ELAD: Explanation-Guided Large Language Models Active Distillation

Yifei Zhang, Bo Pan, Chen Ling, Yuntong Hu, Liang Zhao
Department of Computer Science, Emory University
{yifei.zhang2,bo.pan,chen.ling,yuntong.hu,liang.zhao}@emory.edu

Abstract

The deployment and application of Large Language Models (LLMs) is hindered by their memory inefficiency, computational demands, and the high costs of API inferences. Traditional distillation methods, which transfer the capabilities of LLMs to smaller models, often fail to determine whether the knowledge has been sufficiently transferred, potentially resulting in high costs or incomplete distillation. In this paper, we propose an Explanation-Guided LLMs Active Distillation (ELAD) framework that employs an active learning strategy to optimize the balance between annotation costs and model performance. To improve the efficiency of sample selection, we introduce an explanation-guided sample selection method that identifies samples challenging its reasoning by exploiting uncertainties in reasoning explanation steps. Additionally, we present a customized LLM-annotated explanation revision technique where the teacher model detects and corrects flaws in the student model's reasoning. Our experiments across various reasoning datasets demonstrate that our framework significantly enhances the efficiency of LLMs knowledge distillation.

1 Introduction

The advancement of LLMs (Brown et al., 2020; Hoffmann et al., 2022; Thoppilan et al., 2022; Touvron et al., 2023) has significantly impacted the natural language processing (NLP) domain, showcasing excellent in-context learning and complex reasoning capabilities. Yet, the deployment of these models is hindered by their extensive parameter count, leading to significant computational demands and financial burdens. For instance, deploying LLMs with 100-200 billion parameters would require a cluster of NVIDIA A100 GPUs, where each GPU costs $\$30,000$ in today's market. While cloud computing offers a solution, the costs associated with such services can quickly accumulate. Specifically, a cluster of A100 GPUs costs upwards of $\$25$ per hour, which may lead to a staggering $\$18,000$ monthly if operated non-stop[1]. This financial barrier makes it impractical for many institutions and research labs to adopt LLMs widely, especially in resource-constrained environments (Bai et al., 2024), limiting their audience and application scope. Additionally, relying on API calls to access pre-trained LLMs, e.g., GPT-4, also presents its challenges, including high usage fees, inability to run models locally for customization or fine-tuning, potential data transmission issues, and privacy concerns (Yao et al., 2023c).

Recent research on LLMs *knowledge distillation* (Hinton et al., 2015) enables smaller models to achieve performance similar to LLMs by transferring reasoning capabilities to them, making them more computationally efficient. Tang et al. (2019), Wang et al. (2021), and Arora et al. (2022) demonstrate the training of smaller models using pseudo-labels generated by LLMs, wherein LLMs act as "teachers" to supervise the *fine-tuning* of these "student" models. Recent works (Magister et al., 2022; Ho et al., 2022; Chan et al., 2022; Zhang et al., 2023; Li et al., 2023; Hsieh et al., 2023; Zhao et al., 2024; Pan et al., 2024a) focus on multi-task fine-tuning of student models. They utilize chain-of-thought (CoT) reasoning (Wei et al., 2022) to generate both explanations and final answers generated by LLMs as pseudo-labels to jointly supervise small model fine-tuning. However, a major issue with existing fully supervised learning methods is that they do not sense whether the knowledge has been sufficiently distilled into the small model. Insufficient distillation can lead to suboptimal performance of the small model, while excessive distillation may incur unnecessarily high costs.

To address this challenge, we take LLMs as an

[1] https://charshift.com/llm-true-cost/

Findings of the Association for Computational Linguistics ACL 2024, pages 4463–4475
August 11-16, 2024

agent that guides small language models toward progressive improvement. Throughout this process, the LLMs (teacher) can sense the weaknesses of the small language model (student) and customize its teaching accordingly. Formally, we propose an Explanation-Guided LLMs Active Distillation (ELAD) framework that significantly enhances active learning through the explanations generated by language models. In each iteration, the framework encompasses a student reasoning task and a teacher reasoning task: the student model identifies samples it struggles to predict accurately and reasonably; subsequently, for these selected samples, the teacher model reviews the student's explanations, correcting any erroneous reasoning.

However, first, it is nontrivial to tackle the student task. Current sample selection methods typically focus on finding samples with the wrong predictions (Lewis, 1995; Ren et al., 2021; Bansal and Sharma, 2023), but even if the prediction is correct, the reasoning process can be wrong or flawed. The selection of samples with bad reasoning goes beyond it and is yet to be well explored, which requires the student model to faithfully self-inspect its step-by-step explanation of its prediction and locate the flaw. To address this, we propose a novel explanation-guided sample selection method that identifies the samples that trouble its reasoning by exploiting explanation stepwise uncertainties. Second, how the teacher senses and corrects the flaws in student model reasoning is also a challenging problem. Merely prompting teacher and student to generate their respective explanations separately and compare their difference is problematic because a prediction could be led by different reasoning processes and different explanations (e.g., *Rashomon Effect* (Roth and Mehta, 2002)). We need the teacher model to check the student model's explanation, locate the problem within its reasoning, and correct it, which is not well explored. To accomplish this, we propose a customized LLM-annotated explanation revision technique. It entails sequentially prompting the LLM with the explanation from the small model and then asking it to assess whether the current step is reasonable or necessitates revision.

We evaluate our framework across six reasoning benchmarks, comparing it against existing sample selection methods for active learning and LLMs explanation and answer generation methods. Our findings indicate that the proposed framework notably enhances annotating efficiency.

We summarize our main contributions as follows: *a*) An *Explanation-Guided LLMs Active Distillation* framework that enhances active learning, guided by explanations from small models, during the distillation of LLMs to smaller models. *b*) An explanation-guided sample selection method that identifies the samples that trouble the reasoning of language model by exploiting explanation stepwise uncertainties. *c*) A customized LLM-annotated explanation revision technique that allows LLM to teach customized knowledge by guiding the LLM to pinpoint and correct inaccuracies in the reasoning steps of small models. *d*) Extensive experiments demonstrate that the proposed framework can improve annotating efficiency.

