Tuning LLMs with Explicit Knowledge

Anonymous ACL submission

Abstract

Explicit knowledge is essential for reasoning and understanding, offering interpretability and ease of transfer and editing. In contrast, traditional parameter fine-tuning in neural networks lacks transparency and flexibility.

In this paper, we propose a novel framework for fine-tuning language models using explicit knowledge. Unlike traditional fine-tuning that directly modifies model parameters, our framework enhances the model's capabilities by externalizing its implicit knowledge, allowing for corrections and adjustments on an explicit knowledge base. This knowledge base can be independently validated, modified, and easily transferred.

Our experiments across multiple datasets have shown broad applicability and significant improvements, such as achieving a 33.7% increase in performance on the CLUTRR dataset to 5-shot CoT and comparable or even better performances in contrast to advanced LLMs with parameter fine-tuning.

In conclusion, our approach shifts from implicit to knowledge-level learning in contrast to parameter fine-tuning, which can also be viewed as an additional fine-tuning step compared to knowledge acquisition. Furthermore, we provide the relevant theoretical analysis and proofs.

1 Introduction

In recent years, the efficacy of LLMs have demonstrated remarkable performance across a wide range of natural language understanding (Bubeck et al., 2023). These models have demonstrated unparalleled capabilities in generating information and knowledge through human language, marking a significant advancement in the field of artificial intelligence.

Despite their remarkable capabilities, large language models frequently encounter difficulties when addressing domain-specific or highly specialized queries (Kandpal et al., 2023). These challenges primarily arise from the implicit learning, memorization, and reasoning processes intrinsic to neural networks. Such mechanisms lead to issues such as hallucination of information, problems with outdated data, and so on (Gao et al., 2023b; Rawte et al., 2023; Wang et al., 2023).

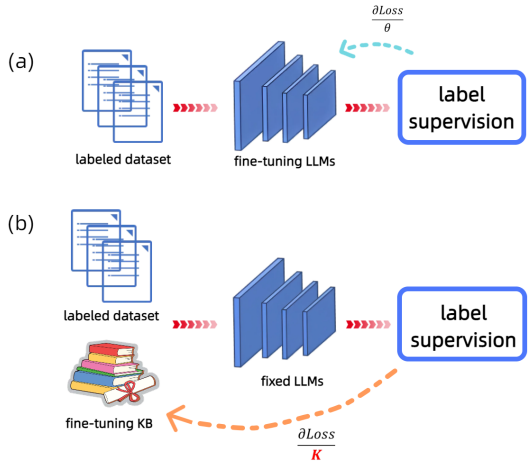

Figure 1: The main paradigms of (a) supervised fine-tuning [1] and (b) our framework for tuning LLMs with explicit knowledge. The main difference is that parameter fine-tuning encodes knowledge with implicit parameters, while our framework fine-tuning LLMs with the knowledge in natural language form.

To address this, we propose the a framework named **knowledge fine-tuning** (K-FT), which shifts the focus from tuning model parameters to refining the internal knowledge within the model (see Figure 1). Knowledge fine-tuning is based on insights from prior research, which shows a strong correlation between the correctness of rationales and the accuracy of predictions (Zhang et al., 2023; Zhu et al., 2023; Ye and Durrett, 2022; Wang

[1] Figure 1.(a) is from doi.org/10.1080/17538947.2024.2328827

et al., 2022). This allows us to calculate the impact of each rationale on the final prediction loss, and then apply a method similar to back-propagation (Pryzant et al., 2023; Gao et al., 2023a), enabling the model to generate and learn the confidence associated with its acquired knowledge.

> Q1: Why opt for **knowledge** over parameters?

Humans always abstract knowledge from observations to navigate various scenarios, it is more generalizable across different domains and adaptable to novel problems (Davis et al., 1993; Wang et al., 2024a). In contrast, the parameters of a neural network are specific to the task it was trained on (Wang et al., 2024b).

We can acquire knowledge from limited examples and then infer general cases. In contrast, neural networks often require large amounts of labeled data to achieve a comparable level of performance. Knowledge is efficient due to the ability to use abstraction, while neural networks rely on massive datasets to "learn" patterns (Lai et al., 2021). □

> Q2: Why prioritize **explicit** over implicit representation? And why build an external knowledge base instead of using knowledge inside LLMs?

Explicit knowledge representation can provide more interpretable and explainable inference processes, easier to transfer and edit if necessary (Chen et al., 2022), which is also a condition for access to higher-order thoughts (Dienes and Perner, 1999). On the other hand, By storing knowledge in an explicit way and in an external knowledge base, models are less prone to the hallucination of facts and better positioned for transparent inference processes (Yang et al., 2024). □

> Q3: Why focus on **tuning** rather than knowledge extraction?

Tuning allows LLMs to verify the correctness of gained knowledge in more cases, which makes our knowledge fine-tuning framework perform better than simple extraction or acquisition. The theoretic analysis and proofs in section 3.5 will show further understanding and insights. □

In our approach, we require the model to explicitly express its knowledge and apply it during inference to generate answers, representing the forward propagation phase. These generated answers are then utilized to calculate scores for each knowledge involved, marking the backward propagation phase.

While this approach is effective for building a knowledge base, we found that the plain forward-backward process often results in misleading knowledge and has difficulty overriding the model's inherent semantic priors.

To address this, we introduced an additional memorization phase to the training cycle, creating a comprehensive **forward-backward-memorize** loop. This iterative process enables the model to retain and reinforce correct and broadly applicable knowledge with high confidence over multiple epochs.

Our method is inspired by a theory of cognitive science, as noted in (Mitchell, 2021; Hofstadter and Sander, 2013). In human cognition, concepts evolve through analogies formed over the years, and our reasoning relies on categorizing and applying these concepts. Following this idea, we categorize the acquired knowledge. In this study, a simple TF-IDF approach is used to extract key nouns from problem-solving rationales as the corresponding knowledge's concepts (or can say categories), and then associate the concepts with the knowledge. For future work, it is easy to explore advanced keyword extraction techniques or topic modeling algorithms to improve this process further.

To validate our claims, we apply our proposed framework to three publicly available datasets, CLUTRR, Lang-8, and SST-2. Our experiments showcase the framework's capability to construct high-quality knowledge bases and leverage them for model enhancement only with generic 0-shot prompts.

Furthermore, we provide theoretical evidence demonstrating the advantages of iterative knowledge fine-tuning over traditional knowledge acquisition methods in section 3.5. This suggests that similar algorithms could also benefit from this approach. Finally, we explore various aspects of knowledge fine-tuning, including its universality, applicability to smaller models, and broader implications.

In summary, our contributions are threefold:

1. We propose a universal discrete optimization framework for tuning LLMs with ex-

plicit knowledge, rather than parameters. This approach not only enhances model performance from the training phase but also induces the creation of an easily comprehensible and transferable knowledge base.

