has similarity with other methods, and the better the performance of the KG enhancement method the higher the RPR of its distill KG, which proves that our KG distillation method still reflects the knowledge information of LM utilizing KG.

5.5 Ablation Study

Distilled Graph Pruning. In our evaluation of GreaseLM, we assess the effectiveness of our pruning strategy. From the spatial efficiency perspective, as illustrated in Table 10a, after implementing both pruning strategy, the number of relation edges in the KG reduced by a notable 17.20%. Intriguingly, rather than seeing a decline in performance, the task accuracy rate experienced a 1.42% boost, mainly due to the fact that pruned edges may in turn introduce noise to the task. And furthermore, since these knowledge graphs are applied on the model for use, this also greatly reduces the time cost of model augmentation; moreover, the pruned knowledge graphs are closer to the original subgraphs in terms of the number of edges. This demonstrates that our pruning approach can improve query efficiency and reduce noises in the distilled KG, further enhancing its task performance.

Prompt Construction. We employ KnowBERT to assess the effectiveness of the proposed query reduction strategy in prompt construction. Given the prohibitive cost of multiple repetitions, we commence our evaluation with a random selection of several entities[4] from the original KG. We compare the query count and the ratio of new entities and query counts (which we define as NE/Q) with and without the query reduction strategy. The evaluation results are shown in Table 10b. Discarding word filtering and multi-granularity prompt construction mechanisms precipitates an exponential surge in the number of queries to 1.74 and 12.68 times of the original, respectively. Despite this, NE/Q witnesses a significant drop to 64% and 18% of the original, illustrating a disproportion between overhead and benefit. Such observations unequivocally corroborate the vital role and indisputable necessity of the introduced mechanism.

Table 10: Ablation Study

(a) Pruning

Method	Edges	Accuracy
Ours	26,052	83.10%
- Pruning Step	31,465	81.68%

(b) Prompt Construction

Method	Queries	NE/Q
Ours	1.00x	1.00x
- Word Filtering	1.74x	0.64x
- Multi-Granularity Prompts	12.68x	0.18x

Table 11: Examples for prompt query.

(a) Case 1: GreaseLM (τ_c = 15)

Eg.1	**Prompt**: *Which is related to glacier?* ***Statements***: *A) Trees B) Passed C) Sunset D) Frozen Water*
	Output: P(A) = -20.3411, P(B) = -12.6022, P(C) = -15.0365, P(D) = 15.5414
	Description: Since confidence of D is greater than the threshold, {*glacier, frozen water, RelatedTo*} is retained.
Eg.2	**Prompt**: *Which belongs to sea anemones?* ***Statements***: *A) Large object B) Invertebrates C) Living organism D) Fibrous tissue*
	Output: P(A) = 6.7267, P(B) = 24.0860, P(C) = 20.9996, P(D) = 18.2507
	Description: Confidence of B,C,D are greater than the threshold, {*invertebrates, sea anemones, BelongsTo*}, {*living organism, sea anemones, BelongsTo*} {*fibrous tissue, sea anemones, BelongsTo*} are retained.
Eg.3	**Prompt**: *Which is related to land mass?* ***Statements***: *A) Cows corn B) Prolongs C) Receiving D) Magnetite*
	Output: P(A) = -9.2758, P(B) = 3.0573, P(C) = -14.3154, P(D) = -4.2480
	Description: Since confidence of no statement is greater than the threshold, no relation edge is retained.

(b) Case 2: KEPLER (τ_c = 18 per token)

Eg.1	**Prompt**: *[MASK] is related to misery.*
	Output: *joy* (P = 21.6423); *pain* (P = 21.2853); *sin* (P = 20.1220)
	Description: With confidence greater than the threshold, {*joy, misery, RelatedTo*}, {*pain, misery, RelatedTo*}, {*sin, misery, RelatedTo*} are retained.
Eg.2	**Prompt**: *West Africa belongs to the [MASK] [MASK] .*
	Output: *african region* (P = 36.4353)
	Description: With cumulative confidence greater than the cumulative threshold, {*west africa, african region, BelongsTo*} is retained.
Eg.3	**Prompt**: *[MASK] [MASK] is a crown jewel.*
	Output: *gold diamond* (P = 36.0872)
	Description: With cumulative confidence greater than the cumulative threshold, {*gold diamond, crown jewel, IsA*} is retained.

[4] To balance the number of queries and the credibility of our validation, we randomly select a moderate number of 1,000 entities.

