BertNet: Harvesting Knowledge Graphs with Arbitrary Relations from Pretrained Language Models

Shibo Hao[1]*, **Bowen Tan**[2]*, **Kaiwen Tang**[1]*, **Bin Ni**[1], **Xiyan Shao**[1], **Hengzhe Zhang**[1], **Eric P. Xing**[2,3], **Zhiting Hu**[1]
[1]UC San Diego, [2]Carnegie Mellon University,
[3]Mohamed bin Zayed University of Artificial Intelligence
{s5hao,zhh019}@ucsd.edu, {btan2}@cs.cmu.edu

Abstract

It is crucial to automatically construct knowledge graphs (KGs) of diverse new relations to support knowledge discovery and broad applications. Previous KG construction methods, based on either crowdsourcing or text mining, are often limited to a small predefined set of relations due to manual cost or restrictions in text corpus. Recent research proposed to use pretrained language models (LMs) as implicit knowledge bases that accept knowledge queries with prompts. Yet, the implicit knowledge lacks many desirable properties of a full-scale symbolic KG, such as easy access, navigation, editing, and quality assurance. In this paper, we propose a new approach of harvesting massive KGs of *arbitrary* relations from pretrained LMs. With minimal input of a relation definition (a prompt and a few shot of example entity pairs), the approach efficiently searches in the vast entity pair space to extract diverse accurate knowledge of the desired relation. We develop an effective search-and-rescore mechanism for improved efficiency and accuracy. We deploy the approach to harvest KGs of over 400 new relations from different LMs. Extensive human and automatic evaluations show our approach manages to extract diverse accurate knowledge, including tuples of complex relations (e.g., `"A is capable of but not good at B"`). The resulting KGs as a symbolic interpretation of the source LMs also reveal new insights into the LMs' knowledge capacities.

1 Introduction

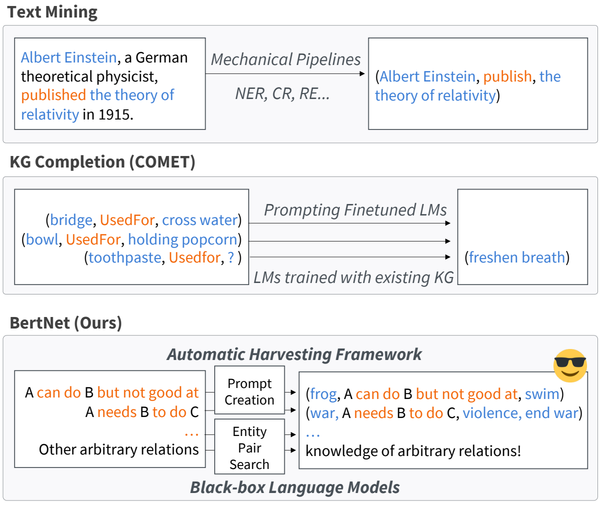

Text Mining
Albert Einstein, a German theoretical physicist, published the theory of relativity in 1915.
Mechanical Pipelines
NER, CR, RE...
(Albert Einstein, publish, the theory of relativity)
KG Completion (COMET)
(bridge, UsedFor, cross water)
(bowl, UsedFor, holding popcorn)
(toothpaste, Usedfor, ?)
Prompting Finetuned LMs
LMs trained with existing KG
(freshen breath)
BertNet (Ours)
Automatic Harvesting Framework
A can do B but not good at
A needs B to do C
...
Other arbitrary relations
Prompt Creation
Entity Pair Search
(frog, A can do B but not good at, swim)
(war, A needs B to do C, violence, end war)
...
knowledge of arbitrary relations!
Black-box Language Models

Figure 1: Different example paradigms of harvesting knowledge. *Text mining* extracts knowledge of relations explicitly mentioned in the text. *KG completion* produces tail entities to complete knowledge of preexisting relations. Our method is capable of harvesting knowledge of arbitrary new relations from LMs.

Symbolic knowledge graphs (KGs) are a powerful tool for indexing rich knowledge about entities and their relationships, and are useful for information access (Google, 2012), decision making (Yang et al., 2021; Santos et al., 2022), and improving machine learning in general (Li et al., 2019; Wang et al., 2019; Tan et al., 2020; Xiong et al., 2017).

It has been a long-term desire to construct KGs of diverse *relations* to comprehensively characterize the structures between entities. The traditional crowdsourcing-based approach (Speer et al., 2017; Fellbaum, 2000; Sap et al., 2019) tends to cover only a restricted relation set, such as ConceptNet (Speer et al., 2017) that contains a small set of 34 relations. The popular method based on text mining (Luan et al., 2019; Zhong and Chen, 2020; Wang et al., 2021b) has a similar limitation, as the text understanding models can often recognize only a predefined set of relations included in training data. Some open-schema text mining approaches (e.g., based on syntactic patterns) exist (Tandon et al., 2014; Romero et al., 2019; Zhang et al., 2020b; Nguyen et al., 2021), yet the extracted relations are limited to those explicitly stated in the text, missing all others that are not mentioned or do not have exact match with the text in the corpus. Similarly, KG completion approaches (Bordes et al., 2013; Bosselut et al., 2019; Yao et al., 2019) is restricted

*Equal contribution. Code available at `https://github.com/tanyuqian/knowledge-harvest-from-lms`. Demo available at `https://lmnet.io`

Findings of the Association for Computational Linguistics: ACL 2023, pages 5000–5015
July 9-14, 2023 ©2023 Association for Computational Linguistics

Method	Module(s)	Outcome	Arbitrary relation
Text mining (Zhang et al., 2020a; Nguyen et al., 2021)	NER, CR, RE, etc.[1]	KG	✗
LAMA (Petroni et al., 2019), LPAQA (Jiang et al., 2020)	LMs	tail entity	✓
COMET (Bosselut et al., 2019)	Finetuned GPT-2	tail entity	✗
Symbolic Knowledge Distillation (West et al., 2022)	GPT-3	KG	✓[2]
BertNet (ours)	LMs	KG	✓

Table 1: Categorization of works on automatic knowledge extraction. Compared to other categories of approaches, our method extracts full *explicit* KGs of *arbitrary new relations* from *any* LMs.

to the preexisting relations (Figure 1).

On the other hand, large language models (LMs) pretrained on massive text corpus, such as BERT (Devlin et al., 2019) and GPT-3 (Brown et al., 2020), have been found to encode a significant amount of knowledge implicitly in their parameters. Recent research attempted to use LMs as flexible knowledge bases by querying the LMs with arbitrary prompts (e.g., `"Obama was born in ___"` for the answer `"Hawaii"`) (Petroni et al., 2019). However, such implicit query-based knowledge falls short of many desirable properties of a full-scale KG such as ConceptNet (AlKhamissi et al., 2022), including easy access, browsing, or even editing (Zhu et al., 2020; Cao et al., 2021), as well as assurance of knowledge quality thanks to the symbolic nature (Anderson et al., 2020). Symbolic Knowledge Distillation (SKD, West et al., 2022) explicitly extracts a knowledge base from GPT-3. However, the approach exclusively relies on the strong in-context learning capability of GPT-3 and thus is not applicable to other rich LMs such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019). Moreover, its use of a quality discriminator trained on existing KGs can limit its generalization to new relations not included in the training data.

In this paper, we propose a new approach of harvesting massive KGs of arbitrary new relations from any pretrained LMs. Given minimal user input of a relation definition, including a prompt and a few shot of example entity pairs, our approach automatically searches within the LM to extract an extensive set of high-quality knowledge about the desired relation. To ensure search efficiency in the vast space of entity pairs, we devise an effective search-and-rescore strategy. We also adapt the previous prompt paraphrasing mechanism (Jiang et al., 2020; Newman et al., 2021) and enhance with our new rescore strategy for prompt weighting, leading to consistent and accurate outcome knowledge.