2. We give theoretic proof of the advantages of our framework and explore the significance and learnability of implicit knowledge for large language models across models of different sizes.

3. We provide preliminary framework codes to expedite related research endeavors.

2 Related work

Discrete and explicit optimization: Prompting techniques can be categorized into “soft" and “hard" prompts, with the former resembling an indistinct, non-transferable pre-module, while hard prompts use clear, reusable tokens to guide the model's outputs (Gimpel et al., 2023).

Previous works such as AutoPrompt (Shin et al., 2020) and GrIPS (Prasad et al., 2023) explore automatic trigger token search and iterative prompt editing, respectively, aiming to optimize LLM outputs for specific tasks. Meanwhile, BBTv2 introduces novel frameworks by using gradient-free optimization to reduce complexity and enhance model performance (Sun et al., 2022). With the development of analysis and chat capabilities of LLMs, researchers try to optimize prompts by the language model itself. Pryzant proposes a non-parametric optimization with “gradient descent" which further combines the traditional fine-tuning process with discrete “gradient" (Pryzant et al., 2023).

However, these approaches primarily concentrate on external prompt adjustments, often overlooking the potential of the model's internal knowledge.

Basic capabilities of large models and In-context learning: Our method shifts focus towards the LLM's inherent knowledge, aiming to not only draw out but also refine and re-inject this knowledge within the model. Compared to prompt tokens and parameters, knowledge bases provide advantages in interpretability, transferability (Zhu et al., 2023), and editability.

Moreover, advanced language models' exceptional capabilities enable us to express knowledge and rules in various forms, including natural sentences (Zhu et al., 2023), tables (Chen, 2023), logical expressions (Rajasekharan et al., 2023), among others, which sets it apart from traditional approaches which represent knowledge or rules limited forms (Tena Cucala et al., 2022; Rocktäschel and Riedel, 2017; Minervini et al., 2020).

Knowledge fine-tuning framework relies on the advanced natural language understanding and generation capability. LLMs have shown the ability to comprehend context (Zhu et al., 2023), utilize contextual information (Dai et al., 2023), and even override semantic priors (Wei et al., 2023). They can benefit from retrieval processes, enabling them to quickly adapt to new domains (Gao et al., 2023b).

Additionally, some studies attempt to explain the principles of ICL. For instance, Dai et al. have equated the formulation of example problems with gradient updates, viewing input examples as meta-gradients (Dai et al., 2023). Similarly, experiments on regression problems by Oswald and others support the meta-gradient perspective (Von Oswald et al., 2023), which also underlies the theoretical rationale for re-injecting knowledge bases after their induction and learning in this work.

3 Method

3.1 Overview

The framework we proposed is delineated into two to three distinct phases, rather than fine-tuning with parameters but focusing on fine-tuning with explicit knowledge. After generating and utilizing explicit knowledge during the forward propagation phase, the knowledge that has been memorized can be updated based on the loss during the backward propagation phase. The three phases are as follows (see Figure 2 and Figure 1 in supplementary material):

Forward Propagation The forward phase should answer the questions about how to generate the knowledge from an implicit model, and how to utilize it in the inference process.

Backward Propagation The backward phase involves adding new knowledge and adjusting the confidence of existing knowledge based on the rationale in the forward phase.

Memorization The final phase demands that the model integrates the learned knowledge base into its existing knowledge. Strategies may include using prompts, Retrieval-Augmented Generation (RAG), and others. In this paper, we propose a memorization-and-retrieval (MAR) strategy,

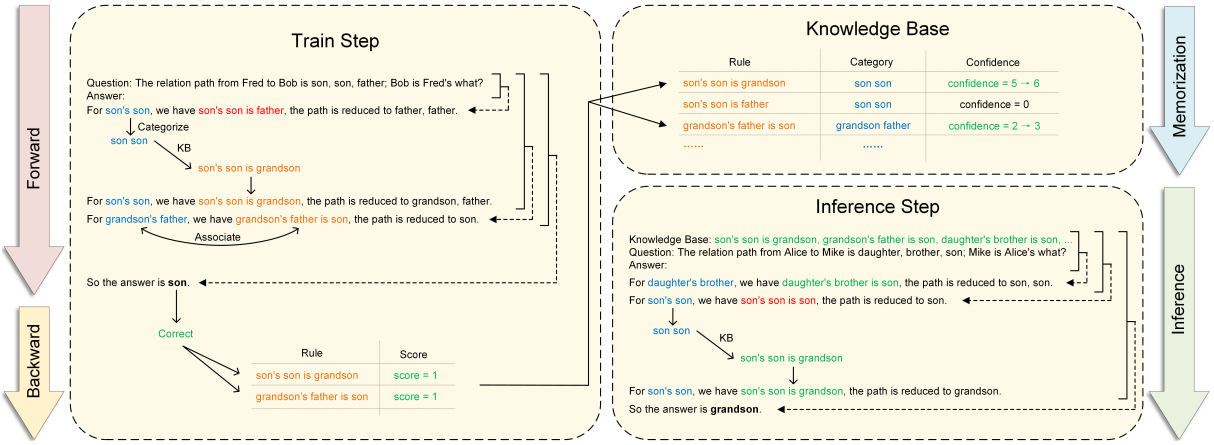

Figure 2: Our approach has three components: forward, backward, and memorization. The forward component includes two steps: externalization and inference. In externalization, we extract knowledge from the LLM through strategic questioning to build a foundational knowledge base. In inference, we inject the knowledge induced and further use the MAR strategy to enhance it. The backward component involves evaluation and refinement of the knowledge base. The memorization component categorizes the contexts (refer to reasoning scenario) of knowledge. These stages are repeated to ensure a comprehensive and accurate knowledge base.

in which some index keys (concepts, embeddings, etc.) are assigned to each knowledge. The keys represent the context/case in which the knowledge occurs and are then used to retrieve appropriate knowledge during the knowledge inference phase.

Knowledge Base The basic version of the knowledge base mainly consists of two ingredients: knowledge and confidence. When a piece of knowledge is utilized and yields accurate outcomes, it is more likely to be regarded as general, correct knowledge. Conversely, if it results in erroneous answers or is not selected by a language model, its credibility is diminished.

Meanwhile, our method supports any natural language, offering enhanced usability, understandability, and generalizability compared to other techniques.

$$\mathcal{K} = \{(k_i, c_i) | i = 1, 2 ... n\}$$

Where k_i is the knowledge text and c_i means the confidence of it.

3.2 Forward propagation

3.2.1 How to Make Knowledge Explicit

In our framework, knowledge explicitness is achieved through a prompt with a certain format to involve knowledge text. The model is instructed to think step by step and extract the knowledge it employs, such as "think step by step and output knowledge by '⟨begin⟩⟨end⟩' format", the prompt used in "Hypotheses-to-Theories" (HtT) (Zhu et al., 2023), and so on.