5.6 Practical Examples

Examples of distillation prompts. We select examples from both cases and analyze the results as follows in Table 11: In both cases, using entity-based question prompts filtered by output confidence threshold accurately reconstructs entity relations. In Case 1, distilled entities and relations align logically due to the fixed entity set without generation of new entities, the possibility of direct noise generation by LM is greatly reduced. In Case 2, using single-mask token prompts maintains strong interconnections among entities, but with multiple mask tokens, occasional semantic coherence issues arise due to the inherent instability of the MLM task. Nevertheless, such illogical predicted words still exhibit strong semantic ties to target entities, underscoring the knowledge retained from the original KGs. For example, in Eg.3 of Table 11b, although *gold diamond* obtained is not a logical phrase, there is a high degree of correlation with *crown jewel* in prompt, which also reflects the relevant knowledge. And also, the threshold ensures that even such lapses form merely a minor part of the distilled KG.

Table 12: KG+LM models interpret the same KG differently.

(a) QA-GNN

Source	Relation	Target
Oaks	***RelatedTo***	*Bush* *Timber* *Wooded* *Acorns* *Forested Area* *Wooded Area* *Thick* *Leaves* *Large*

(b) GreaseLM

Source	Relation	Target
Oaks	***RelatedTo***	*Often Green*
	BelongsTo	*Integral*
	Affects	*Lifeforms* *Potential Energy*
	UsedFor	*Ecosystem* *Nourishment*
	Causes	*Evergreen* *Meadows* *Long Cabins*

Differences in interpretation of KGs. Experiments in Section 5 have shown that our distilled KGs reflect the knowledge ability of the base models quite closely. Accordingly, when distilling knowledge from the same source KG, distilled KGs can still vary based on the KG+LM approach used. As mentioned in Section 5, on the OBQA task, GreaseLM shows an outstanding accuracy of 83.8%, significantly surpassing QA-GNN's score of 67.2%. Meanwhile, as shown in Table 12, when distilling knowledge related to the source entity, *Oaks*, QA-GNN tends to generalize relations, resulting in a weaker knowledge connection to the target entities. Conversely, GreaseLM captures more detailed relations about the entities, which may serve as a rationale of GreaseLM's superior capability to leverage knowledge from KGs. It is essential to understand that distilled KGs serve as lens, revealing how the models interpret and utilize the original KGs with the augmentation method applied.

6 Discussion

Impact of τ and HOP_{max}. For the hyperparameters τ and HOP_{max}, we conduct repeated experiments on the GreaseLM model to evaluate the effect of different values on the number of queries and the performance of distilled KG. The results are shown in Table 13.

For τ, setting it too large will retain only entity nodes with higher confidence and reduce the number of queries, but accordingly, a part of entity-relation edges containing valid knowledge will also be filtered, leading to a decrease in the performance of distilled KG; too small τ will retain a part of entity nodes with lower confidence, which will lead to a great rise in the number of queries, but since most of the content of this part of the entities are noise content, the performance rise of distilled KG is not obvious. Therefore, we choose a more moderate value of τ as we have discussed.

For HOP_{max}, we find that proper pruning contributes to the rise in distilled KG performance, which is due to cutting out some of the redundant information that may interfere with the task performance. However, beyond some fixed threshold, the change in the number of clipped edges is no longer significant. This is due to the fact that semantic logic that is too complex for LM becomes inherently less easy to generate. Due to the low cost of pruning, it can be set easily by preliminary experiments on sample batches.

Table 13: Impact of Hyper-parameters.

Param.	Value	Edges	Queries	Acc (%)
-	Victim	2,487,810	-	83.9
τ	τ = 10	28,124	3.7M	83.5
	τ = 15 (Ori)	26,052	2.4M	83.2
	τ = 20	20,981	2.0M	82.9
HOP_{max}	No Pruning	31,465	2.4M	81.7
	HOP_{max} = 5	30,126	2.4M	81.8
	HOP_{max} = 4	29,012	2.4M	82.8
	HOP_{max} = 3 (Ori)	26,052	2.4M	83.1
	HOP_{max} = 2	11,584	2.4M	81.3

Distinction between knowledge from LM and KG. Our approach hinges on the notion that the KG+LM augmentation proves effective by giving higher confidence to information directly from KG, compared to LM alone. To demonstrate differentiation between knowledge sources, we repeat with GreaseLM in Case 1, but without integrating the KG on the base model. The results, as shown in Table 14, speak volumes. In the absence of KG integration, the distilled KG contains only 201 relation edges, a mere 0.77% compared to when the KG is included. This stark contrast underscores that the lion's share of knowledge in the distilled KG is derived from the original KG, not LM. This solidifies the effectiveness of our threshold in distinguishing between the two knowledge sources.