We apply our approach on a range of LMs of varying capacities, such as RoBERTa, BERT, and DistilBERT. In particular, we harvest knoweldge of over 400 new relations (an order of magnitude more than ConceptNet relations) not available in preexisting KGs and previous extraction methods. Extensive human and automatic evaluations show our approach successfully extracts diverse accurate knowledge, including tuples for complex relations such as "`A is capable of, but not good at, B`" and 3-ary relations such as "`A can do B at C`". Interestingly, the resulting KGs also serve as a symbolic interpretation of the source LMs, revealing new insights into their knowledge capacities in terms of varying factors such as model size, pretraining strategies, and distillation.

2 Related Work

Knowledge graph construction Popular knowledge bases or KGs are usually constructed with heavy human labor. For example, WordNet (Fellbaum, 2000) is a lexical database that links words into semantic relations; ConceptNet (Speer et al., 2017) is a large commonsense knowledge graph presented as a set of knowledge triples; ATOMIC (Sap et al., 2019) is a crowd-sourced social commonsense KG of if-then statements. Recently, Automatic Knowledge Base Construction (AKBC) as a research focus has led to various approaches (summarized in Table 1). Text mining-based works aim for knowledge extraction from text. A typical information extraction system (Angeli et al., 2015) is composed of several sub-tasks like coreference resolution, named entity recognition, and relationship extraction. Some works on commonsense knowledge extraction include WebChild (Tandon et al., 2014), TransOMCS (Zhang et al., 2020a), DISCOS (Fang et al., 2021), Quasimodo (Romero et al., 2019), ASCENT (Nguyen et al., 2021). These ex-

[1] "NER", "CR", "RE" refer to "named entity recognition", "coreference resolution", "relation extraction", respectively.

[2] SKD has an optional filter that requires existing KG to finetune, which doesn't work for arbitrary relations.

traction pipelines are based on linguistic pattern, and involve complex engineering such as corpus selection, term aggregation, filtering, etc. Recent attempts also utilize LMs for AKBC. Wang et al. 2021a finetuned LMs for link prediction. Feldman et al. 2019; Bouraoui et al. 2020 utilized LMs to score entity pairs collected from the Internet or missing edges in existing KGs. COMET (Bosselut et al., 2019) is a generative LM trained to predict tail entities given head entities and relations. West et al. 2021 distill the knowledge in GPT-3 to a generative LM. By prompting GPT-3 (Brown et al., 2020) with examples, they produced ATOMIC$_{10x}$ to teach the student model. Yet, this method requires the strong few-shot learning ability of GPT-3 and is not generally applicable to most LMs. To the best of our knowledge, our framework is the first to construct a KG by extracting purely from an LM (with the minimal definition of relations as input). The new paradigm can also be seen as optimizing a symbolic KG with (pretrained) neural models as supervision (Hu and Xing, 2022), which inverts the conventional problem of using symbolic knowledge to learn neural networks (Hu et al., 2016).

LMs as knowledge bases Another line of works attempted to use LMs as knowledge bases (LAMA, Petroni et al. 2019). These works are also known as factual probing because they measured how much knowledge is encoded in LMs. This is usually implemented by prompting methods and leveraging the masked LM pretraining task. LPAQA (Jiang et al., 2020) proposes to use text mining and paraphrasing to find and select prompts to optimize the prediction of a single or a few correct tail entities, instead of extensively predicting all the valid entity pairs like in our framework. AutoPrompt (Shin et al., 2020), Qin and Eisner, 2021 and OPTIPrompt (Zhong et al., 2021) learn discrete or continuous prompts automatically with an additional training set. Though making prompts unreadable, these methods achieve higher accuracy on the knowledge probing tasks. Our framework differs from these works in that we aim to explicitly harvest knowledge graphs instead of measuring the knowledge in a simplified setting.

Consistency of LMs Consistency is a significant challenge for LMs, which stresses that they should not produce conflicting predictions across inference sessions. For example, models should behave invariantly under inputs with different surface forms but the same meaning. Elazar et al. 2021 analyzed the consistency of pretrained LMs with respect to factual knowledge. Jiang et al. 2020 used paraphrasing to improve factual probing. Newman et al. 2021 trains an additional layer on top of word embedding to improve consistency. Recently, consistency is also shown helpful to improve the reasoning ability of large LMs (Wang et al., 2022; Jung et al., 2022; Hao et al., 2023). In our framework, the extracted entity pairs for each relation are enforced to consistently satisfy a diverse set of prompts and regularized by several scoring terms.

3 Harvesting KGs from LMs

This section presents the proposed framework for extracting a relational KG from a given pretrained LM, where the LM can be arbitrary fill-in-the-blank models such as BERT (Devlin et al., 2019), RoBERTA (Liu et al., 2019), BART (Lewis et al., 2020), or GPT-3 (with appropriate instructions) (Brown et al., 2020). The KG consists of a set of knowledge tuples in the form ⟨HEAD ENTITY (h), RELATION (r), TAIL ENTITY (t)⟩. Our approach utilizes the LM to automatically harvest a large number of appropriate entity pairs $(h_1, t_1), (h_2, t_2), \ldots$, for every given relation r. This presents a more challenging problem than traditional LM probing tasks, which typically predict a single tail entity or a small number of valid tail entities given a head entity and relation.

Our approach for extracting knowledge tuples of a specific relation of interest, such as "potential_risk" as depicted in Figure 2, only requires minimal input information that defines the relation. This includes an initial prompt, such as "The potential risk of A is B" and a small number of example entity pairs, such as ⟨EATING CANDY, TOOTH DECAY⟩. The prompt provides the overall semantics of the relation, while the example entity pairs clarify possible ambiguities. For new relations not included in existing KGs, it is impractical to require a large set (e.g., hundreds) of example entity pairs as in previous knowledge probing or prompt optimization methods (Petroni et al., 2019; Jiang et al., 2020; Shi et al., 2019; Zhong et al., 2021). In contrast, our approach necessitates only a small number of example entity pairs, for example, as few as 2 in our experiments, which can easily be collected or written by users.

In the following sections, we describe the core

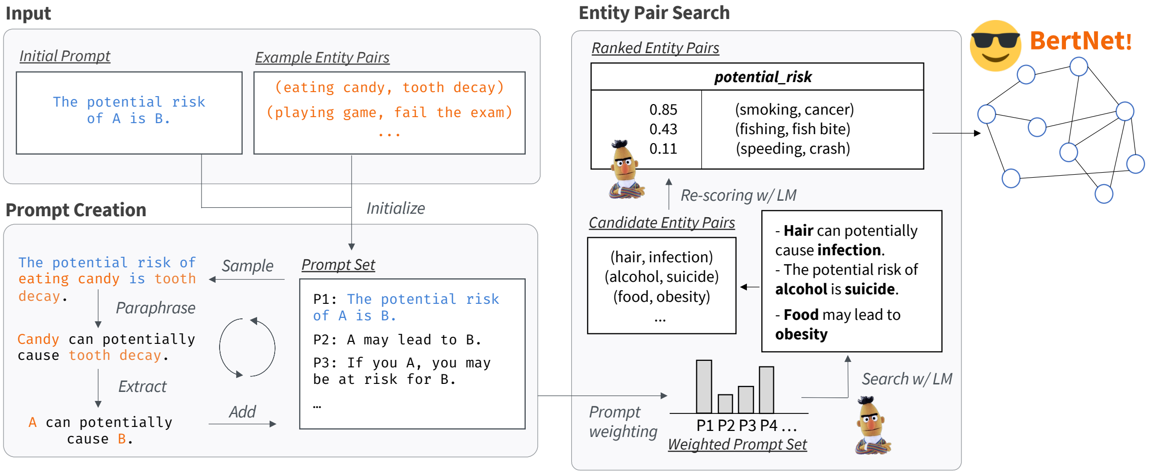

Input
Initial Prompt
The potential risk of A is B.
Example Entity Pairs
(eating candy, tooth decay)
(playing game, fail the exam)
...
Initialize
Prompt Creation
The potential risk of eating candy is tooth decay.
Sample
Paraphrase
Candy can potentially cause tooth decay.
Extract
A can potentially cause B.
Add
Prompt Set
P1: The potential risk of A is B.
P2: A may lead to B.
P3: If you A, you may be at risk for B.
...
Entity Pair Search
Ranked Entity Pairs
potential_risk
0.85 (smoking, cancer)
0.43 (fishing, fish bite)
0.11 (speeding, crash)
Re-scoring w/ LM
Candidate Entity Pairs
(hair, infection)
(alcohol, suicide)
(food, obesity)
...
- Hair can potentially cause infection.
- The potential risk of alcohol is suicide.
- Food may lead to obesity
Search w/ LM
Prompt weighting
P1 P2 P3 P4 ...
Weighted Prompt Set
BertNet!