2 Related Work

2.1 LLMs Knowledge Distillation

DistilBert (Sanh et al., 2019) achieves efficient distillation of the BERT transformer into a student model with minimal performance loss. Tinybert (Jiao et al., 2019) introduces a loss term for matching hidden states between teacher and student. Works like Magister et al. (2022), Finetune-CoT (Ho et al., 2022), KNIFE (Chan et al., 2022), SCoTD (Li et al., 2023), and Distilling step-by-step (Hsieh et al., 2023) emphasize multi-task fine-tuning of student models using both CoT reasoning explanations and LLM-generated answers. Distilling step-by-step (Hsieh et al., 2023) specifically uses task-specific prefixes in prompts to tailor model responses. Li et al. (2022) investigate methods for generating explanations to aid student model learning. SOCRATIC CoT (Shridhar et al., 2023) decomposes problems into subproblems to guide reasoning, while SCOTT (Wang et al., 2023) uses teacher-generated explanations for training on a counterfactual reasoning objective, promoting self-consistency.

2.2 Efficient Annotating and Active Learning

Bansal and Sharma (2023) introduced a method using model uncertainty (Lewis, 1995), dataset density (Ren et al., 2021), and conditional informativeness (Bansal and Sharma, 2023) for one-shot informative sample selection to enhance annotation efficiency. However, these one-shot approaches fail to sense the sufficiency of annotation. Addressing this, active learning, particularly the pool-based paradigm, has been recognized as a crucial technique for reducing annotation costs (Krishnakumar, 2007; Ren et al., 2021). For NLP tasks, the most common strategy is based on the entropy of predicted tokens for sampling (Zhang et al., 2022). Additionally, Yao et al. (2023a) proposed a data diversity-based active learning sampling strategy, leveraging explanation annotations, with a human-in-the-loop setting. BSDETECTOR (Chen and Mueller, 2023) introduces an uncertainty quantification technique for black-box LLMs, focusing on consistency (Wang et al., 2022).

2.3 Explanation Generation for LLMs

Extractive explanations (Lei et al., 2016; Yu et al., 2019) focus on identifying key elements within the input that justify a prediction. However, they are limited in explaining complex reasoning tasks that require detailed natural language explanations (free-text explanation) (Camburu et al., 2018; Rajani et al., 2019). Narang et al. (2020) advanced this by training models to generate explanations post-prediction. Self-rationalization models, such as those discussed by Wiegreffe et al. (2020), aim to simultaneously predict labels and generate text-based explanations. STaR (Zelikman et al., 2022) generates explanations by augmenting ground truth answers as hints when predicted answers are incorrect. Wei et al. (2022) introduced CoT prompting, which uses demonstrations in LLM prompting to elicit intermediate reasoning steps for explanations. Kojima et al. (2022) demonstrated the zero-shot reasoning capabilities of LLMs by employing prompts like "Let's think step by step" to generate an explanation. Tree of Thought (Yao et al., 2023b; Long, 2023) generates reasoning explanations by recursively decomposing complex questions into simpler sub-questions, solving them individually, and integrating their answers.

3 Preliminary Study

3.1 LLMs Reasoning

In the zero-shot CoT prompting scenario, prompting a question q to an LLM triggers the generation of the completion which consists of an answer a and reasoning path (explanation) r, modeled as $(a, r) \sim P(a, r \mid q)$. This process unfolds in an auto-regressive manner, generating r before a and formalizing the conditional probability of the answer as $P(a \mid q) = P(a \mid q, r) \times P(r \mid q)$, where $P(a \mid q, r)$ represents the probability of a given both q and r, and $P(r \mid q)$ denotes the probability of r given q. In few-shot scenarios, demonstration triplets $\{(q_i^p, a_i^p, r_i^p)\}_{i=1}^m$ are included before q in the prompt, facilitating contextual guidance and reasoned answer generation, with m indicating the number of demonstrations in the prompt.

The CoT prompting approach facilitates sequential reasoning in LLMs, generating a series of reasoning steps $r = \{s_1, s_2, \ldots, s_n\}$, where n is the total number of steps. Each step s_i contributes cumulatively to the reasoning explanation, culminating in the final answer. Specifically, the probability of generating the explanation r given the question q, denoted $P(r \mid q)$, is expressed as a product of conditional probabilities, representing the step-by-step reasoning:

$$P(r \mid q) = \prod_{i=1}^{n} P(s_i \mid q, s_1, \ldots, s_{i-1}) \tag{1}$$

where each s_i is predicated on the question q and the preceding steps $s_1, \ldots, s_{i-1}$.

3.2 LLMs Knowledge Distillation

LLMs Knowledge distillation entails a process wherein a large teacher language model $\mathcal{T}$ transfers its knowledge to a small student language model $\mathcal{S}$. In this framework, given an unlabeled dataset $\mathcal{U}$, the teacher model generates a pseudo-answer and a pseudo-explanation for each question q in dataset $\mathcal{U}$. These outputs are represented as the answer-explanation pair (completion) $(\hat{a}, \hat{r})$. This generation process is modeled as:

$$(\hat{a}, \hat{r}) \sim \mathcal{T}(a, r \mid q) \tag{2}$$

The result is a collection of triplets $\{(q, \hat{a}, \hat{r})\}^{|\mathcal{U}|}$ as the training dataset $\mathcal{D}$. Subsequently, the student model $\mathcal{S}$ is fine-tuned using $\mathcal{D}$, employing the standard language modeling loss, formulated as:

$$\max E_{(q,\hat{a},\hat{r})} \sim_{\mathcal{D}} [\mathcal{S}(\hat{a}, \hat{r} \mid q)] \tag{3}$$

4 Problem Setup

In the context of knowledge distillation of LLM $\mathcal{T}$, we address the problem of efficiently annotating an unlabeled question set $\mathcal{U}$. This involves strategically selecting a subset $\mathcal{Q}$ from $\mathcal{U}$, where $\mathcal{Q} \subset \mathcal{U}$ and $|\mathcal{Q}| < |\mathcal{U}|$. Our study's primary goal is to enhance the performance of a smaller model $\mathcal{S}$ through fine-tuning with a training dataset $\mathcal{D} = \{(q_i, \hat{a}_i, \hat{r}_i)\}^{|\mathcal{Q}|}$, where the LLM annotates an answer $\hat{a}$ and a corresponding explanation $\hat{r}$ for each question $q \in \mathcal{Q}$, forming $\mathcal{D}$. The objective is to achieve a performance with $\mathcal{S}$ that is comparable