Table 14: Comparison with plain base model.

Approach	Source	Entities (Rate %)	Edges (Rate %)
GreaseLM	Original	799,273	2,487,810
	LM + KG	5,289 (0.66%)	26,052 (1.05%)
	LM Only	5,289 (0.66%)	201 (0.01%)

Table 15: Variations on distillation relation types.

Original	Less Types	Changed Words	Coarse Only
RelatedTo	*RelatedTo*	*About*	*RelatedTo*
Causes	-	*Generates*	-
BelongsTo	*BelongsTo*	*IsOf*	-
IsA	*IsA*	*InstanceOf*	-
UsedFor	-	*IsFor*	-
Affects	*Affects*	*Influences*	-

Impact of attacker's knowledge. While the attacker aims to extract domain knowledge, there is some prior knowledge about the entities and relations to distill. To account for potential impact on outcomes, we revisit the GreaseLM case and conduct experiments with different initial entities and relation types settings. For entities, we test with different size of initial entity sets. For relations, we introduce vocabulary variations and also consider purely coarse-grained relations for comparison (see Table 15).

For entities, the result is showcased in Table 16a. Results show, even if the attacker only knows 0.33% proportion of the entities in the original KG that are relevant to the task, the resulting distillation profile has only a 2% performance degradation compared to the original KG. For relation types, as is summarized in Table 16b, a broader range of relation types may slightly enhance performance. Interestingly, however, even coarse-grained distillation manages to capture most of the key KG knowledge, with task performance dropping by no more than 1.04%. Similarly, vocabulary variations introduce performance fluctuations, but these remain within a margin of 2.64%. These results can be attributed to two key facets. Firstly, the graph structure of the KG ensures that the logical interconnections between entity nodes remain largely impervious to the nuances of edge types. Secondly, the intrinsic ability of the LM to interpret semantics ensures that knowledge is extracted accurately, even when the employed vocabulary doesn't align perfectly. Overall, there is a limited effect on how much initial knowledge is acquired on our distillation performance. Only a small amount of initial knowledge is needed for our attack to work well.

Table 16: Impact of attacker's knowledge.

(a) Initial Entities

Init Ent Size	Edges (Rate %)	Acc (%)
Victim	2,487,810	83.90
5.3K (0.67%)	26,052 (0.45%)	83.10
4.0K (0.50%)	22,981 (0.38%)	82.72
2.6K (0.33%)	14,721 (0.28%)	81.94

(b) Relation Types

Source	Edges (Rate %)	Acc (%)
Victim	2,487,810	83.90
Distilled	26,052 (1.05%)	83.20
Less Types	24,775 (1.00%)	83.12
Coarse Only	21,932 (0.88%)	82.86
Changed Words	18,723 (0.75%)	81.26

Impact of Base Model. As discussed in Section 5.6, even for the same original knowledge graph, there is variability in the efficiency of knowledge utilization of different models or enhancement methods. As a result, the performance of distilled knowledge graphs acquired by our attack method is also affected. Meanwhile, it is notable that the goal of our attack is that the distilled KG augmented model combination and the original combination are consistent in performance, when the original model itself performs poorly, it is natural that the KG we obtain performs poorly, reflecting the original performance. On the other hand, for attackers, the goal of their attack is to obtain the part of augmented knowledge from the augmentation model that is actually effective. If the base model itself does not make good use of knowledge graph information, attacking such underperforming models is itself of little significance.

Applicability on LLMs. Recently, LLM knowledge graph enhancement techniques such as GraphRAG [16], MindMap [72], etc. have emerged, all of which have better utilization efficiency for knowledge graphs. According to the previous discussion, when the performance of the base model is strong, the acquired knowledge graph also has better performance. Meanwhile, our attack method requires only the model's output API, and the attack performance is proportional to the model's performance and independent of the model's architecture. Although we do not conduct further experiments on these models due to experimental constraints, our approach may still have similar results on such models.

Practical Cost. Since many commercial black-box model APIs now charge users according to the number of tokens they access, conducting such an attack also requires taking into account realistic monetary overheads. We evaluate the practical monetary cost for applying such an attack, since we use short assertion (Table 10), our query is generally no longer than 15-20 token. For ChatGPT-3.5 model ($0.0010 / 1K tokens), the cost for our most complex model (KEPLER, 5.8M queries) is only $87-$116, well within the acceptable range. This proves the practicality of our approach with good cost control in real scenarios as well.