Figure 2: An overview of the knowledge harvesting framework. Given the minimal definition of the relation as input (an initial prompt and a few shot of example entity pairs), the approach first automatically creates a set of prompts expressing the relation in a diverse ways (§3.1). The prompts are weighted with confidence scores. We then use the LM to search a large collection of candidate entity pairs, followed by re-scoring/ranking that yields the top entity pairs as the output knowledge (§3.2).

components of our approach, namely the automatic creation of diverse prompts with confidence weights (§3.1) and the efficient search to discover consistent entity pairs (§3.2) that compose the desired KGs. Figure 2 illustrate the overall framework.

3.1 Creating Diverse Weighted Prompts

Our automated approach utilizes input information, specifically the initial prompt and several example entity pairs, to generate a set of semantically consistent but linguistically diverse prompts for describing the relation of interest. The generated prompts are assigned confidence weights to accurately measure consistency of knowledge in the subsequent step (§3.2).

To generate diverse prompts for a desired relation, we begin by randomly selecting an entity pair from a example set and inserting it into an initial prompt to form a complete sentence. This sentence is then passed through an off-the-shelf text paraphrase model, which produces multiple paraphrased sentences with the same meaning. By removing the entity names, each paraphrased sentence results in a new prompt that describes the desired relation. To ensure a wide range of expressions of the relation, we retain only those prompts that are distinct from one another in terms of edit distance. This process is repeated by continuously paraphrasing the newly created prompts until a minimum of 10 prompts for the relation have been collected.

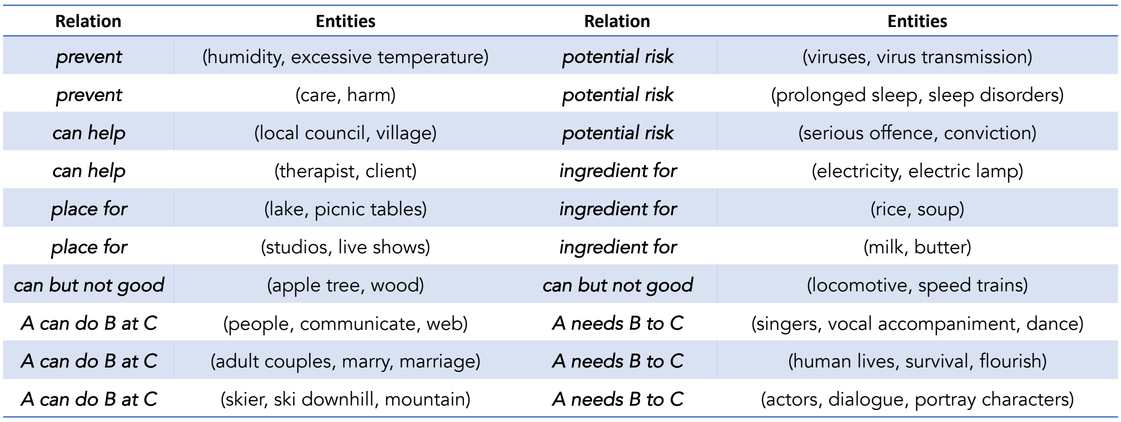
The automatic generation of prompts can be imprecise, resulting in prompts that do not accurately convey the intended relation. To mitigate this, we propose a reweighting method that utilizes compatibility scores to calibrate the impact of each prompt in the subsequent knowledge search step. Specifically, we evaluate the compatibility of new prompts with example entity pairs by measuring the likelihood of the prompts under a LM, considering both the individual entities and the entity pair as a whole. This allows us to determine the appropriate weights for each prompt and improve the precision of the knowledge search process. Formally, the compatibility score between an entity pair (h, t) and a prompt p can be written as:

$$\begin{aligned} f_{\mathrm{LM}}(\langle h, t\rangle, p) &= \alpha \log P_{\mathrm{LM}}(h, t \mid p) \\ &+ (1-\alpha) \min \{\log P_{\mathrm{LM}}(h \mid p), \log P_{\mathrm{LM}}(t \mid p, h)\} \end{aligned} \tag{1}$$

where the first term is the joint log-likelihood under the LM distribution P_{LM}, the second term is the minimum individual log-likelihood given the prompt (and the other entity), and α is a balancing factor ($\alpha = 2/3$ in our experiments). We compute the average compatibility score of each created prompt over all example entity pairs, and the weight of the prompt is then defined as the softmax-normalized score across all prompts.

3.2 Efficient Search for Consistent Knowledge

With the set of prompts and corresponding confidence weights obtained in the steps described in Section 3.1, we proceed to search entity pairs that consistently align with all prompts. To guide the searching process and evaluate the compatibility of searched-out entity pairs (h^{new}, t^{new}), we reuse the previously defined prompt/entity-pair compati-



Relation	Entities	Relation	Entities
prevent	(humidity, excessive temperature)	*potential risk*	(viruses, virus transmission)
prevent	(care, harm)	*potential risk*	(prolonged sleep, sleep disorders)
can help	(local council, village)	*potential risk*	(serious offence, conviction)
can help	(therapist, client)	*ingredient for*	(electricity, electric lamp)
place for	(lake, picnic tables)	*ingredient for*	(rice, soup)
place for	(studios, live shows)	*ingredient for*	(milk, butter)
can but not good	(apple tree, wood)	*can but not good*	(locomotive, speed trains)
A can do B at C	(people, communicate, web)	*A needs B to C*	(singers, vocal accompaniment, dance)
A can do B at C	(adult couples, marry, marriage)	*A needs B to C*	(human lives, survival, flourish)
A can do B at C	(skier, ski downhill, mountain)	*A needs B to C*	(actors, dialogue, portray characters)

Figure 3: Examples of knowledge tuples harvested from DISTILLBERT (randomly sampled). The first 7 rows shows relations with two entities (head and tail), and last 3 rows shows more complex relations with 3 entities.

bility function (Eq.1), and intuitively define consistency as the weighted average of its compatibility with the various prompts, i.e.,

$$\text{consistency}((h^{\text{new}}, t^{\text{new}})) = \sum\nolimits_p w_p \cdot f_{\text{LM}}((h^{\text{new}}, t^{\text{new}}), p) \tag{2}$$

where w_p is the prompt weight and the sum is over all automatically created prompts as above, so that entity pairs compatible with all prompts are considered to be consistent.

Based on the consistency criterion, we develop an efficient search strategy to search for consistent entity pairs. A straightforward approach involves enumerating all possible pairs of entities, calculating their respective consistency scores, and selecting the top-K entity pairs with the highest scores as the resulting knowledge. However, this approach can be computationally expensive due to the large vocabulary size V (e.g., $V = 50,265$ for ROBERTA) and the high time complexity of the enumeration process (i.e., $O(V^2)$ even when each entity consists of only one token). To overcome this limitation, we have proposed an appropriate approximation that leads to a more efficient *search and re-scoring* method. Specifically, we first use the minimum individual log-likelihoods (i.e., the second term in the compatibility score Eq.1) weighted averaged across different prompts (similar as in Eq.2), to propose a large set of candidate entity pairs. The use of the minimum individual log-likelihoods allows us to apply pruning strategies, such as maintaining a heap and eliminating entities ranked outside top-K in every single searching step. Once we have collected a large number of proposals, we re-rank them using the full consistency score in Eq.2 and select the top-K instances as the output knowledge. We describe more nuanced handling in the search procedure (e.g., the processing of multi-token entities, detailed pruning strategies) in the appendix.