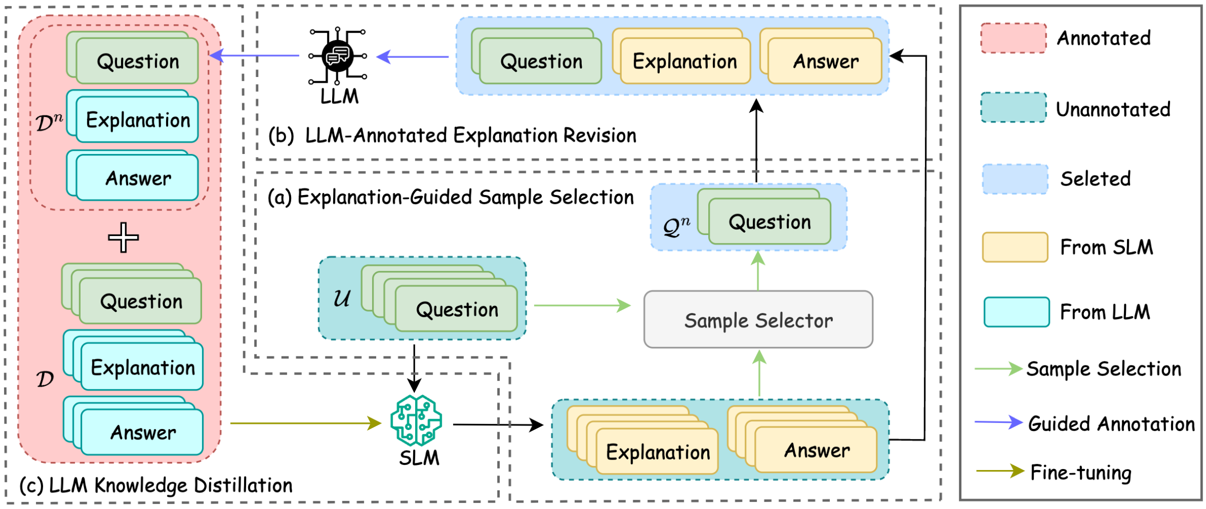


Figure 1: Overview of the Explanation-Guided LLM Active Distillation (ELAD) framework: (a) illustrates the Explanation-Guided Sample Selection method, (b) depicts the Customized LLM-Annotated Explanation Revision technique, and (c) showcases the LLM Knowledge Distillation (small model fine-tuning) process.

to that of $\mathcal{T}$ while minimizing the size of the annotated dataset $\mathcal{Q}$. This approach aims to maximize the efficiency of the small model and minimize the amount of annotated data required from the LLM.

5 Methodology

In this section, we first present the overview of our proposed *Explanation-Guided LLMs Active Distillation* framework. We then proceed to present a novel explanation-guided sample selection method. Lastly, we present a customized LLM-annotated explanation revision method.

5.1 Explanation-Guided LLMs Active Distillation Framework

We propose a novel *Explanation-Guided LLMs Active Distillation* framework to optimize the trade-off between sufficient distillation and annotation costs for LLM knowledge distillation tasks via an active learning strategy. Our overall framework is depicted in Figure 1. We first collect an unlabeled dataset $\mathcal{U}$. At the n-th iteration of active learning, during the *sample selection phase*, as depicted in Figure 1 (a), we employ the standard pool-based setting. The small model $\mathcal{S}$ generates answers and explanations (completions) for all samples $q \in \mathcal{U}$, resulting in a set $\{(q_i, a_i, r_i)\}^{|\mathcal{U}|}$. Then, an *explanation-guided sample selection* method f (details to be provided in Section 5.2) is adopted to select m samples with high *uncertainty* in their generated answers and explanations from this set, forming the selected subset. This process can be represented as

$$\mathcal{Q}^n = f(\{(q_i, a_i, r_i)\}^{|\mathcal{U}|}; m) \tag{4}$$

We then create the batch $\mathcal{B}^n$ comprising triples $\{(q_i, a_i, r_i)\}^m$ for each $q \in \mathcal{Q}^n$. Subsequently, we progress to the *annotation phase* of active learning, as illustrated in Figure 1 (b). The customized LLM-annotated explanation revision function g (to be discussed in Section 5.3) annotates completions $(\hat{a}, \hat{r})$ for the selected samples in $\mathcal{Q}^n$ using the LLM $\mathcal{T}$, guided by the completion (a, r) generated by the small model, represented as

$$\mathcal{D}^n = g(\{(q_i, a_i, r_i)\}^m; \mathcal{T}) \tag{5}$$

For each $q \in \mathcal{Q}^n$, this results in the dataset $\mathcal{D}^n = \{(q_i, \hat{a}_i, \hat{r}_i)\}^m$ for small model fine-tuning. The datasets are updated by removing $\mathcal{Q}^n$ from $\mathcal{U}$ and adding $\mathcal{D}^n$ to the cumulative training set $\mathcal{D}$. Finally, in the *model update phase* of active learning, depicted in Figure 1 (c), we fine-tune the small model $\mathcal{S}$ on the training set $\mathcal{D}$. This process is repeated until the LLM annotating (labeling) budget Bu is depleted or other stopping criteria are met (e.g., marginal improvement of the small model falls below a certain threshold). The overall algorithm is summarized in Algorithm 1.

5.2 Explanation-Guided Sample Selection

This section presents the *explanation-guided sample selection* method to select samples with high *uncertainty*. This *uncertainty* stems from the complexity and instability inherent in the step-by-step

Algorithm 1: ELAD

Require: $\mathcal{U}, \mathcal{D}, \mathcal{T}, \mathcal{S}$, annotating budget Bu, number of samples to select each iteration m

Ensure: Fine-tuned student model $\mathcal{S}$

while $Bu > 0$ **do**
- Generate answers and explanations for samples from $\mathcal{U}$ using small model $\mathcal{S}$
- Select m most informative samples as $\mathcal{Q}^n$ using Equation 4 to form batch $\mathcal{B}^n$
- Annotate $\mathcal{B}^n$ using LLM $\mathcal{T}$ as per Equation 5, obtaining $\mathcal{D}^n$
- Update $\mathcal{D} \leftarrow \mathcal{D} \cup \mathcal{D}^n$; $\mathcal{U} \leftarrow \mathcal{U} \setminus \mathcal{Q}^n$
- Retrain $\mathcal{S}$ on updated $\mathcal{D}$
- $Bu \leftarrow Bu - m$

end while

Perform final retraining of $\mathcal{S}$ on $\mathcal{D}$

reasoning process. We estimate it across two dimensions: 1) Intra-explanation uncertainty, which explores the uncertainty within individual steps of an explanation, and 2) Inter-explanation uncertainty, which examines the uncertainty across the aggregated answers from different reasoning paths.