Limitation. Similar to black-box model distillation, the attacker is often limited by the API access. On the input side, for instance, in Case 1, the API might restrict the option numbers for the multiple-choice model. Consequently, our prompts must be adjusted to work within these constraints. On the output side, when the API returns less information for each query, we might need more queries. Another major issue is that in some situations, confidence might not be available. We rely on them for two reasons: they guide the entity query order, and determine which entities to retain.

However, without them, we can adapt in a couple of ways. First, if the target model provides multiple options to a query, we can select entities based on their inherent output ranking to bypass the need for confidence-based ordering. In real-world applications, models with hard-label outputs may have underlying threshold mechanisms. For instance, recommendation systems typically display only the top results with the highest confidence, which can be exploited by our attack. If even this information is not available, we might end up including more low-confidence entities, leading to more queries. Yet, our primary objective, which is extracting knowledge from the original KG, remains achievable.

Though our approach is much better than brute-force, it still requires a significant number of queries. This is because the number of nodes and relation types in the original knowledge graph itself is of a very large scale. The query efficiency maybe improved by concatenating multiple entities to construct a prompt, and then filtering to see the role weights that different entities occupy in the prediction results, which can be our further work.

Defense. For those considering potential countermeasures, it's worth noting that our attack method can be influenced by certain API output behaviors. If the model were to shuffle the order of candidate results, our entity selection priority during distillation could be disrupted. The hard-label output means discussed in the previous section can also be used to defend against our attack. Additionally, reducing the weight of KG information in the output might impede our attack. However, it's essential to realize that while these defenses might reduce the efficiency of our attack, they cannot fully block the extraction of knowledge from original KGs. We have carried out KG distillation using only hard-label data on both GreaseLM and QA-GNN methods to simulate the possible defense approaches, and the results are shown in Table 17:

Table 17: Impact of Hard-Label Defense.

Approach	Source	Edges	Queries	Acc (%)
GreaseLM	Victim	2,487,810	-	83.9
	Distilled	26,052	2.4M	83.2
	Hard-Label	19,889	5.7M	82.4
QA-GNN	Victim	2,487,810	-	67.8
	Distilled	11,084	1.5M	67.2
	Hard-Label	9,817	3.9M	66.8

The results show that using hard labels increases our query counts, but as far as the task performance of the KG is concerned, the performance of our refined KG remains similar to that of the original KG, which proves that our attack is robust to this defense. Meanwhile, as shown in Table 6, even when efficiency is compromised, valid queries during the same time amount may be greatly reduced, a significant portion of the knowledge can still be extracted at the starting rounds. More importantly, such defensive measures might compromise the model's accuracy and usefulness.

7 Conclusion

In this paper, we propose KGDist, the first knowledge graph distillation attack against language models augmented with knowledge graphs. Our approach involves the appropriate selection of entities for prompt construction and querying, the filtering of relation edges utilizing a confidence threshold, iterative updates to query entities, and the application of pruning and merging to the distilled KG. Experimental results demonstrate that KGDist effectively transfers domain knowledge from victim KG to distilled KG, maintaining augmented models' performance. We conclusively show that our method adeptly differentiates between the KG knowledge and extraneous noise. Furthermore, the distilled KGs keep similar graph properties compared with the original ones.

Acknowledgments

We thank the Shepherd and reviewers for their constructive feedback. The authors are supported in part by NSFC (92270204) and Youth Innovation Promotion Association CAS.

References

[1] Dimitrios Alivanistos, Selene B'aez Santamar'ia, Michael Cochez, Jan-Christoph Kalo, Emile van Krieken, and Thiviyan Thanapalasingam. 2022. Prompting as Probing: Using Language Models for Knowledge Base Construction. *ArXiv* abs/2208.11057 (2022).

[2] Michael Atkin. 2023. Knowledge graph implementation: Costs and obstacles to consider. https://www.ontotext.com/knowledgehub/white_paper/knowledge-graph-implementation-costs-and-obstacles-to-consider/

[3] Prithu Banerjee, Lingyang Chu, Yong Zhang, Laks V.S. Lakshmanan, and Lanjun Wang. 2021. Stealthy Targeted Data Poisoning Attack on Knowledge Graphs. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. 2069–2074. https://doi.org/10.1109/ICDE51399.2021.00202

[4] Peru Bhardwaj, John D. Kelleher, Luca Costabello, and Declan O'Sullivan. 2021. Adversarial Attacks on Knowledge Graph Embeddings via Instance Attribution Methods. *ArXiv* abs/2111.03120 (2021).