Generalization to complex relations Most existing KGs or knowledge bases include relations that are predicates connecting two entities, e.g., `"A is capable of B"`. However, many real-life relations are more complex. Our approach is flexible and easily extensible to extract knowledge about these complex relations. We demonstrate this in our experiments by exploring two cases: (1) ***highly customized relations*** that have specific and sophisticated meanings, such as `"A is capable of, but not good at, B"`. This type of sophisticated knowledge is often difficult for humans to write down on a large scale. Our automatic approach naturally supports harvesting this kind of knowledge given only an initial prompt and a few example entities that can be collected easily, e.g., ⟨DOG, SWIM⟩, ⟨CHICKEN, FLY⟩, etc.; (2) ***N-ary relations*** involving more than two entities, such as `"A can do B at C"`. Our approach can straightforwardly be extended to handle n-ary relations by generalizing the compatibility score and search strategy accordingly to accommodate more than two entities.

Symbolic interpretation of neural LMs The harvested knowledge tuples, as consistently recognized across varying prompts by the LM, can be considered as the underlying "beliefs" of the LM about the world (Stich, 1979; Hase et al., 2021). These fully symbolic and interpretable tuples provide a means for easily browsing and analyzing the knowledge capabilities of the black-box LM. For example, via these outcome KGs, one can compare

Paradigm	Method (Size)	Relation Set	#Relations	Accuracy (%)	Novelty (%)
Ours	RobertaNet (122.2k)	Auto	487	65.3	-
	RobertaNet (2.2K)	Human	12	81.8	-
	RobertaNet (7.3K)	Human	12	68.6	-
	RobertaNet (23.6k)	Human	12	58.6	-
	RobertaNet (6.7K)	ConceptNet	20	88.0	64.4
	RobertaNet (24.3K)	ConceptNet	20	81.6	68.8
	RobertaNet (230K)	ConceptNet	20	55.0	87.0
KG Completion	COMET (6.7K)	ConceptNet	20	92.0	35.5
	COMET (230K)	ConceptNet	20	66.6	72.4
Text Mining	WebChild (4.6M)	-	20	82.0*	-
	ASCENT (8.6M)	-	-	79.2*	-
	TransOMCS (18.4M)	ConceptNet	20	56.0*	98.3

Table 2: Statistics of KGs constructed with different methods. Different paradigms of works **can not be directly compared** due to their different settings discussed in Table 1. We put the results together for reference purpose. *Novelty* refers to the proportion of entities that do not appear in ConceptNet, so only the methods with ConceptNet relations set have *Novelty* numbers. The accuracy with * are from the original papers and subject to different evaluation protocol. As a finetuned knowledge base completion model, COMET(Bosselut et al., 2019) can only predict the tail entity given a source entity and a relation, we generate KGs with COMET by feeding it the head entity produced by our ROBERTANET. The bottom block of the table summarizes the results from some major text mining methods described in Table 1, including WebChild (Tandon et al., 2014), ASCENT (Nguyen et al., 2021) and TransOMCS (Zhang et al., 2020a).

different LMs to understand the performance impact of diverse configurations, such as model sizes and pretraining strategies, as demonstrated in our experiments.

4 Experiments

To evaluate our framework, we extract knowledge of diverse new relations from various language models, and conduct human evaluation. We then make deeper analysis of prompt creation and scoring function in our framework. Finally, by utilizing our framework as a tool to interpret the knowledge stored in language models, we have made noteworthy observations regarding the knowledge capacity of black-box models.

4.1 Setup

Relations We evaluate our framework with several relation sets: (1) **ConceptNet** (Speer et al., 2017): Following Li et al. 2016, we filter the KG and use a set of 20 common relations (e.g. HAS_SUBEVENT, MOTIVATED_BY_GOAL). The initial prompts for these relations are from the ConceptNet repository, and we randomly sample 5 example entity pairs from the ConceptNet KG for each relation. (2) **LAMA** (Petroni et al., 2019): Following previous works, we use the T-REx split (41 relations from WikiPedia, such as capital_of, member_of). For each relation, the human-written prompt provided in Petroni et al. 2019 is used as the initial prompt and we randomly sample 5 example entity pairs for each relation. (3) **Human**: We write 12 new relations of interests that can hardly be found in any existing KGs, and manually write an initial prompt and 5 example entity pairs for them. The resulting relations include complex relations as described in Section 3.2. (4) **Auto**: Besides relations from existing KGs and human-written ones, we automatically derive a large set of relations from E-KAR (Chen et al., 2022), a dataset for analogical reasoning. In the original dataset, given an entity pair, e.g. ⟨ID_CARD, IDENTITY⟩, the task is to select an analogous tuple from multiple choices, e.g. ⟨PRACTICE LICENSE, QUALIFICATION⟩. To turn a sample in E-KAR into a relation, we use the tuple in the question and the correct choices as 2 example entity pairs, and extract the initial prompt from the explanation provided in E-KAR (e.g. *Proof of A requires B.*), resulting in 487 relations. Some of the relations are not straightforward, making this relation set more difficult than other ones. [3]

4.2 Extracting Knowledge of Diverse New Relations

Our framework is applied to extract knowledge graphs from LMs with relations of ConceptNet, Auto, and Human. The accuracy of the extracted knowledge is then evaluated with human annotation using Amazon Mechanical Turk (MTurk). Each extracted knowledge tuple is la-

[3] For reference, finetuned ROBERTA-LARGE achieves about 50% accuracy on the original dataset.

Methods	Acc	Rej
AUTOPROMPT	0.33	0.47
HUMAN PROMPT	0.60	0.27
TOP-1 PROMPT (Ours)	0.69	0.23
MULTI PROMPTS (Ours)	**0.73**	**0.20**

Table 3: The portions of accepted and rejected tuples in human evaluation across settings, with the ROBERTA-LARGE as the LM.

Source LMs	Acc	Rej
DISTILBERT	0.67	0.24
BERT-BASE	0.63	0.26
BERT-LARGE	0.70	0.22
ROBERTA-BASE	0.70	0.22
ROBERTA-LARGE	0.73	0.20

Table 4: The portions of accepted and rejected tuples in human evaluation across different LMs, using the MULTI-PROMPTS approach.

beled for correctness by three annotators using a True/False/Unjudgeable judge. A tuple is considered "accepted" if at least two annotators deem it to be true knowledge, and "rejected" if at least two annotators rate it as false. Here we refers portion of accepted tuples as accuracy.

The statistics of our resulting KGs are listed in Table 2. Besides, we also put the results of other paradigms of methods, including COMET for KG completion and text-mining based methods (Figure 1). Note that the results across different paradigms are generally not directly comparable due to vastly different settings. Yet we still collect the results together for reference purpose. From our RebertaNet with relation set "Auto", we are able to extract a reasonably large sets of knowledge (122K), by extracting knowledge with 487 easy-to-collect "Auto" relations. The set of relation is an order of magnitude larger than the predefined set of relations in both KG completion and text mining based on ConceptNet as shown in the table. The accuracy of 65% is at a comparable level with that of COMET (230K) and TransOMCS (18.4M), which is reasonable especially considering our method solely uses an LM as the source of knowledge without any external training data, bringing flexibility to dynamically incorporate new relations. Besides, for our RobertaNet on ConceptNet relations, although the numbers listed in the table are not simply comparable, we can still find that RobertaNet achieves similar accuracy and absolutely higher novelty comparing with the knowledge from COMET, which is already finetuned using large number of knowledge terms under the same set of ConceptNet relations. Further, our results on the "human" relation set demonstrate that our RobertaNet keeps working comfortably on our highly realistic relations of user interests, including the complex ones as described in section §3.2. We showcase knowledge samples harvested from DISTILLBERT in Figure 3.