Intra-explanation uncertainty As stated in Section 3, for an explanation $r = \{s_1, s_2, \ldots, s_n\}$, each reasoning step s_i builds upon the question and all preceding steps and influences subsequent steps and the final answer. To address the challenge of estimating the uncertainty in the explanation generation resulting from step-by-step reasoning, we introduce a novel method for estimating intra-explanation uncertainty. This method utilizes a step-wise technique to evaluate the consistency of final answers, whether they are conditioned on specific reasoning steps or not. By comparing outcomes in both scenarios, we effectively measure the uncertainty associated with each step in the explanation. To be more specific, for the i-th reasoning step, the reasoning and answer before the i-th step can be written as:

$$(a, s_{\geq i}) \sim \mathcal{S}(a, s_{\geq i} \mid q, s_{<n}) \tag{6}$$

Similarly, the reasoning process conditioned on the i-th reasoning step can be written as:

$$(\hat{a}, s_{>i}) \sim \mathcal{S}(\hat{a}, s_{>i} \mid q, s_{<i}, s_i) \tag{7}$$

where $\hat{a}$ is the sampled answer conditioned on i-th reasoning step. The above two scenarios are illustrated in Figure 2. We prompt the small model with $(q, s_{<i})$ and $(q, s_{\leq i})$ for each of the n steps to obtain the corresponding answers, resulting in a set $\{(\hat{a}_i, a_i)\}_{i=1}^{n}$ that records the outcomes. The uncertainty of the explanation is then quantified by calculating the frequency of instances where predictions remain unchanged despite the removal of a reasoning step in the prompt as

$$\mathcal{H}_{Reasoning} := \frac{1}{n} \sum\nolimits_{i=1}^{n} \mathbb{I}\left(\hat{a}_i = a_i\right) \tag{8}$$

where $\mathbb{I}\left(\hat{a}_i = a_i\right)$ is an indicator function that returns 1 (or 0) if the predicted answer is unchanged (or not). This intra-reasoning uncertainty score $\mathcal{H}_{Reasoning}$ measures the uncertainty of a single explanation.

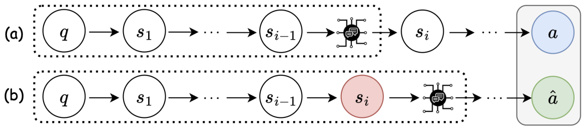


Figure 2: (a) illustrates reasoning not conditioned on the i-th reasoning step; (b) depicts reasoning conditioned on the i-th reasoning step.

Inter-explanation uncertainty The answers and explanations generated by language models can exhibit diversity due to the randomness introduced by sampling temperature. To assess the uncertainty arising from this randomness, we propose a consistency-based method for evaluating inter-explanation uncertainty. For each question, we apply a multiple-path decoding strategy, prompting the model k times to generate k distinct reasoning paths. This process can produce different final answers, leading to N unique answer values, with each unique value occurring c_i times. We assess the consistency among these multiple final answers by calculating the frequency of occurrence c_i for every unique answer and subsequently computing the probability of each answer as $p_i = \frac{c_i}{k}$. To quantify the uncertainty in the probability distribution of the output answers derived from multiple promptings, we utilize Shannon entropy, calculated as follows:

$$\mathcal{H}_{Consistency} := -\sum\nolimits_{i=1}^{N} p_i \log(p_i) \tag{9}$$

This inter-explanation uncertainty score serves as an indicator of the model's reasoning uncertainty arising from different reasoning paths.

Overall Uncertainty Estimation and Sample Selection Based on the two types of uncertainty illustrated in Equations 8 and 9, we define the overall reasoning uncertainty $\mathcal{H}$ as:

$$\mathcal{H} = \mathcal{H}_{Consistency} + \sum\nolimits_{i=1}^{k} \mathcal{H}_{Reasoning}^{(i)} \tag{10}$$

For all samples in dataset $\mathcal{U}$, we select m samples with the highest uncertainty scores to form the selected subset $\mathcal{Q}$. Based on the above, the Equation 4 is formalized as:

$$\mathcal{Q} = \underset{q \in \mathcal{U}}{\text{argmax-m}}\, \mathcal{H}(q) \tag{11}$$

where $\mathcal{H}(q)$ denotes the computed uncertainty score for a question q using Equation 10.

5.3 Customized LLM-Annotated Explanation Revision

After selecting the samples the small model troubles the reasoning, the next step is transferring knowledge from the LLM to the small model. This process involves using the LLM to generate pseudo-completion to fine-tune small model. To achieve this, we introduce a *customized LLM-annotated explanation revision* technique. This approach leverages the advanced capabilities of the LLM to provide customized guidance to the small model by allowing the LLM to detect and correct flaws in the small model's reasoning.

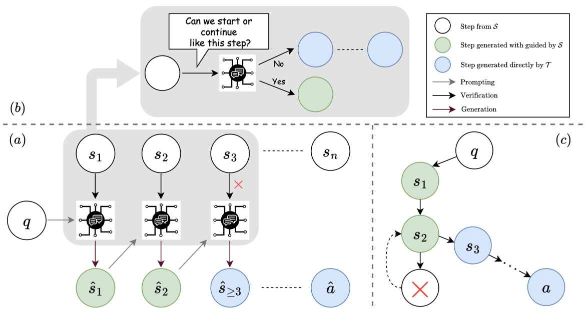


Figure 3: Customized LLM-Annotated Explanation Revision. (a) and (b) illustrate the process by which the LLM is prompted to revise the explanation and answer from the small model. (c) shows the DFS-based reasoning steps searching strategy.

As Figure 3 (a) and (b) illustrate, our method prompts the LLM to annotate customized completion for the selected questions, conditioned on the output completion provided by the small model to make detection and possible revision (correction), as $(\hat{r}, \hat{a}) \sim \mathcal{T}(\hat{r}, \hat{a} \mid q, r, a)$. Specifically, we employ a Depth-First Search (DFS)-based strategy (Yao et al., 2023b), where for each reasoning step generated by the small model, the LLM is prompted to perform verification to ascertain the validity of the current step. This verification process is iterative, continuing along the current reasoning path until it becomes infeasible to proceed further. At this point, the LLM is prompted to complete the reasoning process by generating the remaining steps and providing the final answer. As depicted in Figure 3 (c), the process is represented as: $q \rightarrow s_1 \rightarrow \ldots \rightarrow s_{i-1} \rightarrow \times \rightarrow \hat{s}_i \rightarrow \ldots \rightarrow \hat{s}_n \rightarrow a$, denotes the s_{i-1} reasoning step is infeasible. s_i are the steps generated by the small model, and $\hat{s}_i$ are the steps generated (revised) by the LLM.