[5] Olivier Bodenreider. 2004. The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic acids research* 32 Database issue (2004), D267–70.

[6] Sergey Brin and Lawrence Page. 1998. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput. Networks* 30 (1998), 107–117. https://api.semanticscholar.org/CorpusID:7587743

[7] Nicholas Carlini, Daniel Paleka, Krishnamurthy Dvijotham, Thomas Steinke, Jonathan Hayase, A. Feder Cooper, Katherine Lee, Matthew Jagielski, Milad Nasr, Arthur Conmy, Eric Wallace, David Rolnick, and Florian Tramèr. 2024. Stealing Part of a Production Language Model. *ArXiv* abs/2403.06634 (2024). https://api.semanticscholar.org/CorpusID:268357903

[8] Graham Caron and Shashank Srivastava. 2022. Identifying and Manipulating the Personality Traits of Language Models. *ArXiv* abs/2212.10276 (2022). https://api.semanticscholar.org/CorpusID:254877016

[9] Donghyun Choi, Myeong Cheol Shin, EungGyun Kim, and Dong Ryeol Shin. 2021. OutFlip: Generating Examples for Unknown Intent Detection with Natural Language Attack. In *Findings*.

[10] Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-Fine Entity Typing. *ArXiv* abs/1807.04905 (2018).

[11] Gautam Chutani. 2024. Unlocking LLM confidence through Logprobs. https://gautam75.medium.com/unlocking-llm-confidence-through-logprobs-54b26ed1b48a

[12] Peter Clark, Oren Etzioni, Daniel Khashabi, Tushar Khot, Bhavana Dalvi, Kyle Richardson, Ashish Sabharwal, Carissa Schoenick, Oyvind Tafjord, Niket Tandon, Sumithra Bhakthavatsalam, Dirk Groeneveld, Michal Guerquin, and Michael Schmitz. 2019. From 'F' to 'A' on the N.Y. Regents Science Exams: An Overview of the Aristo Project. *AI Mag.* 41 (2019), 39–53.

[13] Giannis Daras and Alexandros G. Dimakis. 2022. Discovering the Hidden Vocabulary of DALLE-2. *ArXiv* abs/2206.00169 (2022).

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *ArXiv* abs/1810.04805 (2019).

[15] Jialiang Dong, Shen Wang, Longfei Wu, Huoyuan Dong, and Zhitao Guan. 2022. A Textual Adversarial Attack Scheme for Domain-Specific Models. In *International Conference on Machine Learning for Cyber Security*.

[16] Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130* (2024).

[17] FactNexus EKG. 2023. *Knowledge Graph Enterprise.* https://kgkg.factnexus.com/@3782~167.html/

[18] Stefano Faralli, Irene Finocchi, Simone Paolo Ponzetto, and Paola Velardi. 2018. Efficient Pruning of Large Knowledge Graphs. In *International Joint Conference on Artificial Intelligence*.

[19] Zhangyin Feng, Daya Guo, Duyu Tang, Nan Duan, Xiaocheng Feng, Ming Gong, Linjun Shou, Bing Qin, Ting Liu, Daxin Jiang, and Ming Zhou. 2020. CodeBERT: A Pre-Trained Model for Programming and Natural Languages. *ArXiv* abs/2002.08155 (2020).

[20] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2022. PAL: Program-aided Language Models. *ArXiv* abs/2211.10435 (2022).

[21] Tianyu Gao, Xu Han, Hao Zhu, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2019. FewRel 2.0: Towards More Challenging Few-Shot Relation Classification. In *Conference on Empirical Methods in Natural Language Processing*.

[22] Andrew V. Goldberg. 1987. Efficient graph algorithms for sequential and parallel computers. https://api.semanticscholar.org/CorpusID:37561100

[23] Thomas R. Gruber. 2008. Collective knowledge systems: Where the Social Web meets the Semantic Web. *J. Web Semant.* 6 (2008), 4–13. https://api.semanticscholar.org/CorpusID:14754882

[24] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-Augmented Language Model Pre-Training. *ArXiv* abs/2002.08909 (2020).

[25] Joy He-Yueya, Gabriel Poesia, Rose E. Wang, and Noah D. Goodman. 2023. Solving Math Word Problems by Combining Language Models With Symbolic Solvers. *ArXiv* abs/2304.09102 (2023).