3.2.2 Knowledge Inference

The inference stage emphasizes reasoning based on the injected knowledge rather than adjusting the model's parameters. Concretely, We construct the training prompt by combining (see section A in the Appendix) an instruction or few-shot examples with some knowledge from the knowledge base. This prompt is then provided to the large language model (LLM) to generate answers and rationales. The generated rationales are subsequently used to extract knowledge and answers, which, along with the labels, are employed in the evaluation process to compute loss.

Formally, we can express the forward phase by:

$$x_t = \arg\max_{v \in V} P(v | I, F, \mathcal{K}, x_{<t}) \quad (1)$$

where I denotes the instruction, F represents the few-shot prompt, V is the vocabulary and $v \in V$ is the candidate token.

3.3 Backward propagation

In the backward stage, knowledge utilized from the text is extracted, and the loss is calculated based on the provided answers. Subsequently, the confidence in the extracted knowledge is updated accordingly.

The process can be formally expressed as follows:

1. Knowledge Extraction:

$$k = f_{\text{extract}}(x)$$

where k represents the knowledge extracted from the text x, and f_{extract} denotes the extraction function.

2. Loss Calculation:

$$L(\mathcal{K}) = \mathcal{L}(L, P)$$

Here, $L(\mathcal{K})$ is the loss function, L_i refers to the labels, P_i is the final predictions, and $\mathcal{L}$ is the loss function measuring the discrepancy between the predicted answer and the label, just a distant supervision signal.

3. Parameter and Confidence Update:

$$\mathcal{K}(k) \leftarrow \mathcal{K}(k) - \nabla_{\mathcal{K}(k)} L(\mathcal{K})$$

where $\nabla_{\mathcal{K}} L(\mathcal{K})$ represents the gradient of the loss to the knowledge base. This step will update the confidence in the extracted knowledge, the gradient of non-chosen knowledge is equal to 0.

3.4 Knowledge Memorization

Furthermore, we observe that the model struggles to fully memorize and utilize the knowledge simply added in the prompts (details can be seen in the section B in the appendix), although the simple forward-backward pipeline can work. Consequently, we introduce an additional phase named the memorization-and-retrieval (MAR).

Motivated by (Mitchell, 2021; Hofstadter and Sander, 2013), a school of cognitive scholars believes that humans categorize and conceptualize new scenarios through analogy. Based on this insight, we incorporate the creation of keys from original sentences after updating the knowledge base, treating these keys as conceptual representations. During inference, we extract concepts from sentences in a similar fashion and use them to retrieve the relevant rules.

Specifically, we employ the tf-idf algorithm to select the top k words from the paragraph containing the rule as the keys for retrieval. During inference, the same method is used to select k words from the input sentence, which are then used to retrieve the relevant rules.

Formally, the construction process can be written as:

$$C_{<t} = \text{ExtractConcepts}(x_{<t})$$

Where:

$x_{<t}$ is the token sequence before time step t, $C_{<t}$ represents the key concepts are extracted as retrieval keys, which represent the encountered scenario.

The knowledge memory is built as:

$$KM = \{(C_{<t}, r) \mid t = 1, 2, \ldots, T, r \in KB\}$$

Where $r_{<t}$ are the associated rules or responses.

And the revised forward process with MAR changing from equation 1:

for a given time step t,

$$C_t = \text{ExtractConcepts}(x_{<t})$$

$$r_t^* = \arg\max_{r<t} Similarity(KM, C_{<t})$$

Where r^* is the most relevant rule based on the similarity between C_u and $C_{<t}$, which influences token generation x_t.

$$x_t = \begin{cases} \arg\max_{v \in V} P(v|I, F, \mathcal{K}, x_{<t}), & \text{if } C_t \notin KM \\ KM(C_t), & \text{if } C_t \in KM \wedge \text{with } p \end{cases} \tag{2}$$

where KM(C_t) refers to the corresponding knowledge based on context C_t and decode by KM with probability p [2].

3.5 Knowledge Fine-tuning V.S. Knowledge Acquisition

We can note that an external knowledge base is the key component and product of our proposed work. Therefore, a natural question arises concerning the difference between our proposed framework and simple knowledge acquisition methods. Indeed, one could simply require the model to repeat answering questions multiple times, compute the accuracy of each certain knowledge, and employ them by prompt engineering or an RAG approach during the test phase.

An intuitive explanation suggests that correct knowledge ought to be universal, meaning that when a piece of knowledge consistently leads to the correct answer across various scenarios, we have more reason to believe in its accuracy.

Through the memorization process, we can regulate the frequency of knowledge utilization. If we merely rely on repetitive answering of the same questions, the model's outputs would likely be similar, thereby squandering many opportunities to verify knowledge and reinforcing spurious rationales that can result in correct answers, leading to a local optimum.

Based on this, we need only prove the following lemma: utilizing a larger number of examples

[2] We set $p = 0.5$ when training and $p = 1$ when testing.

is beneficial. Specifically, we will demonstrate that injecting incorrect knowledge leads to a higher likelihood of incorrect answers compared to correct knowledge. Although intuitive, this conclusion is not trivial; it highlights that in long reasoning chains, the impact of each new piece of knowledge can vary in terms of overall accuracy due to the spurious program.

Lemma 1 presents a weaker version, suggesting a penalty for introducing erroneous knowledge. Lemmas 2 and 3 provide complete proofs.

Lemma 1. *Formally, consider any sample (X, Y) with a reasoning depth of h steps, where each step uses knowledge denoted as $k_1, \ldots, k_h$. Suppose the knowledge injected is the i-th piece, denoted k_i. Let $P(y \mid x, k_1, \ldots, k_h)$ define the probability that the answer is y when answering x using $k_1, \ldots, k_h$. We will prove that when k_i is incorrect, $P(y = Y \mid x, k_1, \ldots, k_i, \ldots, k_h) < P(y \neq Y \mid x, k_1, \ldots, k_i, \ldots, k_h)$.*

Case 1: We consider the case where $k = 2$ in the CLUTRR dataset as an example. We assume each rule guides towards the correct answer with p_i, and the incorrect answer with q_i. A more reliable rule signifies a greater difference $p_i - q_i$. We then consider the possibility of a spurious rationale, correct answers derived from incorrect processes, typically resulting from two incorrect rules canceling their effect. For instance, in the CLUTRR dataset, an example is: "father's son is son. son's father is father", hence "father's son's father is father". We assume that in the knowledge chain, every two incorrect rules might cancel each other out with a s probability. Indeed, s is generally much smaller in most scenarios such as mathematics or standard QA tasks because illogical CoT easily leads to wrong answers (Zhang et al., 2023; Wei et al., 2022). Therefore, we can assume $s \leq 0.5$.