Based on the above statement, the Equation 5 can be formalized as the following process. We initiate the process with the question and prompt the LLM to determine if the current step s_i from the small model is valid for problem-solving. If the LLM's response is "Yes", we prompt it to generate its own reasoning step $\hat{s}_i$ as $\hat{s}_i \sim P(\hat{s}_i \mid q, s_i, \hat{s}_{<i})$. This process is continued until the LLM responds "No" to the i-th step. At this point, we prompt the LLM to generate the remaining reasoning steps and the final answer as $(a, \hat{s}_{\geq i}) \sim P(a, \hat{s}_{\geq i} \mid q, \hat{s}_{<i})$. The specifics of this verification and prompting process take the following form. Initially, we combine the first reasoning step from the student model s_1 with the question q to create the prompt: "For question <q>, can we start with this step: <s_1>?" If the LLM's answer is "Yes", we adopt the first reasoning step from the LLM $\hat{s}_1$ as the annotated step and proceed to the next reasoning step. For the i-th reasoning step from the student small model, we define the prompt as: "Can we continue with this step: <s_i>?" If the response is "No," we then prompt the LLM with: "What are the rest of the reasoning procedures and the answer?" to generate the subsequent reasoning steps and the final answer. An example is shown below:

Customized LLM-Annotated Explanation Revision

Prompt: For question <q>. Let's think step by step. Can we start with this step: <s_1>? If yes, give me your step. If no, give me the rest steps and the final answer.
Response: Yes, we can start with that <$\hat{s}_1$>
Prompt: Can we continue with this step <s_2>?
Response: Yes, the second step is <$\hat{s}_2$>.
Prompt: Can we continue with this step <s_3>?
Response: No, we should proceed as <$\hat{s}_{\geq 3}$>, the final answer is <$\hat{a}$>.

6 Experiments

Datasets The experiments are conducted on six well-known benchmark datasets across 3 different reasoning tasks: GSM8K (Cobbe et al., 2021) and AQuA (Geva et al., 2021) for arithmetic reasoning tasks, ANLI (Nie et al., 2019) and e-SNLI (Camburu et al., 2018) for natural language inference (NLI) tasks, and StrategyQA (Geva et al., 2021) and CommonSenseQA (Talmor et al., 2018) for

Method	Annotating	*Arithmetic*		*NLI*		*Commonsense*	
		GSM8K	AQuA	ANLI	e-SNLI	CommonSenseQA	StrategyQA
Teacher: `GPT-3.5-turbo`							
Zero-shot-CoT	–	73.45	54.96	68.02	67.67	68.94	69.78
Student: `LLaMA-2-7B`							
Zero-shot-CoT	–	10.04	21.07	33.94	28.98	41.28	44.71
Fine-Tuned Student							
Random	CoT Prompting	28.42	26.86	54.22	48.60	45.66	48.76
	CLAER	30.31	27.05	57.12	48.56	48.54	50.89
Maximum Entropy	CoT Prompting	27.58	27.67	52.56	47.98	46.35	49.03
	CLAER	29.04	27.42	53.75	51.76	48.86	51.05
Least Confidence	CoT Prompting	28.42	25.8	52.26	48.21	45.93	47.53
	CLAER	28.68	27.19	53.63	48.65	48.52	51.23
Disagreement	CoT Prompting	30.11	25.91	55.59	50.32	48.64	48.60
	CLAER	31.49	27.23	58.71	54.32	52.46	53.81
Self-Confidence	CoT Prompting	26.41	26.04	52.69	46.01	48.53	49.69
	CLAER	27.95	25.57	54.32	49.21	49.03	52.44
EGSS	CoT Prompting	30.01	26.91	55.87	51.16	49.64	50.32
	CLAER	32.72	28.43	58.02	54.44	53.53	55.63

Table 1: Performance of ELAD. Accuracy (%) of Fine-tuned LLaMA-2 model with ELAD (EGSS and CLEAR) and with different baseline sample selection strategies and completion generation methods. We report results at 50% annotation budget for all datasets in this table for comparison. Blue cells denote results of ELAD.

common sense reasoning task. Further details are provided in the Appendix A.

Evaluation Metric To assess question-answering performance for the above reasoning tasks, we calculate the *accuracy* based on the final answers given by the student small model.

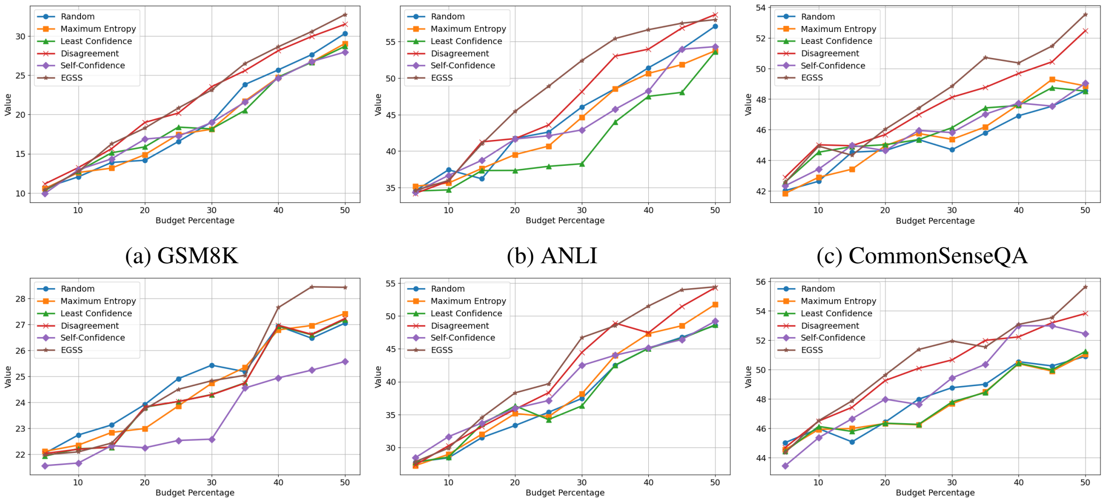
Setup We use `GPT-3.5-turbo` as a teacher via OpenAI API. We use `LLaMA-2-7B` (Touvron et al., 2023) as our student model. Further implementation details are provided in the Appendix A.