[26] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *ArXiv* abs/1503.02531 (2015).

[27] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. TinyBERT: Distilling BERT for Natural Language Understanding. In *Findings*.

[28] Joaogante. 2023. [announcement] generation: Get probabilities for generated output. https://discuss.huggingface.co/t/announcement-generation-get-probabilities-for-generated-output/30075

[29] Darek Kleczek. 2023. A gentle introduction to LLM Apis. https://wandb.ai/darek/llmapps/reports/A-Gentle-Introduction-to-LLM-APIs--Vmlldzo0NjM0MTMz

[30] Judith Knoblach, Nikhil Acharya, Bhavya Koranemkattil, Andreas Both, and Diego Collarana. 2022. Combining Knowledge Graphs and Language Models to Answer Questions over Tables. In *International Conference on Semantic Systems*.

[31] Tatsuki Kuribayashi, Takumi Ito, Jun Suzuki, and Kentaro Inui. 2020. Language Models as an Alternative Evaluator of Word Order Hypotheses: A Case Study in Japanese. *ArXiv* abs/2005.00842 (2020). https://api.semanticscholar.org/CorpusID:218487721

[32] Aapo Kyrola, Guy Blelloch, and Carlos Guestrin. 2012. GraphChi: Large-Scale Graph Computation on Just a PC. In *10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. USENIX Association, Hollywood, CA, 31–46. https://www.usenix.org/conference/osdi12/technical-sessions/presentation/kyrola

[33] Matthew Landen, Key whan Chung, Moses Ike, Sarah Mackay, Jean-Paul Watson, and Wenke Lee. 2022. DRAGON: Deep Reinforcement Learning for Autonomous Grid Operation and Attack Detection. *Proceedings of the 38th Annual Computer Security Applications Conference* (2022). https://api.semanticscholar.org/CorpusID:254151725

[34] Chen Li, Xutan Peng, Yuhang Niu, Shanghang Zhang, Hao Peng, Chuan Zhou, and Jianxin Li. 2021. Learning graph attention-aware knowledge graph embedding. *Neurocomputing* 461 (2021), 516–529. https://api.semanticscholar.org/CorpusID:238417287

[35] Zongjie Li, Chaozheng Wang, Pingchuan Ma, Chaowei Liu, Shuai Wang, Daoyuan Wu, and Cuiyun Gao. 2023. On Extracting Specialized Code Abilities from Large Language Models: A Feasibility Study. *2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)* (2023), 893–905. https://api.semanticscholar.org/CorpusID:257365453

[36] Jinzhi Liao, Xiang Zhao, Jiuyang Tang, Weixin Zeng, and Zhen Tan. 2021. To hop or not, that is the question: Towards effective multi-hop reasoning over

knowledge graphs. *World Wide Web* 24 (2021), 1837–1856.

[37] Vivian Liu and Lydia B. Chilton. 2021. Design Guidelines for Prompt Engineering Text-to-Image Generative Models. *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (2021).

[38] 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. *ArXiv* abs/1907.11692 (2019).

[39] Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ramakanth Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented Language Models: a Survey. *ArXiv* abs/2302.07842 (2023).

[40] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. In *Conference on Empirical Methods in Natural Language Processing*.

[41] Safikureshi Mondal and Nandini Mukherjee. 2018. A BFS-Based Pruning Algorithm for Disease-Symptom Knowledge Graph Database. *Information and Communication Technology for Intelligent Systems* (2018). https://api.semanticscholar.org/CorpusID:69298160

[42] Reiichiro Nakano, Jacob Hilton, S. Arun Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2021. WebGPT: Browser-assisted question-answering with human feedback. *ArXiv* abs/2112.09332 (2021).

[43] Ehsan Nowroozi, Abhishek, Mohammadreza Mohammadi, and Mauro Conti. 2022. An Adversarial Attack Analysis on Malicious Advertisement URL Detection Framework. *IEEE Transactions on Network and Service Management* 20 (2022), 1332–1344. https://api.semanticscholar.org/CorpusID:248427252

[44] Onotext. 2023. *What is a Knowledge Graph?* https://www.ontotext.com/knowledgehub/fundamentals/what-is-a-knowledge-graph/

[45] OpenAI. 2023. GPT-4 Technical Report. *ArXiv* abs/2303.08774 (2023).

[46] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. 2018. Knockoff Nets: Stealing Functionality of Black-Box Models. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), 4949–4958.