Consider a chain $x \to k_1 \to k_2 \to y$, without loss of generality, let's assume the injected knowledge is k_2, i.e., $i = 2$ and $q_2 > p_2$.

$$P(Y) = P(y = Y|x, k_1, \ldots, k_h) = p_1 p_2 + s q_1 q_2 \tag{3}$$

$$P(\neg Y) = P(y \neq Y|x, k_1, \ldots, k_h) = p_1 q_2 + p_2 q_1 + (1 - s) q_1 q_2 \tag{4}$$

$$\begin{aligned} & P(\neg Y) - P(Y) \\ & = [p_1 q_2 + p_2 q_1 + (1 - s) q_1 q_2] - [p_1 p_2 + s q_1 q_2] \\ & = p_1 (q_2 - p_2) + p_2 q_1 + (1 - 2s) q_1 q_2 \\ & = p_1 (q_2 - p_2) + q_1 (p_2 + (1 - 2s) q_2) \\ & > 0 \end{aligned} \tag{5}$$

Thus, it can be seen that when a rule is incorrect, under a knowledge fine-tuning framework, the mathematical expectation for confidence penalization is definitively higher. However, for knowledge acquisition, it becomes more challenging because the extraction of an incorrect rule implies that the rationale was spurious, and hence, the question no longer serves to verify the current rule.

The complete proofs of lemma $1 - 3$ can be seen in section E in the appendix.

Lemma 2. *For $P(Y) = P(y = Y|x, k_1, \ldots, k_h)$, there is a s chance that the two incorrect knowledge will result in a correct answer. In other scenarios, correct knowledge yields correct answers, and incorrect knowledge yields incorrect ones. We assume that there are n correct knowledge and m pieces of wrong ones. The probability of $P(Y) = f(m, n) = f(m) = 0 + \Sigma_{i=2}^{m} (-1)^m \cdot s^{m-1}$.*

Lemma 3. *$P(y, k_i) = P(y = Y|k_1, ...k_i, ..., k_h) > P(y = \neg Y, k_j) = P(y = Y|k_1, ..., k'_i, ..., k_h)$ in average when k_i is correct and k'_i is wrong. It will indicate that the incorrect knowledge will get more punishment in our framework so that the verification usage from the knowledge fine-tuning process is meaningful, in contrast with the knowledge extraction.*

4 Results and Discussion

4.1 Experimental Setup

To evaluate our model's performance across various tasks, we conducted experiments on the CLUTRR, cLang-8, and SST_2 datasets, which represent diverse challenges in natural language processing. we train multiple open-source and closed-source LLMs on these datasets and their different split. The details can be seen in section C due to the page limit.

Training Samples	Prompt	N-hop accuracy									Average
		2	3	4	5	6	7	8	9	10	
0	0-shot	50	15	21	18	7	20	11	22	7	20.3
	5-shot CoT (GPT-3.5)	9	32	41	44	19	24	26	24	19	27.3
	5-shot CoT (GPT-4)	63	59	62	59	50	30	46	29	20	45.1
200	LLama2-7B	88	76	29	28	31	19	13	31	16	30.7
	ChatGLM3-6B	100	89	52	53	25	17	10	33	24	39.3
	K-FT (MAR, GPT-3.5)	96	61	62	61	31	38	40	36	35	**48.1**
2000	LLama2-7B	100	100	84	76	38	48	43	73	60	66.8
	ChatGLM3-6B	100	87	75	56	29	17	7	36	33	44.7
	LLama3-8B	100	100	81	75	42	56	24	50	63	63.3
	EdgeTransformer	100	88	94	88	63	81	47	54	56	72
	HtT (train on GPT-4) [3]	87	52	57	36	33	24	36	29	20	38.5
	K-FT (base, GPT-3.5)	55	35	35	39	23	19	35	25	18	29.8
	K-FT (MAR, GPT-3.5)	100	80	68	75	54	65	51	41	47	**61.0**
	K-FT (MAR, test on GPT-4) [4]	100	83	81	80	75	71	70	54	56	**72**

Table 1: Accuracy on CLUTRR dataset across different learning & prompting strategies and on GPT-3.5-turbo-1106 model. In the prompt section, the description left to the arrow is the train prompt set, while the description right is the inject prompt set. The base stands for knowledge fine-tuning prompts for general-purpose tasks, while MAR stands for dataset-specific prompts but also executes training phase under knowledge fine-tuning framework.

4.2 Fine-tuning with Knowledge on Relational Tasks

In this section, we explore the performance of our approach across different language models[5] and datasets. We first evaluate the effectiveness of the knowledge fine-tuning framework on relational reasoning tasks through experiments on CLUTRR. We use the same setup as HtT, but since it does not release the training and test datasets, we select 2000 training examples randomly and follow the same proportion as HtT to randomly select 200 test examples (seed=42). Table 1 displays performance across various methods.

K-FT_base method only utilizes general prompt and simply concatenates knowledge (plain forward-backward pipeline) in front of the question, K-FT_MAR represent the whole framework. We can observe both of them consistently yield improvements across various settings. Our K-FT framework with a 5-shot CoT prompt results in substantial improvements (+33.7% compared to CoT) and has a competitive performance compared to fine-tuned LLMs, especially in the low-resource scenario. Furthermore, our K-FT approach using GPT-3.5 experiments outperforms the HtT with extracting rules based on GPT-4, consistent with the findings presented in section 3.5. As the number of training rounds increases, the accuracy of the knowledge base also increases (see Figure 5).

Besides, although K-FT_base only achieved minor improvements, our framework itself can induce a knowledge base and holds promise for aiding other downstream tasks. In fact, subsequent ablation experiments on our knowledge base can be completed using only a plain forward-backward process.

4.3 Q1: Does the knowledge fine-tuning framework work for other benchmarks?

We apply the proposed framework to multiple datasets requiring domain knowledge. We utilize a subset of 2000 training examples and the complete test dataset for these experiments consistently. We only use K-FT_base settings like the one in 4.2. This section primarily aims to evaluate the framework's universality. We employ gpt-3.5-turbo-1106 for training over 5 epochs, comparing the outcomes across zero-shot, 5-shot Chain-of-Thought (CoT), and the results of directly concatenating the 5-shot CoT after tuning (see Table 3 in the appendix).

In our experimentation utilizing the GPT-3.5-turbo-1106 model, we conducted training on each

[3]The knowledge base is released in HtT paper and trained on GPT-4. We use the same settings and test on GPT-3.5.

[4]We use the knowledge base trained on GPT-3.5 under our framework, and then test on GPT-4.