4.3 Analyzing Automatic Prompt Creation

To evaluate the effect of our automatic creation of prompts, we compare the generated KGs under several settings on the Human relations: (1) **Multi-Prompts** refers to the the full framework described in §3 which use the automatically created diverse prompts in knowledge search. (2) **Top-1 Prompt**: To ablate the effect of ensembling multiple prompts, we evaluate the variant that uses only the prompt with largest weight (§3.1) for knowledge extraction. (3) **Human Prompt**: To further understand the effectiveness of the automatically created prompts, we assess the variant that uses the initial prompt of each relation. (4) **AutoPrompt** (Shin et al., 2020), which was proposed to learn prompts by optimizing the likelihood of tail entity prediction on the training set. To fit in our setting, we adapt it to optimize the compatibility score (Eq.1) on the example entity pairs. We omit other prompt tuning work (e.g., Zhong et al., 2021; Qin and Eisner, 2021) because they either are difficult to fit in our problem or require more training data and fail with only the few shot of example entity pairs in our setting.

We harvest 1000 tuples for each Human relation, and evaluate them with human annotation. The annotation results are presented in Table 3 (We also list the detailed results per relation in Table 5 for reference) Our TOP-1 PROMPT significantly improves the accuracy up to 9% over the HUMAN PROMPT, demonstrating the effectiveness of our prompt searching algorithm in generating high-quality prompts. MULTI-PROMPTS further improves the accuracy by an additional 4%, indicating that the combination of diverse prompts better captures the semantics of a relation. However, the method utilizing the optimized prompt by AUTOPROMPT results in lower accuracy than the use of human or searched prompts. This can be attributed to the insufficient number of example knowledge tuples used to learn effective prompts for the desired relations.

Based on the results above, we move a step for-

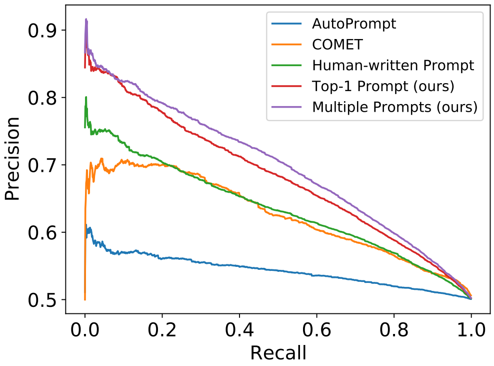


Figure 4: Precision-recall on ConceptNet relations.

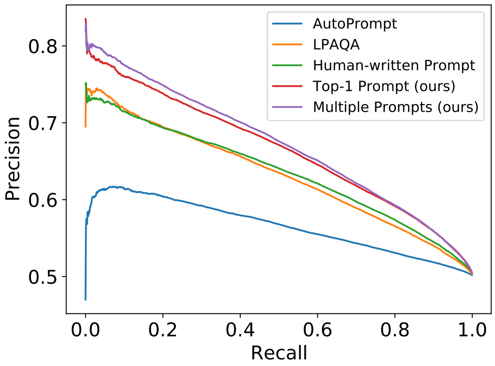


Figure 5: Precision-recall curve on LAMA relations.

ward to see how the created prompts influence the subsequent scoring module in the framework. Specifically, we study both the precision and recall of our scoring function parameterized by the prompts, to see if the automatically created prompts (§3.1) bring the consistency scoring (§3.2) better balance of knowledge accuracy (precision) and coverage (recall). To compare with other scoring methods that are restricted to specific sets of relations, this experiment was conducted using existing terms from both the ConceptNet and LAMA datasets.

Specifically, we use the knowledge tuples from ConceptNet and LAMA as positive samples (§4.1), and synthesize the same amount of negative samples with the same strategy in Li et al. (2016) by random replacing entities or relations in a true knowledge tuple. Each scoring function ranks the samples based on the scores from high to low. We can then compute both the *precision* and *recall* of positive samples at different cut-off points along the ranking, and plot the precision-recall curves for each method.

The automatic evaluation setting on given knowledge terms enables us to adapt existing prevalent works, e.g., KG completion and factual probing (Table 1), for comparison with our approach: **(1) COMET** (Bosselut et al., 2019) is a transformer-based KG completion model trained to predict the tail entity t conditioning on the head entity and relation (h, r) on ConceptNet. We use its log-likelihood $\log P(t|h, r)$ as the score for each given knowledge tuple. **(2) LPAQA** (Jiang et al., 2020) collects a set of prompts on LAMA with text mining and paraphrasing, and optimize their weights towards the objective of $\log P(t|h, r)$ on training samples.

The resulting precision-recall curves on ConceptNet and LAMA knowledge are shown in Figure 4 and Figure 5, respectively. Scoring with multiple prompts always achieves best performance, followed by Top-1 prompts and then Human-written prompts. The finding is consistent with previous experiments, which verified the effectiveness of our scoring function design. Our framework also outperforms other baselines, such as COMET on ConceptNet and LPAQA on LAMA. Though trained with labeled data, these methods are only optimized to completing a tail entity given a query, in stead of scoring an entity pair, which is essential to extract KGs from LMs.

4.4 Analysis of Knowledge in Different LMs

As previously mentioned in Section §3, the resulting knowledge graphs can be viewed as a symbolic interpretation of LMs. We extract knowledge graphs from 5 distinct language models and submit them to human annotation evaluation. The findings are presented in Table 4 (The detailed results per relation is listed in Table 5), which sheds some new light on several knowledge-related questions regarding the LMs' knowledge capacity.

Does a larger LM encode better knowledge? The large version of BERT and RoBERTa have the same pretraining corpus and tasks as their base versions, but have larger model architecture in terms of layers (24 v.s. 12), attention heads (16 v.s. 12), and the number of parameters (340M v.s. 110M). We can see that the accuracies of BertNet-large and RoBERTaNet-large are around 7% and 3% higher than their base version, separately, indicating the larger models indeed encoded better knowledge than the base models.

Does better pretraining bring better knowledge? RoBERTa uses the same architecture as BERT but with better pretraining strategies, like dynamic masking, larger batch size, etc. In their corresponding KGs from our framework, RoBERTaNet-large performs better than BertNet-large (0.73 v.s. 0.70), and RoBERTaNet-base is also better than BertNet-base (0.70 v.s. 0.63), showing that the better pretraining in RoBERTa leads to better knowledge learning and storage.

Is knowledge really kept in the knowledge distillation process? DistilBERT is trained by distilling BERT-base, and it reduces 40% parameters from the latter. Interestingly, the knowledge distillation process instead improves around 4% of accuracy in the result knowledge graph. This should be attributed to the knowledge distillation process which might eliminate some noisy information from the teacher model.

5 Conclusion

We have developed an automatic framework that extracts a KG from a pretrained LM (e.g, BERT, ROBERTA), in an efficient and scalable way, resulting in a family of new KGs, which we refer to as BERTNET, ROBERTANET, etc. Our framework is capable of extracting knowledge of arbitrary new relation types and entities, without being restricted by pre-existing knowledge or corpora. The resulting KGs also serve as interpretation of source LMs.

Limitations Our current design and experimental studies are limited on LMs in the generic domain, and are not yet been studied in specific domains such as extracting healthcare knowledge from relevant neural models. We leave the exciting work of harvesting knowledge from various kinds of neural networks across applications and domains in the future work.

Ethical considerations In this work, the harvested knowledge is automatically generated by LMs. We would like to note that the language models could possibly generate unethical knowledge tuples, same with the risks of other applications using language models for generation. We hope that the knowledge extraction study could offer techniques to better interpret and understand the language models, and in turn foster the future research of language model ethics. Since the knowledge graph only consists simple phrases, we think filtering sensitive words would be effective. No foreseeable negative societal impacts are caused by the method itself.