[47] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank Citation Ranking : Bringing Order to the Web. In *The Web Conference*. https://api.semanticscholar.org/CorpusID:1508503

[48] Nicolas Papernot, Patrick Mcdaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2016. Practical Black-Box Attacks against Machine Learning. *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (2016).

[49] Sarah Perez. 2012. Wikipedia's next big thing: Wikidata, a machine-readable, user-editable database funded by Google, Paul Allen and others. https://techcrunch.com/2012/03/30/wikipedias-next-big-thing-wikidata-a-machine-readable-user-editable-database-funded-by-google-paul-allen-and-others/

[50] Matthew E. Peters, Mark Neumann, IV RobertL.Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge Enhanced Contextual Word Representations. In *Conference on Empirical Methods in Natural Language Processing*.

[51] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language Models as Knowledge Bases? *ArXiv* abs/1909.01066 (2019).

[52] Mohammad Taher Pilehvar and José Camacho-Collados. 2018. WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations. In *North American Chapter of the Association for Computational Linguistics*.

[53] Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. E-BERT: Efficient-Yet-Effective Entity Embeddings for BERT. In *Findings*.

[54] Vinit Ravishankar, Mostafa Abdou, Artur Kulmizev, and Anders Søgaard. 2022. Word Order Does Matter and Shuffled Language Models Know It. In *Annual Meeting of the Association for Computational Linguistics*. https://api.semanticscholar.org/CorpusID:247594659

[55] Randy Rockinson. 2023. https://blog.google/products/shopping/shopping-graph-explained/

[56] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David A. Sontag. 2017. Learning a Health Knowledge Graph from Electronic Medical Records. *Scientific Reports* 7 (2017).

[57] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *ArXiv* abs/1910.01108 (2019).

[58] Ilia Shumailov, Yiren Zhao, Daniel Bates, Nicolas Papernot, Robert D. Mullins, and Ross Anderson. 2020. Sponge Examples: Energy-Latency Attacks on Neural Networks. *2021 IEEE European Symposium on Security and Privacy (EuroS&P)* (2020), 212–231.

[59] Robyn Speer, Joshua Chin, and Catherine Havasi. 2016. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. *ArXiv* abs/1612.03975 (2016).

[60] Hendrik Strobelt, Albert Webson, Victor Sanh, Benjamin Hoover, Johanna Beyer, Hanspeter Pfister, and Alexander M. Rush. 2022. Interactive and Visual Prompt Engineering for Ad-hoc Task Adaptation with Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* 29 (2022), 1146–1156.

[61] Vinitra Swamy, Angelika Romanou, and Martin Jaggi. 2021. Interpreting Language Models Through Knowledge Graph Extraction. *ArXiv* abs/2111.08546 (2021).

[62] Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy J. Lin. 2019. Distilling Task-Specific Knowledge from BERT into Simple Neural Networks. *ArXiv* abs/1903.12136 (2019). https://api.semanticscholar.org/CorpusID:85543565

[63] Rakshit S. Trivedi, Hanjun Dai, Yichen Wang, and Le Song. 2017. Know-Evolve: Deep Temporal Reasoning for Dynamic Knowledge Graphs. In *International Conference on Machine Learning*.

[64] Mike Tung. 2023. Grounded natural language generation with knowledge graphs. https://blog.diffbot.com/grounded-natural-language-generation-with-knowledge-graphs/

[65] Jacopo Urbani, Sourav Dutta, Sairam Gurajada, and Gerhard Weikum. 2016. KOGNAC: Efficient Encoding of Large Knowledge Graphs. *ArXiv* abs/1604.04795 (2016).

[66] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM* 57 (2014), 78–85.

[67] Qingyun Wang, Manling Li, Xuan Wang, Nikolaus Nova Parulian, Guangxing Han, Jiawei Ma, Jingxuan Tu, Ying Lin, H. Zhang, Weili Liu, Aabhas Chauhan, Yingjun Guan, Bangzheng Li, Ruisong Li, Xiangchen Song, Heng Ji, Jiawei Han, Shih-Fu Chang, James Pustejovsky, David Liem, Ahmed Elsayed, Martha Palmer, Jasmine Rah, Cynthia Schneider, and Boyan A. Onyshkevych. 2020. COVID-19 Literature Knowledge Graph Construction and Drug Repurposing Report Generation. In *North American Chapter of the Association for Computational Linguistics*.