[5]We use Firefly (https://github.com/yangjianxin1/Firefly) to train ChatGLM and LLaMA models and we were surprised to find that the models trained with 'default' template are better than these with model-specific templates. All experiment settings are lr=$1e-05$, batchsize=32, epoch=5, 10.

dataset for a total of five epochs. The training dataset consisted of 2000 samples, consistent with the CLUTRR dataset. Our findings demonstrate a consistent improvement across both datasets, with a particularly significant enhancement observed in the quality of answers provided.

4.4 Q2: Whether explicit knowledge can significantly affect the model

This exploration aims to understand the boundaries of knowledge fine-tuning's effectiveness and its implications for model performance under various conditions. Concretely, we conducted experiments on the CLUTRR dataset, examining the impact of a perfect knowledge base, the inclusion of irrelevant knowledge, and the introduction of erroneous rules.

Experiments show that correct knowledge can lead to the correct answers significantly, and both irrelevant knowledge and wrong knowledge have negative effects on the model performance, even they cause similar effects because wrong irrelevant knowledge will not interfere with the model further.

The experiment details are shown in section F.2 in the appendix.

4.5 Q3: Does increased data yield better KB?

We train GPT on datasets with different sizes. Figure 4 shows that the number of correct rules at the beginning of the rule base, derived from training, increases with the number of training examples. The picture is shown in the appendix.

4.6 Q4: Does this framework have to use high-quality annotations?

We adopted a fixed demonstration example approach during training to explore the impact of varying proportions of substandard demonstrations on the outcome of the final knowledge base. Our experiments revealed that the process could accommodate up to 40% of low-quality demonstrations, making the automated knowledge fine-tuning process feasible (see Table 5). See Section F.4 in the appendix for details.

4.7 Knowledge fine-tuning v.s. standard fine-tuning

In further experiments, we compared the performance of knowledge fine-tuning and traditional parameter fine-tuning approaches using the ChatGLM3-6B and LLAMA3-8B models on the CLUTRR dataset. Specifically, we evaluated both methods across subsets of the data with varying sample sizes and analyzed the impact of different training samples on each approach.

For the first type of training samples, we chose the first k training samples in their original order (predominantly consisting of 2-hop instances due to the dataset's distribution). For the second type, we selected a random subset from the training data, ensuring a relatively uniform distribution of 2-hop and 3-hop samples, similar to the setup in Table 3.

Our findings in table 6 indicate that, in the random sampling scenario, knowledge fine-tuning demonstrated competitive performance with parameter fine-tuning and performed better under low-resource scenarios. Meanwhile, our framework is more comfortable for simple examples, which is easy to annotate. And K-FT performs much better than parameter fine-tuning on filter setup.

This comprehensive analysis underscores the advantages of knowledge fine-tuning in scenarios where training and testing sample distributions differ significantly and highlights its robustness in handling diverse reasoning paths.

It is also evidence that neural networks don't study the real rule but just some pattern pattern-matching or implicit vectors (Mirzadeh et al., 2024).

Detail can be viewed in section F.5 in the appendix.

5 Conclusion

We have introduced a framework for tuning LLMs with explicit knowledge, which bears many similarities to parameter fine-tuning in its processes of training and injecting knowledge. However, what sets it apart is the explicit nature of its training process, yielding a final "parameter" representation in the form of a knowledge repository. This representation is easily comprehensible, modifiable, and editable. Additionally, we have provided an initial framework to advance further research in this area. Finally, we compare our framework with parametric fine-tuning on different data sizes and distributions. Experimental results show the advantages of fine-tuning with knowledge in low-resource scenarios.

Limitations

1. The knowledge base may be excessively vast. This limitation comes with the representation of natural language or facing an open field. The knowledge extracted from LLM tends

to be too much. A summarizer might be needed to summarize the extracted knowledge to highly concentrated knowledge.

2. This methodology hinges on the ability to tap into the internal knowledge of the model. Consequently, if the model cannot perform a task with external guidance or cannot be trained to tackle it, the process becomes unfeasible and may hit a roadblock.

References

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*.

Wenhu Chen. 2023. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1120–1130, Dubrovnik, Croatia. Association for Computational Linguistics.

Xiang Chen, Ningyu Zhang, Xin Xie, Shumin Deng, Yunzhi Yao, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2022. Knowprompt: Knowledge-aware prompt-tuning with synergistic optimization for relation extraction. In *Proceedings of the ACM Web conference 2022*, pages 2778–2788.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Shuming Ma, Zhifang Sui, and Furu Wei. 2023. Why can GPT learn in-context? language models secretly perform gradient descent as meta-optimizers. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4005–4019, Toronto, Canada. Association for Computational Linguistics.

Randall Davis, Howard Shrobe, and Peter Szolovits. 1993. What is a knowledge representation? *AI magazine*, 14(1):17–17.

Zoltan Dienes and Josef Perner. 1999. A theory of implicit and explicit knowledge. *Behavioral and brain sciences*, 22(5):735–808.

Chang Gao, Haiyun Jiang, Deng Cai, Shuming Shi, and Wai Lam. 2023a. Strategyllm: Large language models as strategy generators, executors, optimizers, and evaluators for problem solving. *arXiv preprint arXiv:2311.08803*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023b. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Henner Gimpel, Kristina Hall, Stefan Decker, Torsten Eymann, Luis Lämmermann, Alexander Mädche, Maximilian Röglinger, Caroline Ruiner, Manfred Schoch, Mareike Schoop, Nils Urbach, and Steffen Vandrik. 2023. Unlocking the power of generative ai models and systems such as gpt-4 and chatgpt for higher education: A guide for students and lecturers. Hohenheim Discussion Papers in Business, Economics and Social Sciences 02-2023, Stuttgart. Urn:nbn:de:bsz:100-opus-21463.

Douglas R Hofstadter and Emmanuel Sander. 2013. *Surfaces and essences: Analogy as the fuel and fire of thinking*. Basic books.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR.

Yuxuan Lai, Chen Zhang, Yansong Feng, Quzhe Huang, and Dongyan Zhao. 2021. Why machine reading comprehension models learn shortcuts? *arXiv preprint arXiv:2106.01024*.

Pasquale Minervini, Sebastian Riedel, Pontus Stenetorp, Edward Grefenstette, and Tim Rocktäschel. 2020. Learning reasoning strategies in end-to-end differentiable proving. In *International Conference on Machine Learning*, pages 6938–6949. PMLR.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. 2024. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*.

Melanie Mitchell. 2021. Abstraction and analogy-making in artificial intelligence. *Annals of the New York Academy of Sciences*, 1505(1):79–101.

Archiki Prasad, Peter Hase, Xiang Zhou, and Mohit Bansal. 2023. GrIPS: Gradient-free, edit-based instruction search for prompting large language models. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3845–3864, Dubrovnik, Croatia. Association for Computational Linguistics.

Reid Pryzant, Dan Iter, Jerry Li, Yin Lee, Chenguang Zhu, and Michael Zeng. 2023. Automatic prompt optimization with "gradient descent" and beam search. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7957–7968, Singapore. Association for Computational Linguistics.