References

Badr AlKhamissi, Millicent Li, Asli Celikyilmaz, Mona Diab, and Marjan Ghazvininejad. 2022. A review on language models as knowledge bases. *arXiv preprint arXiv:2204.06031*.

Greg Anderson, Abhinav Verma, Isil Dillig, and Swarat Chaudhuri. 2020. Neurosymbolic reinforcement learning with formally verified exploration. *Advances in neural information processing systems*, 33:6172–6183.

Gabor Angeli, Melvin Johnson, and Christopher D. Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *ACL*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for knowledge graph construction. *The Association for Computational Linguistics*.

Zied Bouraoui, José Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from bert. In *AAAI*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models.

Jiangjie Chen, Rui Xu, Ziquan Fu, Wei Shi, Zhongqiao Li, Xinbo Zhang, Changzhi Sun, Lei Li, Yanghua Xiao, and Hao Zhou. 2022. E-kar: A benchmark for rationalizing natural language analogical reasoning. *arXiv preprint arXiv:2203.08480*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Tianqing Fang, Hongming Zhang, Weiqi Wang, Yangqiu Song, and Bin He. 2021. Discos: Bridging the gap between discourse knowledge and commonsense knowledge. In *Proceedings of the Web Conference 2021*, pages 2648–2659.

Joshua Feldman, Joe Davison, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *EMNLP*.

Christiane D. Fellbaum. 2000. Wordnet : an electronic lexical database. *Language*, 76:706.

Google. 2012. Introducing the knowledge graph: things, not strings.

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model.

Peter Hase, Mona T. Diab, Asli Çelikyilmaz, Xian Li, Zornitsa Kozareva, Veselin Stoyanov, Mohit Bansal, and Srini Iyer. 2021. Do language models have beliefs? methods for detecting, updating, and visualizing model beliefs. *ArXiv*, abs/2111.13654.

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H Hovy, and Eric P Xing. 2016. Harnessing deep neural networks with logic rules. In *ACL (1)*.

Zhiting Hu and Eric P. Xing. 2022. Toward a 'Standard Model' of Machine Learning. *Harvard Data Science Review*, 4(4). Https://hdsr.mitpress.mit.edu/pub/zkib7xth.

Zhengbao Jiang, Frank F. Xu, J. Araki, and Graham Neubig. 2020. How can we know what language models know? *TACL*.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. *arXiv preprint arXiv:2205.11822*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*.

Christy Y. Li, Xiaodan Liang, Zhiting Hu, and Eric P. Xing. 2019. Knowledge-driven encode, retrieve, paraphrase for medical image report generation. In *AAAI*.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1445–1455, Berlin, Germany. Association for Computational Linguistics.

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.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*.

Benjamin Newman, Prafulla Kumar Choubey, and Nazneen Rajani. 2021. P-adapters: Robustly extracting factual information from language models with diverse prompts. *ArXiv*, abs/2110.07280.

Tuan-Phong Nguyen, Simon Razniewski, Julien Romero, and Gerhard Weikum. 2021. Refined commonsense knowledge from large-scale web contents. *arXiv preprint arXiv:2112.04596*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *EMNLP*.

Guanghui Qin and Jas' Eisner. 2021. Learning how to ask: Querying lms with mixtures of soft prompts. In *NAACL*.

Julien Romero, Simon Razniewski, Koninika Pal, Jeff Z. Pan, Archit Sakhadeo, and Gerhard Weikum. 2019. Commonsense properties from query logs and question answering forums. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1411–1420.

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

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. 2019. Atomic: An atlas of machine commonsense for if-then reasoning. *ArXiv*, abs/1811.00146.

Shaoyun Shi, Hanxiong Chen, Min Zhang, and Yongfeng Zhang. 2019. Neural logic networks. *ArXiv*, abs/1910.08629.

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Eliciting knowledge from language models using automatically generated prompts. *EMNLP*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Stephen P Stich. 1979. Do animals have beliefs? *Australasian Journal of Philosophy*, 57(1):15–28.

Bowen Tan, Lianhui Qin, Eric Xing, and Zhiting Hu. 2020. Summarizing text on any aspects: A knowledge-informed weakly-supervised approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6301–6309.

Niket Tandon, Gerard De Melo, Fabian Suchanek, and Gerhard Weikum. 2014. Webchild: Harvesting and organizing commonsense knowledge from the web. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 523–532.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, and Yi Chang. 2021a. Structure-augmented text representation learning for efficient knowledge graph completion. *Proceedings of the Web Conference 2021*.

Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Multi-task feature learning for knowledge graph enhanced recommendation. *The World Wide Web Conference*.

Liming Wang, Siyuan Feng, Mark Hasegawa-Johnson, and Chang Yoo. 2022. Self-supervised semantic-driven phoneme discovery for zero-resource speech recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8027–8047, Dublin, Ireland. Association for Computational Linguistics.

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. 2021b. Covid-19 literature knowledge graph construction and drug repurposing report generation. In *NAACL*.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2022. Symbolic knowledge distillation: from general language models to commonsense models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4602–4625, Seattle, United States. Association for Computational Linguistics.

Peter West, Chandra Bhagavatula, Jack Hessel, Jena D Hwang, Liwei Jiang, Ronan Le Bras, Ximing Lu, Sean Welleck, and Yejin Choi. 2021. Symbolic knowledge distillation: from general language models to commonsense models. *arXiv preprint arXiv:2110.07178*.

Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. *Proceedings of the 26th International Conference on World Wide Web*.

Yunrong Yang, Zhidong Cao, Pengfei Zhao, Dajun Daniel Zeng, Qingpeng Zhang, and Yin Luo. 2021. Constructing public health evidence knowledge graph for decision-making support from COVID-19 literature of modelling study. *Journal of Safety Science and Resilience*, 2(3):146–156.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *ArXiv*, abs/1909.03193.

Hongming Zhang, Daniel Khashabi, Yangqiu Song, and Dan Roth. 2020a. Transomcs: From linguistic graphs to commonsense knowledge. In *IJCAI*.

Hongming Zhang, Xin Liu, Haojie Pan, Yangqiu Song, and Cane Wing-Ki Leung. 2020b. Aser: A large-scale eventuality knowledge graph. In *Proceedings of the web conference 2020*, pages 201–211.

Zexuan Zhong and Danqi Chen. 2020. A frustratingly easy approach for entity and relation extraction. *arXiv preprint arXiv:2010.12812*.

Zexuan Zhong, Dan Friedman, and Danqi Chen. 2021. Factual probing is [mask]: Learning vs. learning to recall. *NAACL*.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix X. Yu, and Sanjiv Kumar. 2020. Modifying memories in transformer models. *ArXiv*, abs/2012.00363.

A Detailed Results of Harvested Knowledge

In Table 3 and Table 4, we show the human-annotated results of harvested knowledge in different settings. Here we list the detailed results per relation in Table 5.

B Preprocessing of ConceptNet

We filter out some linguistic relations (e.g. `etymologically derived from`) and some trivial relations (e.g. `related to`). We only consider the tuples with confidence higher than 1, and filter out relations comprising less than 1000 eligible tuples. We don't directly take the test set from (Li et al., 2016) because they reserve a lot of tuples for training, resulting in a small and unbalanced test set.

C Efficient knowledge tuple search

In the candidate entity pairs proposal step, we use the minimum token log-likelihoods (shorted as MTL) instead of the full Equation 2, which allows us to apply a pruning strategy. The pseudo-code is shown in Algorithm 1. For simplicity of the pseudo-code, we only include the case where each entity is composed of a single token. Appendix **??** illustrates the processing of multi-token entities. It's worth noting that our algorithm is an exact search algorithm instead of approximated algorithms like beam search, which prevents the results from biasing towards more probable head entities.