Baseline Methods We compare the proposed ELAD framework with two different categories of baselines: 1) sample selection methods, and 2) completion generation methods. To be more specific, we provide a comparison of *Explanation-Guided Sample Selection* (EGSS) with five different sample selection methods: Random, Maximum Entropy (Krishnakumar, 2007), Least Confidence (Culotta and McCallum, 2005), Disagreement (Vote Entropy) (Engelson and Dagan, 1996), and Self-Confidence (Kadavath et al., 2022). Further, we delineate the efficacy of our *Customized LLM-Annotated Explanation Revision* (CLAER) method in contrast to the conventional vanilla annotating with zero-shot-CoT prompting (Kojima et al., 2022) method applied post-sample selection. We also include comparative results from student and teacher models, assessed without fine-tuning and using direct prompting to answer questions.

6.1 Results and Analysis

This section evaluates the reasoning performance of models using our proposed ELAD framework, comparing it with baseline methods. We highlight improvements in sample selection and completion generation. Performance trends from a 5% to 50% annotating budget are depicted in Figure 4, illustrating the effectiveness of our EGSS method against other selection baselines. Furthermore, we detail reasoning performance at a 50% annotating budget for our CLAER completion annotation method and the CoT prompting baseline in Table 1.

Comparison with sample selection baselines From Table 1 we can observe that the EGSS method demonstrates significant performance improvements compared with traditional sample selection baselines for active learning. For arithmetic reasoning tasks (GSM8K and AQuA), EGSS with CLEAR exhibits a remarkable performance advantage. Specifically, it shows an increase of approximately 2.41% and 1.38% in accuracy over the next best-performing method for GSM8K and AQuA, respectively. In the context of natural language inference and commonsense reasoning tasks, such as ANLI, e-SNLI, CommonSenseQA, and StrategyQA, EGSS continues to set the benchmark. For



(a) GSM8K (b) ANLI (c) CommonSenseQA

(d) AQuA (e) e-SNLI (f) StrategyQA

Figure 4: Performance curves of different sample selection methods for active learning. The y-axis denotes the accuracy of the question-answering task, and the x-axis represents the percentage of samples annotated by the LLM for small model fine-tuning. In this case, 100% denotes that all samples from the training set have been annotated.

instance, in the ANLI dataset, EGSS achieves a performance boost of nearly 3.27% over the Least Confidence method with CLEAR. Similarly, for StrategyQA, EGSS demonstrates a substantial increase of 4.82% in accuracy compared to the Disagreement strategy. From Figure 4, It is evident that the proposed EGSS method effectively selects the most informative unlabeled questions, as evidenced by performance gains that align with increases in annotation budget. Initially, differences between EGSS and Disagreement strategies are minimal, likely due to the dominance of Inter-explanation uncertainty. However, as the annotation budget grows, EGSS significantly outperforms the Disagreement strategy, highlighting the crucial role of Intra-explanation uncertainty in identifying the most valuable samples for annotation.

Evaluating customized LLM-annotated explanation revision method Table 1 showcases the CLAER technique's effectiveness over the baseline CoT Prompting across several tasks. In arithmetic tasks like GSM8K and AQuA, CLAER outperforms vanilla CoT Prompting annotation method by up to 2.71% and 1.52%, respectively, under the EGSS framework, highlighting its superior capability in refining reasoning skills. In NLI and Commonsense Reasoning tasks, such as ANLI and StrategyQA, CLAER demonstrates notable accuracy improvements of 2.15% and 5.31%, respectively. These results underline the method's strength in leveraging detailed explanations to enhance model understanding and performance significantly.

6.2 Ablation Studies

In this section, we conduct an ablation study to investigate the importance of each component in the ELAD framework we propose, and the results are reported in Table 2. The results reveal that the full proposed ELAD framework outperforms configurations lacking EGSS (w/o EGSS) and CLAER (w/o CLAER) across all tasks. Ours demonstrates a notable performance advantage, with improvements up to 2.41% in arithmetic tasks, 6.88% in NLI tasks, and 5.09% in commonsense reasoning tasks over the "w/o EGSS" setup. This highlights the critical contributions of EGSS and CLAER to the framework's overall performance. The diminished performance in configurations without these components underscores their importance in enhancing model reasoning ability.

Setting	*Arithmetic*		*NLI*		*Commonsense*	
	GSM8K	AQuA	ANLI	e-SNLI	CommonSenseQA	StrategyQA
ELAD (Ours)	32.72	28.43	58.02	54.44	53.53	55.63
w/o EGSS	30.31	27.05	57.12	48.56	48.54	50.89
w/o CLAER	30.01	26.91	55.87	51.16	49.64	50.32

Table 2: Ablation Study. We report the performance of our ELAD framework under different settings.

7 Conclusion

This paper introduced the *Explanation-Guided LLMs Active Distillation* framework to address the challenges of deploying LLMs due to the high memory and computational demands. Our proposed framework provides an efficient distillation of LLMs, incorporating an active learning method. This includes an explanation-guided sample selection technique and a customized LLM-annotated explanation revision method tailored for active learning. Extensive experiments on various reasoning datasets demonstrate the effectiveness of our approach in enhancing the distillation efficiency.

Limitations

Our proposed ELAD framework, which utilizes LLMs as agents in active learning, is influenced by the design of prompts for the LLM, potentially affecting the quality of generated explanations and answers. Similarly, the prompt design for the small model can impact its reasoning abilities. Additionally, our approach requires submitting questions (data) to third-party services via APIs (e.g., OpenAI), posing a risk of data leakage. Additionally, due to budget constraints, we did not utilize the most recently released GPT-4.0 or other LLMs as the teacher model in our experiments. We plan to explore this in future research.

Ethical Considerations

All datasets and models used in this study are open-source, and references to previous work are properly cited. For fine-tuning the smaller student language model, we solely used triples generated by `GPT-3.5-Turbo` and `LLaMA-2-7B`, both of which are publicly accessible. This work complies with ethical guidelines, and no ethical concerns have been identified.

Acknowledgments

This work was supported by the NSF Grant No. 2432418, No. 2414115, No. 2007716, No. 2007976, No. 1942594, No. 1907805, No. 2318831, Cisco Faculty Research Award, Amazon Research Award.