Abhiramon Rajasekharan, Yankai Zeng, Parth Padalkar, and Gopal Gupta. 2023. Reliable natural language understanding with large language models and answer set programming. *arXiv preprint arXiv:2302.03780*.

Vipula Rawte, Amit Sheth, and Amitava Das. 2023. A survey of hallucination in large foundation models. *arXiv preprint arXiv:2309.05922*.

Tim Rocktäschel and Sebastian Riedel. 2017. End-to-end differentiable proving. *Advances in neural information processing systems*, 30.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Tianxiang Sun, Zhengfu He, Hong Qian, Yunhua Zhou, Xuanjing Huang, and Xipeng Qiu. 2022. BBTv2: Towards a gradient-free future with large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3916–3930, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

David J. Tena Cucala, Bernardo Cuenca Grau, and Boris Motik. 2022. Faithful Approaches to Rule Learning. In *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, pages 484–493.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Boshi Wang, Sewon Min, Xiang Deng, Jiaming Shen, You Wu, Luke Zettlemoyer, and Huan Sun. 2022. Towards understanding chain-of-thought prompting: An empirical study of what matters. *arXiv preprint arXiv:2212.10001*.

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

Mengru Wang, Yunzhi Yao, Ziwen Xu, Shuofei Qiao, Shumin Deng, Peng Wang, Xiang Chen, Jia-Chen Gu, Yong Jiang, Pengjun Xie, et al. 2024a. Knowledge mechanisms in large language models: A survey and perspective. *arXiv preprint arXiv:2407.15017*.

Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. 2024b. Can LLMs reason with rules? logic scaffolding for stress-testing and improving LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7523–7543, Bangkok, Thailand. Association for Computational Linguistics.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. 2023. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Hongkang Yang, Zehao Lin, Wenjin Wang, Hao Wu, Zhiyu Li, Bo Tang, Wenqiang Wei, Jinbo Wang, Zeyun Tang, Shichao Song, et al. 2024. $memory3$: Language modeling with explicit memory. *arXiv preprint arXiv:2407.01178*.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. *Advances in neural information processing systems*, 35:30378–30392.

Zhuosheng Zhang, Aston Zhang, Mu Li, Hai Zhao, George Karypis, and Alex Smola. 2023. Multimodal chain-of-thought reasoning in language models. *arXiv preprint arXiv:2302.00923*.

Zhaocheng Zhu, Yuan Xue, Xinyun Chen, Denny Zhou, Jian Tang, Dale Schuurmans, and Hanjun Dai. 2023. Large language models can learn rules. *arXiv preprint arXiv:2310.07064*.

A Prompt for knowledge inference

Inspired by (Dai et al., 2023), we conceptualize the role of prompt as a meta-optimizer during forward propagation. Consequently, we use a universal scheme for knowledge memorization, which only requires the model to utilize its learned knowledge base to update its intrinsic knowledge (see Table 2). Through this methodology, it becomes feasible to integrate the knowledge base into the model by simply appending it in front of other prompts, such as the simple 5-shot Chain of Thought (CoT), without the need for designing specialized prompts to retrieve and apply knowledge. This approach eliminates extraneous variables, enhancing the universality of our proposed framework and the reliability of the experimental conclusions.

B Details for designing memorization phase

We identified several limitations in tuning LLMs with explicit knowledge through the conventional forward-backward method.

- LLMs often fails to accurately retrieve the provided knowledge.
- LLMs may output incorrect answers even when the correct knowledge is retrieved.

This highlights a crucial distinction: while fine-tuning enhances the model's performance, it is

The following knowledge base involves some proven knowledge that you can use to {task_description}. Utilize the knowledge base whenever possible. Only introduce your intrinsic knowledge when necessary. Note: You should use that knowledge with high confidence (which means they are more likely to be correct) to override implicit knowledge in your model. Knowledge base:
Task description SST-2: analyze whether the sentiment of a given sentence is positive or negative cLang-8: correct the grammatical errors CLUTRR: calculate the kinship

Table 2: Prompt for injecting knowledge base

important to recognize that the underlying architecture does not operate as a human-level logical reasoning engine. Rather than relying on logical deductions based on knowledge, the model's core function is driven by parameter-based pattern recognition. Consequently, the presence of correct knowledge does not always equate to accurate reasoning.

Nevertheless, the model retains fundamental capabilities for analysis and generation, functioning as a reasoning engine in a limited sense. Similar inputs will yield similar word vectors and outcomes. To address these limitations, we introduce a memorization phase into the traditional fine-tuning process. This phase mirrors the human reasoning mechanism, where analogies are drawn between novel and previously encountered situations. For example, humans often respond to a story by saying, "The same thing happened to me," even when the circumstances are superficially different. Analogical reasoning is central to how humans navigate and make sense of new experiences (Mitchell, 2021).

In our system, we simplify this analogy-making process by summarizing the contextual use of knowledge. This allows the model to better index and retrieve relevant information. To enhance this process, we recommend placing the model's scenario requirements before the rules in the prompt, facilitating more efficient indexing and recognition of multiple rules when answering complex queries. For simplicity and reliability in the validation process, we just leverage a simple TF-IDF-based method, which demonstrate this pipeline has a wide generalization.

C Experimental Setup

C.1 Datasets and Metrics

CLUTRR The CLUTRR dataset is designed to test the capabilities of understanding natural language and performing relational reasoning about family relationships. We use accuracy as its evaluation metric. We use the same setup with HtT (Zhu et al., 2023).

cLang-8 The Lang-8 dataset is a collection of texts from language learners, including their original writings and the corrections made by native speakers, from the Lang-8 language learning platform. It is commonly used in NLP for error correction and language learning research. Evaluation metrics for tasks are the same as the original paper, including recall, precision, and F1 score, focusing on the model's ability to accurately detect and correct errors compared to human corrections.

However, due to the low quality of the dataset itself, we have regenerated the gold labels using GPT-4 Turbo-1106. For detailed information, please refer to Section 3 in supplementary material.

SST_2 The SST2 dataset is a popular dataset used in natural language processing (NLP) for sentiment analysis tasks. It consists of sentences from movie reviews, labeled with their sentiment polarity as either positive or negative. It's evaluated using an accuracy metric.

D Cleaning of Lang-8 Dataset

Building upon the foundation of cLang-8, we cleaned a test set consisting of the last 2000 entries from the total dataset with 2 million items. We leveraged GPT-4 for zero-shot responses, which were subsequently treated as gold labels. Given our reliance on GPT's zero-shot capabilities for cleaning, we refrain from comparing performance against zero-shot capabilities during the testing phase.

The cleaning process is mainly aimed at the problems such as many text emojis and inappropriate punctuation in the lang8 data set, which will cause trouble to the language model.