As a running example, when we are searching for 100 entity tuples, we maintain a minimum heap to keep track of the MTL of the entity tuples. The maximum size of this heap is 100, and the heap top can be used as a threshold for future search because it's the 100-th largest MTL: When we are searching for a new entity tuple, once we find the log-likelihood at any time step is lower than the threshold, we can prune the continuous searching immediately, because this means the MTL of this tuple will never surpass any existing tuples in the heap. If a new entity tuple is searched out without being pruned, we will pop the heap and push the MTL of the new tuple. Intuitively, the pruning process makes sure that the generated part of the tuple in searching is reasonable for the given prompt.

Algorithm 1 Efficient Entity Tuple Search

```
Input: LM: A language model; $n_r$: The entity number for
  a tuple of relation $r$; $N$: maximum number of candidate
  tuples; $P_r$: The set of prompts describing relation $r$
Output: tuple_list: A list of $N$ entity tuples
  heap ← MinHeap()
  function DFS(cur_tuple, cur_MTL)
      idx←Count(cur_tuple)
      if idx = $n_r$ then
          heap.push((cur_tuple, cur_MTL))
          if len(heap) > N then
              heap.pop()
          end if
      end if
      for v ∈ Vocab(LM) do
          cur_L ← $\log p_{LM}(v|cur\_tuple, P_r)$
          cur_MTL = min(cur_L, cur_MTL)
          if Count(cur_tuple > 0) and cur_MTL < heap.top() then return    ▷ Pruning
          end if
          cur_tuple.append(v)
          DFS(cur_tuple, cur_MTL)
      end for
  end function
  DFS(EmptyList(), 0)
  tuple_list ← list(heap)
```

D Detailed Experiment setting

We use GPT-3 with the instruction "paraphrase:sentence" with a few examples as the off-the-shelf paraphraser. In entity pair searching, we restrict every entity to appear no more than 10 times to improve the diversity of generated knowledge and search out at most 50,000 entity tuples for each relation. We finally use various score thresholds to get the outcome KGs in different scales, including (1) 50%: taking half of all searched-out entity pairs with higher consistency for each relation (2) base-k: Naturally, there are different numbers of valid tuples for different relations (e.g. tuples of $\langle$ $\dots$, CAPITAL_OF, $\dots\rangle$ should not exceed 200 as that is the number of all the countries in the world). We design a relation-specific thresholding method, that is to set 10% of the k-th consistency as the threshold (i.e., 0.1 × consistency$_k$), and retain all tuples with consistency above the threshold. We name the settings base-10 and base-100 when k is 10 and 100, respectively. We list the truncation method applied to each variant of ROBERTANET listed in Table 2:

- RobertaNet (122.2k) - Auto: base-10
- RobertaNet (6.7K) - ConceptNet: base-10
- RobertaNet (24.3K) ConceptNet: base-100
- RobertaNet (230K) ConceptNet: 50%

Table 5: Detailed result of human evaluation. The numbers indicate the portions of accepted and rejected tuples. Ro-l, DB, B-b, B-l, Ro-b are short for Roberta-large, DistilBert, Bert-large, Bert-base, Roberta-base. Human, Auto, Top-1, and Multi stand for methods that use Human Prompt, Autoprompt, Top-1 Prompt (Ours), and Multi Prompts (Ours).

Model	Ro-l	Ro-l	Ro-l	Ro-l	DB	B-b	B-l	Ro-b
Prompt	Human	Auto	Top-1	Multi	Multi	Multi	Multi	Multi
business	0.60/0.32	0.76/0.13	0.75/0.16	0.88/0.07	0.54/0.27	0.64/0.23	0.76/0.13	0.74/0.19
help	0.77/0.12	0.52/0.34	0.92/0.03	0.87/0.05	0.91/0.04	0.81/0.04	0.88/0.06	0.88/0.06
ingredient for	0.59/0.33	0.33/0.59	0.73/0.20	0.71/0.24	0.70/0.26	0.55/0.40	0.72/0.23	0.51/0.40
place for	0.76/0.10	0.41/0.36	0.63/0.32	0.89/0.07	0.84/0.14	0.78/0.18	0.87/0.11	0.88/0.09
prevent	0.42/0.42	0.18/0.67	0.60/0.25	0.40/0.45	0.60/0.32	0.44/0.39	0.62/0.25	0.68/0.25
source of	0.76/0.17	0.21/0.67	0.52/0.44	0.60/0.33	0.63/0.36	0.65/0.32	0.75/0.24	0.55/0.37
separated by the ocean	0.48/0.38	0.16/0.48	0.56/0.35	0.55/0.40	0.51/0.24	0.57/0.26	0.44/0.46	0.44/0.49
antonym	0.50/0.41	0.10/0.83	0.50/0.48	0.55/0.44	0.38/0.56	0.41/0.56	0.52/0.42	0.75/0.22
featured thing	0.85/0.12	0.38/0.40	0.88/0.06	0.89/0.10	0.37/0.44	0.44/0.40	0.46/0.44	0.65/0.20
need A to do B	0.71/0.18	0.62/0.21	0.66/0.22	0.79/0.10	0.83/0.12	0.62/0.25	0.65/0.18	0.72/0.17
can but not good at	0.52/0.34	0.29/0.42	0.61/0.19	0.44/0.21	0.51/0.31	0.60/0.21	0.64/0.22	0.39/0.35
worth celebrating	0.47/0.29	0.23/0.51	0.81/0.05	0.85/0.08	0.79/0.12	0.74/0.14	0.84/0.10	0.83/0.10
potential risk	0.40/0.23	0.31/0.45	0.70/0.21	0.76/0.19	0.87/0.05	0.66/0.22	0.72/0.16	0.79/0.08
A do B at	0.56/0.33	0.14/0.55	0.79/0.14	0.97/0.03	0.93/0.07	0.93/0.05	0.94/0.06	0.94/0.06
Average	0.60/0.27	0.33/0.47	0.69/0.22	0.73/0.20	0.67/0.24	0.63/0.26	0.70/0.22	0.70/0.22

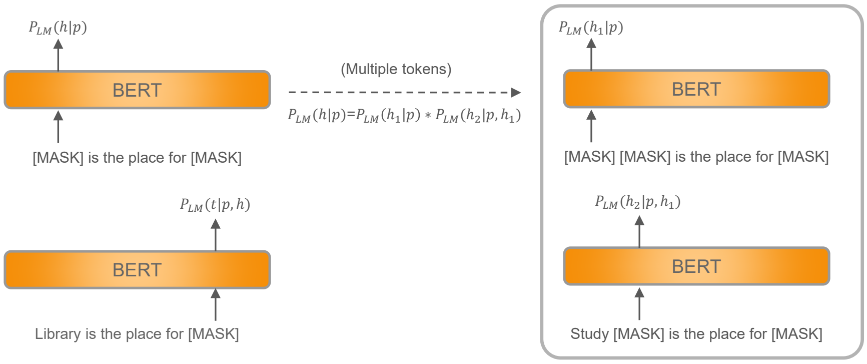

$P_{LM}(h|p)$
BERT
[MASK] is the place for [MASK]
(Multiple tokens)
$P_{LM}(h|p)$=$P_{LM}(h_1|p) * P_{LM}(h_2|p,h_1)$
$P_{LM}(h_1|p)$
BERT
[MASK] [MASK] is the place for [MASK]
$P_{LM}(t|p,h)$
BERT
Library is the place for [MASK]
$P_{LM}(h_2|p,h_1)$
BERT
Study [MASK] is the place for [MASK]

Figure 6: We demonstrate the calculation with an example where p ="A is the place for B". The left two figures shows how we calculate $P_{LM}(h|p)$ and $P_{LM}(t|p,h)$. In this example, h ="library" when we set both head and tail entities to have one single token. The right block shows how we calculate the conditional probability of multiple-token entities by decomposing it into two steps. In this example, the first token of the head entity h_1 ="study".