[68] Rui Wang, Yongkun Li, Hong Xie, Yinlong Xu, and John C.S. Lui. 2020. GraphWalker: An I/O-Efficient and Resource-Friendly Graph Analytic System for Fast and Scalable Random Walks. In *USENIX Annual Technical Conference*. https://api.semanticscholar.org/CorpusID:220657879

[69] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters. In *Findings*.

[70] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. *ArXiv* abs/2002.10957 (2020). https://api.semanticscholar.org/CorpusID:211296536

[71] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juan-Zi Li, and Jian Tang. 2019. KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation. *Transactions of the Association for Computational Linguistics* 9 (2019), 176–194.

[72] Yilin Wen, Zifeng Wang, and Jimeng Sun. 2023. Mindmap: Knowledge graph prompting sparks graph of thoughts in large language models. *arXiv preprint arXiv:2308.09729* (2023).

[73] Wikidata. 2023. Wikidata:statistics - wikidata. https://www.wikidata.org/wiki/Wikidata:Statistics

[74] Wikipedia contributors. 2022. Open Mind Common Sense — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Open_Mind_Common_Sense&oldid=1124381133 [Online; accessed 13-May-2023].

[75] Chao Wu, Zeyu Zeng, Yajing Yang, Mao Chen, Xicheng Peng, and Sannyuya Liu. 2023. Task-driven cleaning and pruning of noisy knowledge graph. *Inf. Sci.* 646 (2023), 119406. https://api.semanticscholar.org/CorpusID:260069888

[76] Bo Xu, Jiaqing Liang, Chenhao Xie, Bin Liang, Lihan Chen, and Yanghua Xiao. 2019. CN-DBpedia2: An Extraction and Verification Framework for Enriching Chinese Encyclopedia Knowledge Base. *Data Intelligence* 1 (2019), 271–288.

[77] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *Neural Information Processing Systems*.

[78] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 20744–20757. https://proceedings.neurips.cc/paper_files/paper/2022/file/82ad13ec01f9fe44c01cb91814fd7b8c-Paper-Conference.pdf

[79] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. *ArXiv* abs/2210.03629 (2022).

[80] Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D. Manning, Percy Liang, and Jure Leskovec. 2022. Deep Bidirectional Language-Knowledge Graph Pretraining. *ArXiv* abs/2210.09338 (2022). https://api.semanticscholar.org/CorpusID:252968266

[81] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering. *ArXiv* abs/2104.06378 (2021).

[82] Shan You, Chang Xu, Chao Xu, and Dacheng Tao. 2018. Learning With Single-Teacher Multi-Student. In *AAAI Conference on Artificial Intelligence*.

[83] Xia Zheng You, Beina Sheng, Daizong Ding, Mi Zhang, Xudong Pan, Min Yang, and Fuli Feng. 2023. MaSS: Model-agnostic, Semantic and Stealthy Data Poisoning

Attack on Knowledge Graph Embedding. *Proceedings of the ACM Web Conference 2023* (2023).

[84] Hengtong Zhang, T. Zheng, Jing Gao, Chenglin Miao, Lu Su, Yaliang Li, and Kui Ren. 2019. Data Poisoning Attack against Knowledge Graph Embedding. In *International Joint Conference on Artificial Intelligence*.

[85] Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. 2021. Differentiable Prompt Makes Pre-trained Language Models Better Few-shot Learners. *ArXiv* abs/2108.13161 (2021).

[86] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. OPT: Open Pre-trained Transformer Language Models. *ArXiv* abs/2205.01068 (2022).

[87] Xikun Zhang, Antoine Bosselut, Michihiro Yasunaga, Hongyu Ren, Percy Liang, Christopher D. Manning, and Jure Leskovec. 2022. GreaseLM: Graph REASoning Enhanced Language Models. In *International Conference on Learning Representations*.

[88] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In *Annual Meeting of the Association for Computational Linguistics*.

[89] Zeru Zhang, Zijie Zhang, Yang Zhou, Lingfei Wu, Sixing Wu, Xiaoying Han, Dejing Dou, Tianshi Che, and Da Yan. 2021. Adversarial Attack against Cross-lingual Knowledge Graph Alignment. In *Conference on Empirical Methods in Natural Language Processing*.

[90] Shangfei Zheng, Wei Chen, Pengpeng Zhao, An Liu, Junhua Fang, and Lei Zhao. 2021. When Hardness Makes a Difference: Multi-Hop Knowledge Graph Reasoning over Few-Shot Relations. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (2021).

[91] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2021. Learning to Prompt for Vision-Language Models. *International Journal of Computer Vision* 130 (2021), 2337 – 2348.