E Complete proofs

Lemma 1. (cont.)

Proof. As mentioned above, we have found that $P(\neg Y) > P(Y)$ if there is a knowledge k_2 such that $h = 2$ and $q_2 > p_2$, and it is obvious that the difference between $P(\neg Y)$ and $P(Y)$ is larger when q_2 is larger.

For the more general case, we consider h to be any natural number. The first two pieces of knowledge satisfy the above proof and we can conclude that $q_{k_1,k_2} > p_{k_1,k_2}$. If we regard the two knowledge k_1, k_2 as one new compositional knowledge (k_1, k_2), we can reduce the h to $h-1$ and the knowledge (k_1, k_2) and k_3 also satisfy the condition of the above lemma. Then we will draw a conclusion that $P(\neg Y) > P(Y)$ still holds such that $h = 3$ and $q_{k_1,k_2} > p_{k_1,k_2}$. Therefore, $P(\neg Y) > P(Y)$ holds for any $h \in \mathbb{N}$ as long as there is at least one incorrect knowledge in the reasoning chain and the difference is larger when q is larger.

□

In fact, our abstract lemmas omit several details, such as whether the memorization or retrieval is flawless, whether the model comprehends the knowledge, and whether the model's output capacity is sufficient. However, since these features uniformly affect $P(Y)$ and $P(\neg Y)$, we can see, without further proof, that these do not impact the qualitative relationship described above, though they might slightly alter the quantitative differences.

Besides, the integration of knowledge with high confidence also reduces the actual reasoning hops and then the proportion of false positive examples, which can be achieved by carefully designing the confidence update strategy.

Lemma 2. For $P(Y) = P(y = Y|x, k_1, \ldots, k_h)$, there is a s chance that the two incorrect knowledge will result in a correct answer. In other scenarios, correct knowledge yields correct answers, and incorrect knowledge yields incorrect ones. We assume that there are n correct knowledge and m pieces of wrong ones. The probability of $P(Y) = f(m, n) = f(m) = 0 + \Sigma_{i=2}^{m}(-1)^m \cdot s^{m-1}$.

Proof. For every two knowledge k_1, k_2, there are four cases including both are correct or incorrect, k_1 is correct while k_2 is incorrect and vice versa. Two correct knowledge will be bound to the correct answer and two incorrect knowledge will have s chance to lead to it. Otherwise, we will get an incorrect answer. Notice that the correct knowledge will not affect the correctness of the final answer. So we only consider the relation between several incorrect knowledge in reasoning chain $k_1, \ldots, k_h$ and $P(Y)$. Therefore, the following knowledge symbol k will only refer to the incorrect knowledge.

Base Cases:

When $m = 1$, $P(Y) = P(y = Y|x, k_1) = 0 = f(1)$,

$m = 2, P(Y) = P(y = Y|x, k_1, k_2) = s = 0 + (-1)^2 \cdot s^{2-1} = f(2)$.

Inductive Step:

Then, we will prove if $f(c)$, that is $f(c, n)$, holds then $f(c+1)$ holds. $P(y = Y|x, k_1, \ldots, k_{c+1}) = (1-P(y = \neg Y|x, k_1, \ldots, k_c))\cdot s = (1-f(c))\cdot s = s - (s^2 \ldots + (-1)^c \cdot s^{c-1}) = s - s^2 ... + (-1)^{c+1} \cdot s^c = f(c+1)$. Therefore, if $f(c)$ holds, then $f(c+1)$ also holds.

By the principle of mathematical induction, we have proved that $f(m, n) = f(m) = 0 + \Sigma_{i=2}^{m}(-1)^m \cdot s^{m-1}$ holds for all positive integers m.

□

Lemma 3. $P(y, k_i) = P(y = Y|k_1, ...k_i, ..., k_h) > P(y = \neg Y, k_j) = P(y = Y|k_1, ..., k_i', ..., k_h)$ in average when k_i is correct and k_i' is wrong. It will indicate that the incorrect knowledge will get more punishment in our framework so that the verification usage from the knowledge fine-tuning process is meaningful, in contrast with the knowledge extraction.

Proof. We use n and m to represent the number of correct knowledge and incorrect knowledge respectively, and $f(x)$ means the probability of $P(Y)$. On average, we should consider all the cases that $k_j, j \neq i$ is either correct or incorrect and any reasoning steps j. So $P(Y, k_i) - P(Y, k_i')$ in average is equal to

$\Sigma_{h=1}^{\infty}[\Sigma_{m=0}^{m=h-1} C_h^m p^m q^{h-1-m} f(m) - \Sigma_{m=0}^{m=h-1} C_h^m p^m q^{h-1-m} f(m+1)] \geq 0$ with $p + q = 1$

$= \Sigma_{m=0}^{\infty}\Sigma_{h=m+1}^{\infty} C_h^m p^m q^{h-1-m} f(m) - \Sigma_{m=0}^{\infty}\Sigma_{h=m}^{\infty} C_h^m p^m q^{h-1-m} f(m+1)$

$= \Sigma_{m=0}^{\infty}[\frac{p^m f(m)}{m!}\Sigma_{h=m+1}^{\infty}\frac{h!}{(h-m)!}q^{h-1-m}] - \Sigma_{m=1}^{\infty}[\frac{p^m f(m+1)}{m!}\Sigma_{h=m}^{\infty}\frac{h!}{(h-m)!}q^{h-1-m}]$

$= \Sigma_{m=0}^{\infty}[\frac{p^m f(m)}{m!}\Sigma_{h=m+1}^{\infty}((\Pi_{k=m+1}^{h} k) \cdot q^{h-1-m})] - \Sigma_{m=0}^{\infty}[\frac{p^m f(m+1)}{m!}\Sigma_{h=m}^{\infty}((\Pi_{k=m+1}^{h} k) \cdot q^{h-1-m})]$

Let $t(q) = \Sigma_{i=0}^{\infty} q^i = \frac{1}{1-q}$, we can find that $\Sigma_{h=m+1}^{\infty}(\Pi_{k=m+1}^{h} k) q^{h-1-m} = t^{(m)}(q)$ where $t^{(m)}(q)$ means n-th derivative of $t(q)$.