- RobertaNet (2.2K) Human: base-10
- RobertaNet (7.3K) Human: base-100
- RobertaNet (23.6k) Human: 50%

E Human evaluation

We present the screenshot of the instruction in Figure 7 and question in Figure 8. The inter-annotator agreement (Krippendorff's Alpha) is 0.27, showing fair agreement.

F Compute resource

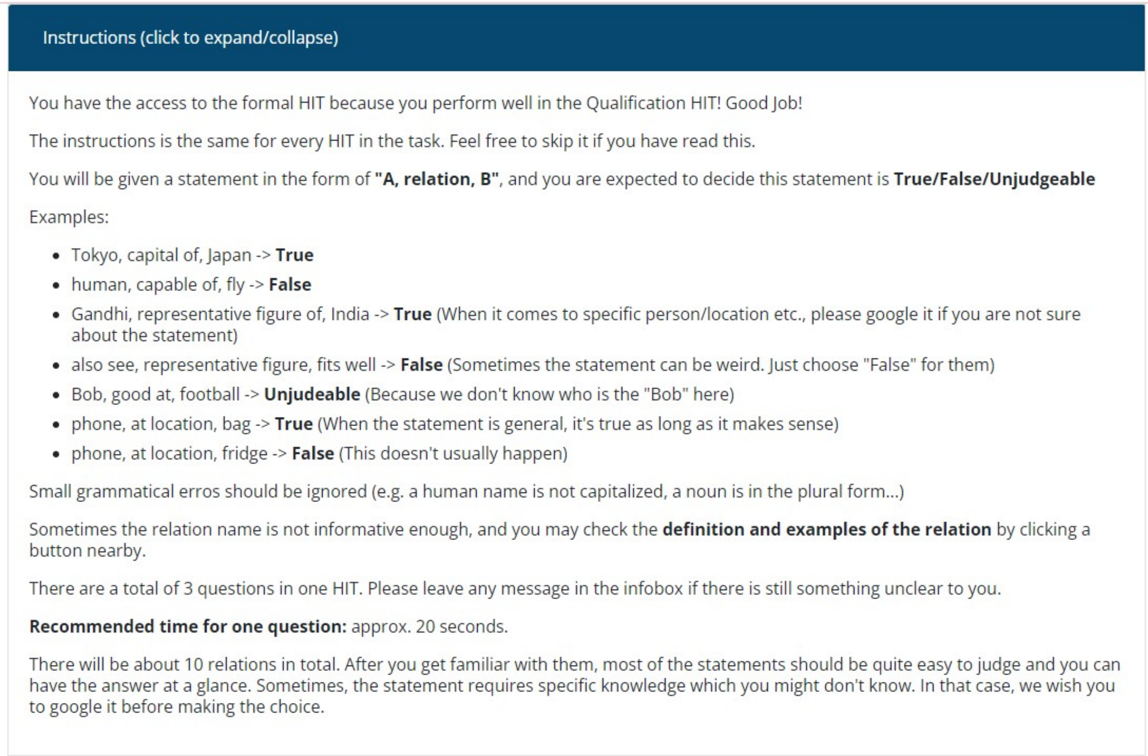
All of our experiments are running on a single Nvidia GTX1080Ti GPU. Harvesting a knowledge graph of one relation with Roberta-large takes about one hour.

G The license of the assets

All the data we used in this paper, including datasets, relation definitions, seed entity pairs, etc., are officially public resources.

H Potential Risks

We identify that our system is minimal in risks. Our proposed system produce results only based on the source language models like BERT. The risks of language models are well studied and our methods do not perpetuate or add to the known risks. However, we acknowledge the methods could be applied to maliciously trained language models and discourage such uses.



Instructions (click to expand/collapse)

You have the access to the formal HIT because you perform well in the Qualification HIT! Good Job!

The instructions is the same for every HIT in the task. Feel free to skip it if you have read this.

You will be given a statement in the form of **"A, relation, B"**, and you are expected to decide this statement is **True/False/Unjudgeable**

Examples:

- Tokyo, capital of, Japan -> **True**
- human, capable of, fly -> **False**
- Gandhi, representative figure of, India -> **True** (When it comes to specific person/location etc., please google it if you are not sure about the statement)
- also see, representative figure, fits well -> **False** (Sometimes the statement can be weird. Just choose "False" for them)
- Bob, good at, football -> **Unjudeable** (Because we don't know who is the "Bob" here)
- phone, at location, bag -> **True** (When the statement is general, it's true as long as it makes sense)
- phone, at location, fridge -> **False** (This doesn't usually happen)

Small grammatical erros should be ignored (e.g. a human name is not capitalized, a noun is in the plural form...)

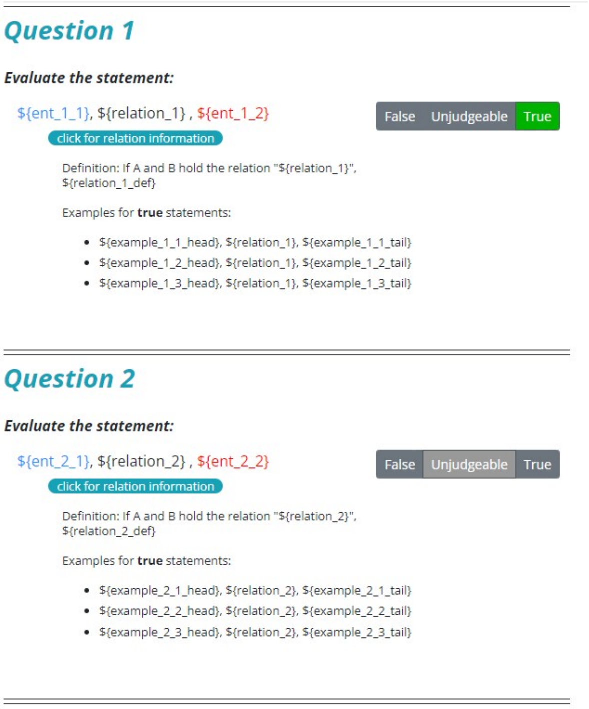
Sometimes the relation name is not informative enough, and you may check the **definition and examples of the relation** by clicking a button nearby.

There are a total of 3 questions in one HIT. Please leave any message in the infobox if there is still something unclear to you.

Recommended time for one question: approx. 20 seconds.

There will be about 10 relations in total. After you get familiar with them, most of the statements should be quite easy to judge and you can have the answer at a glance. Sometimes, the statement requires specific knowledge which you might don't know. In that case, we wish you to google it before making the choice.

Figure 7: The instruction to annotators



Question 1

Evaluate the statement:

${ent_1_1}, ${relation_1} , ${ent_1_2}

False
Unjudgeable
True
click for relation information

Definition: If A and B hold the relation "${relation_1}", ${relation_1_def}

Examples for **true** statements:

- ${example_1_1_head}, ${relation_1}, ${example_1_1_tail}
- ${example_1_2_head}, ${relation_1}, ${example_1_2_tail}
- ${example_1_3_head}, ${relation_1}, ${example_1_3_tail}

Question 2

Evaluate the statement:

${ent_2_1}, ${relation_2} , ${ent_2_2}

False
Unjudgeable
True
click for relation information

Definition: If A and B hold the relation "${relation_2}", ${relation_2_def}

Examples for **true** statements:

- ${example_2_1_head}, ${relation_2}, ${example_2_1_tail}
- ${example_2_2_head}, ${relation_2}, ${example_2_2_tail}
- ${example_2_3_head}, ${relation_2}, ${example_2_3_tail}

Figure 8: The questions to annotators

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations section

☑ A2. Did you discuss any potential risks of your work?
Appendix F

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Yes. Abstract and section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4

☑ B1. Did you cite the creators of artifacts you used?
Section 4

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
4.1

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
4.1

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
4.1

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
4.1

C ☑ Did you run computational experiments?

4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
4

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?
4.2.2

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
appendix E

☑ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
appendix E

☑ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
appendix E

☑ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
4.2

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.