References

Simran Arora, Avanika Narayan, Mayee F Chen, Laurel Orr, Neel Guha, Kush Bhatia, Ines Chami, Frederic Sala, and Christopher Ré. 2022. Ask me anything: A simple strategy for prompting language models. *arXiv preprint arXiv:2210.02441*.

Guangji Bai, Zheng Chai, Chen Ling, Shiyu Wang, Jiaying Lu, Nan Zhang, Tingwei Shi, Ziyang Yu, Mengdan Zhu, Yifei Zhang, et al. 2024. Beyond efficiency: A systematic survey of resource-efficient large language models. *arXiv preprint arXiv:2401.00625*.

Parikshit Bansal and Amit Sharma. 2023. Large language models as annotators: Enhancing generalization of nlp models at minimal cost. *arXiv preprint arXiv:2306.15766*.

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

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. *Advances in Neural Information Processing Systems*, 31.

Aaron Chan, Zhiyuan Zeng, Wyatt Lake, Brihi Joshi, Hanjie Chen, and Xiang Ren. 2022. Knife: Distilling reasoning knowledge from free-text rationales. *arXiv preprint arXiv:2212.09721*.

Jiuhai Chen and Jonas Mueller. 2023. Quantifying uncertainty in answers from any language model via intrinsic and extrinsic confidence assessment. *arXiv preprint arXiv:2308.16175*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Aron Culotta and Andrew McCallum. 2005. Reducing labeling effort for structured prediction tasks. In *AAAI*, volume 5, pages 746–751.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *arXiv preprint arXiv:2305.14314*.

Sean P Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. *arXiv preprint cmp-lg/9606030*.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2022. Large language models are reasoning teachers. *arXiv preprint arXiv:2212.10071*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. 2023. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *arXiv preprint arXiv:2305.02301*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling bert for natural language understanding. *arXiv preprint arXiv:1909.10351*.

Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, et al. 2022. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*.

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

Anita Krishnakumar. 2007. Active learning literature survey. *Tech. rep., Technical reports, University of California, Santa Cruz.*, 42.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155*.

David D Lewis. 1995. A sequential algorithm for training text classifiers: Corrigendum and additional data. In *Acm Sigir Forum*, volume 29, pages 13–19. ACM New York, NY, USA.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023. Symbolic chain-of-thought distillation: Small models can also" think" step-by-step. *arXiv preprint arXiv:2306.14050*.

Shiyang Li, Jianshu Chen, Yelong Shen, Zhiyu Chen, Xinlu Zhang, Zekun Li, Hong Wang, Jing Qian, Baolin Peng, Yi Mao, et al. 2022. Explanations from large language models make small reasoners better. *arXiv preprint arXiv:2210.06726*.

Jieyi Long. 2023. Large language model guided tree-of-thought. *arXiv preprint arXiv:2305.08291*.

Lucie Charlotte Magister, Jonathan Mallinson, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. Teaching small language models to reason. *arXiv preprint arXiv:2212.08410*.

Sharan Narang, Colin Raffel, Katherine Lee, Adam Roberts, Noah Fiedel, and Karishma Malkan. 2020. Wt5?! training text-to-text models to explain their predictions. *arXiv preprint arXiv:2004.14546*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2019. Adversarial nli: A new benchmark for natural language understanding. *arXiv preprint arXiv:1910.14599*.

Bo Pan, Zheng Zhang, Yifei Zhang, Yuntong Hu, and Liang Zhao. 2024a. Distilling large language models for text-attributed graph learning. *arXiv preprint arXiv:2402.12022*.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Menglin Xia, Xufang Luo, Jue Zhang, Qingwei Lin, Victor Rühle, Yuqing Yang, Chin-Yew Lin, et al. 2024b. Llmlingua-2: Data distillation for efficient and faithful task-agnostic prompt compression. *arXiv preprint arXiv:2403.12968*.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. *arXiv preprint arXiv:1906.02361*.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. 2021. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40.

Wendy D Roth and Jal D Mehta. 2002. The rashomon effect: Combining positivist and interpretivist approaches in the analysis of contested events. *Sociological methods & research*, 31(2):131–173.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7059–7073.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. 2019. Distilling task-specific knowledge from bert into simple neural networks. *arXiv preprint arXiv:1903.12136*.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, et al. 2022. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Peifeng Wang, Zhengyang Wang, Zheng Li, Yifan Gao, Bing Yin, and Xiang Ren. 2023. Scott: Self-consistent chain-of-thought distillation. *arXiv preprint arXiv:2305.01879*.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? gpt-3 can help. *arXiv preprint arXiv:2108.13487*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

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.

Sarah Wiegreffe, Ana Marasović, and Noah A Smith. 2020. Measuring association between labels and free-text rationales. *arXiv preprint arXiv:2010.12762*.

Bingsheng Yao, Ishan Jindal, Lucian Popa, Yannis Katsis, Sayan Ghosh, Lihong He, Yuxuan Lu, Shashank Srivastava, Yunyao Li, James Hendler, et al. 2023a. Beyond labels: Empowering human annotators with natural language explanations through a novel active-learning architecture. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 11629–11643.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023b. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Eric Sun, and Yue Zhang. 2023c. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *arXiv preprint arXiv:2312.02003*.

Mo Yu, Shiyu Chang, Yang Zhang, and Tommi S Jaakkola. 2019. Rethinking cooperative rationalization: Introspective extraction and complement control. *arXiv preprint arXiv:1910.13294*.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. 2022. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

Yifei Zhang, Siyi Gu, Bo Pan, Guangji Bai, Xiaofeng Yang, and Liang Zhao. 2023. Visual attention-prompted prediction and learning. *arXiv preprint arXiv:2310.08420*.

Zheng Zhang, Fan Yang, Ziyan Jiang, Zheng Chen, Zhengyang Zhao, Chengyuan Ma, Liang Zhao, and Yang Liu. 2024. Position-aware parameter efficient fine-tuning approach for reducing positional bias in llms. *arXiv preprint arXiv:2404.01430*.

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A survey of active learning for natural language processing. *arXiv preprint arXiv:2210.10109*.

Qilong Zhao, Yifei Zhang, Mengdan Zhu, Siyi Gu, Yuyang Gao, Xiaofeng Yang, and Liang Zhao. 2024. Due: Dynamic uncertainty-aware explanation supervision via 3d imputation. *arXiv preprint arXiv:2403.10831*.