$t^{(m)}(q) = \frac{m!}{(1-q)^{m+1}}$, therefore, we can simplify above formula into

$$\Sigma_{m=0}^{\infty}[\frac{p^m f(m)}{m!} \cdot \frac{m!}{(1-q)^{m+1}}] - \Sigma_{m=0}^{\infty}[\frac{p^m f(m+1)}{m!} \cdot \frac{m!}{(1-q)^{m+1}}]$$
$$= \Sigma_{m=0}^{\infty} \frac{f(m)}{p} - \Sigma_{m=0}^{\infty} \frac{f(m+1)}{p}$$
$$= \Sigma_{m=0}^{\infty} \frac{f(m)-f(m+1)}{p}$$
$$= \Sigma_{m=0}^{\infty} \frac{\Sigma_{i=1}^{m}(-1)^{i+1}s^i - \Sigma_{i=1}^{m+1}(-1)^{i+1}s^i}{p}$$
$$= \frac{1-s}{p} + \Sigma_{m=1}^{\infty} \frac{(-1)^{m+1}s^{m+1}}{p}$$
$$= \frac{(1-s)^2+s^2}{(1-s)p} \geq 0 \qquad \square$$

F Supplementary and ablation experiment

F.1 Q1: experiments on other benchmarks

F.2 Q2: Whether explicit knowledge can significantly affect the model

1. Utilizing a kinship calculator, we build all the necessary knowledge for each testing sample in the format "{}'s {} is {}", which was simply concatenated to the front of each question. This approach was examined under 0-shot prompt and 5-shot CoT settings to evaluate the upper limit of knowledge fine-tuning (see Table 4).

2. We randomly incorporated a certain proportion of irrelevant knowledge into the knowledge base to observe its effect on model performance (see Figure 3a).

3. Explore the effect of error knowledge (see Figure 3b, 3c).

Scenario	average accurancy
Zero-shot	71.5%
5-shot CoT	76.6%

Table 4: Accuracy for adding perfect knowledge base

F.3 Q3: Does increased data yield better KB?

There are related pictures.

F.4 Q4: Does this framework have to use high-quality annotations?

The training process asks the model to express the prior knowledge needed in a unified format, relying on the 5-shot CoT prompts. Naturally, high-quality demos lead to a high-quality learning process. However, an excessively stringent requirement on the quality of demos could constrain the generality of the framework or inflate the human labor cost.

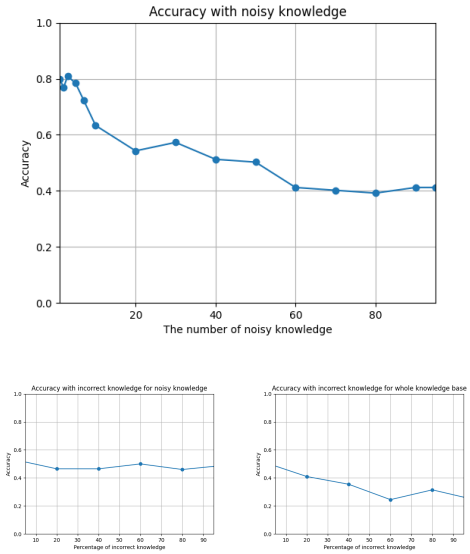

Figure 3: Accuracy for adding noisy knowledge and incorrect knowledge

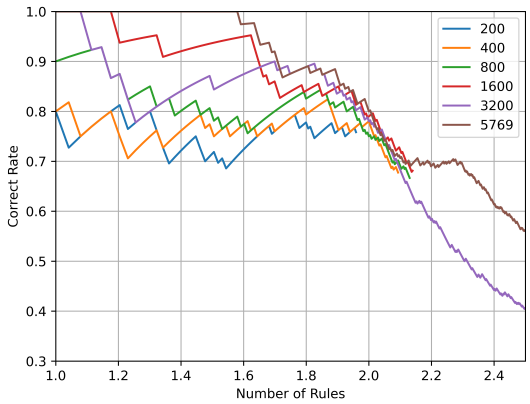

Figure 4: Correctness of rules with increased train dataset. The x-axis is a 10-based log axis. With larger train datasets, rule fine-tuning extracts more, accurate rules.

Benchmark	SST-2	cLang-8		
Metric	accuracy	recall	precision	F0.5
0-shot prompt	87.7	-	-	-
5-shot CoT	92.1	**47.2**	31.2	42.8
5-shot K-FT	**93.3**	46.9	**39.3**	**45.1**

Table 3: Performance on SST-2 and cLang-8 dataset, our method (K-FT) can improve performance.

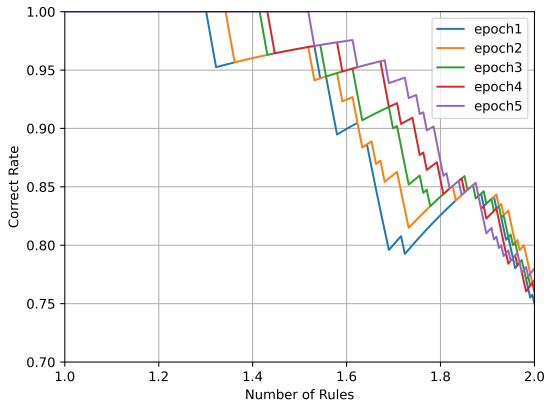

Figure 5: Correctness of rules with further training. The x-axis is a 10-based log axis. We can see the correctness of rules is improved during further training.

good demos	**bad demos**	**correct rate of learned KB**
0	5	16.22%
1	4	19.52%
2	3	35.16%
3	2	57.51%
4	1	77.44%
5	0	82.76%

Table 5: Statistics about prompts with different quality

F.5 Knowledge fine-tuning v.s. standard fine-tuning

Under the original order condition, knowledge fine-tuning significantly outperformed parameter fine-tuning in the 200-2000 sample size range. This result aligns with our expectations, given that parameter fine-tuning struggles with large discrepancies between training and test samples (e.g., training on predominantly 2-hop samples while testing involves 10-hop samples). Conversely, knowledge fine-tuning adjusts primarily based on knowledge and relies on the correctness of the final answer as a distant supervision signal, which benefits from shorter reasoning paths to avoid incorporating erroneous knowledge (like the spurious programs in Weakly supervised learning).

Given that simpler examples are easier and less costly to obtain and annotate than complex ones (with the annotation difficulty not increasing linearly), this finding is significant. Moreover, our approach, based on the MAR framework, does not suffer adversely from the presence of 3-hop relations, thus maintaining robust performance in multi-hop scenarios.

Furthermore, the CLUTRR dataset contains significant repetition, which hindered our approach's ability to extract sufficient rules for answering. Therefore we designed an experiment that chose the samples with different relation paths from 2000 random training samples. Under this setting, the discrepancy between knowledge fine-tuning and parameter fine-tuning was substantial. We finally deliberately selected 381 non-redundant samples related to familial relationships, and knowledge fine-tuning exhibited stronger performance than each advanced LLM.

number	K-FT (3.5)	6B	8B
200 (2-hop)	**44**	25	23
2000 (2-hop)	**45**	33	26
200 (2,3-hop)	**48**	39	46
2000 (2,3-hop)	61	45	**62**
381* (2,3-hop)	**57**	47	48

Table 6: The experiments about K-FT and parameter fine-tuning on different setup of CLUTRR.
*381 is filtered from 2000