A Appendix

Datasets Details

We provide more detailed descriptions of the datasets used in our experiments. We include a more detailed introduction and original sources released from the authors as follows:

GSM8K (Grade School Math 8K) (Cobbe et al., 2021): A dataset containing approximately 8,000 math word problems designed for grade school students, testing a variety of mathematical skills in natural language. For more information, visit the GSM8K GitHub repository.

AQuA (Algebra Question Answering) (Geva et al., 2021): Features algebraic word problems with multiple-choice answers, aimed at evaluating algebraic problem-solving in AI systems. Available on Kaggle.

ANLI (Adversarial Natural Language Inference) (Nie et al., 2019): A dataset with natural language inference tasks, including adversarial examples, to test models' understanding of human language beyond existing NLI datasets. For more details, refer to the ANLI GitHub repository.

e-SNLI (Explainable Stanford Natural Language Inference) (Camburu et al., 2018): Extends the SNLI dataset by providing human-annotated explanations for NLI decisions, assessing models on inference and explanation generation. Visit the e-SNLI GitHub repository for more information.

CommonSenseQA (Talmor et al., 2018): A question-answering dataset that focuses on commonsense reasoning, requiring an understanding of everyday concepts for correct answers. More details can be found on the CommonSenseQA website.

StrategyQA (Geva et al., 2021): Tests models on strategic question answering, particularly on reasoning about implicit strategies for yes/no questions. Information is available on the AllenAI website.

For each dataset where a validation set is not originally provided, we randomly subsample 10% of the original training set to serve as a validation set. The dataset statistics are provided in Table 3.

Dataset	Task Type	#Train	#Validation	#Test
GSM8K	Arithmetic	7,473	–	1,319
AQuA	Arithmetic	10,000	–	254
ANLI	NLI	16,946	1,000	1,000
e-SNLI	NLI	549,367	9,842	9,824
CommonSenseQA	Commonsense	9,741	975	1,221
StrategyQA	Commonsense	1,603	490	687

Table 3: Dataset statistics used in our experiments.

Implementation Details

The main experiments were conducted on a single NVIDIA GTX 3090 GPU with 24GB of memory. We utilize QLoRA4 (Dettmers et al., 2023) by default to conduct parameter-efficient fine-tuning. We set the attention dimension as 64, the alpha parameter as 16, and the dropout rate as 0.1. For fine-tuning the `Llama-2-7b-chat-hf` model, we utilized the Huggingface libraries[2] and set the batch size to 16 and the learning rate to 1e-4 using the AdamW optimizer.

Examples for Customized LLM-Annotated Explanation Revision

We present two examples of Customized LLM-Annotated Explanation Revision. Example 4 illustrates a case where the small model makes incorrect reasoning in step 3. The explanation from the small model is sequentially prompted to LLMs for verification. At step 3, the LLM responds with "No" and proceeds to revise step 3 and generate the remaining steps and the final answer. Example 5 provides an example where the small model errs in step 1 of its reasoning. The step 1 from the small model fails the LLM's verification, and LLM directly generates the entire completion.

Discussion of API Costs

The proposed CLAER method leverages LLM APIs, such as GPT-3.5 and GPT-4, to generate pseudo answers and explanations. The costs associated with these API calls are based on the number of input and output tokens, with output tokens priced significantly higher than input tokens—approximately three times higher[3]. Compared to the zero-shot CoT prompting method, which generates both an answer and an explanation in a single prompt, our CLAER method does not substantially increase the output token counts. In fact, it can even result in fewer tokens if the initial explanation steps input is accurate and succinct. Also, our designed prompts are not long, which does not substantially increase the input token counts (Pan et al., 2024b; Zhang et al., 2024). Therefore, the main additional cost arises from input tokens, which are considerably less expensive than output tokens. Our analysis, based on comparing the average input and output token counts between CLAER and zero-shot CoT prompting, suggests that the incremental costs are minor relative to the benefits of significantly reducing the need to annotate large numbers of unlabeled samples through an efficient sampling strategy.

[2]https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
[3]https://openai.com/api/pricing/

Question: Anne purchased a box of sweets that contains 15 packs, and each pack has 60 pieces. She kept two packs and gave the rest to her 10 friends equally. How many pieces of sweets did each of her friends receive? **Response from Small Model:** Let's think step by step. Step 1: Identify the information given in the problem. Number of packs of sweets Anne purchased: 15 Number of pieces in each pack: 60 Number of friends Anne gave the sweets to: 10. Step 2: Identify the unknown quantity. Number of pieces of sweets each friend received: x (where x is the unknown quantity). Step 3: Use the information given in the problem to find the unknown quantity. Since Anne kept two packs for herself, the total number of pieces she gave to her friends is 15 x 10 = 150. Therefore, the number of pieces each friend received is 150 / 10 = 15 pieces. So, each of Anne's 10 friends received 15 pieces of sweets from her.
Prompt: Anne purchased a box of sweets that contains 15 packs, and each pack has 60 pieces. She kept two packs and gave the rest to her 10 friends equally. How many pieces of sweets did each of her friends receive? Let's think step by step. Can we start with this step: Step 1: Identify the information given in the problem. Number of packs of sweets Anne purchased: 15 Number of pieces in each pack: 60 Number of friends Anne gave the sweets to: 10? Yes or no? If yes, give me your answer for this step. If no, give me the rest steps and final answer.
Response: ✓ Yes, we can start with that step. Step 1: Identify the information given in the problem. Number of packs of sweets Anne purchased: 15. Number of pieces in each pack: 60. Number of friends Anne gave the sweets to: 10.
Prompt: Can we continue with this step Step 2: Identify the unknown quantity. Number of pieces of sweets each friend received: x (where x is the unknown quantity)?
Response: ✓ Yes, we can continue with Step 2: Step 2: Identify the unknown quantity. Number of pieces of sweets each friend received: x (where x is the unknown quantity).
Prompt: Can we continue with Step 3: Use the information given in the problem to find the unknown quantity. Since Anne kept two packs for herself, the total number of pieces she gave to her friends is 15 x 10 = 150. Therefore, the number of pieces each friend received is 150 / 10 = 15 pieces.?
Response: × No, we should proceed as Step 3: Use the information given in the problem to find the unknown quantity. Since Anne kept two packs for herself, the total number of pieces she gave to her friends is (15 packs - 2 packs) x 60 pieces/pack = 13 packs x 60 pieces/pack = 780 pieces. Therefore, the number of pieces each friend received is 780 pieces / 10 friends = 78 pieces. So, each of Anne's friends received 78 pieces of sweets.

Table 4: Example that small model makes incorrect reasoning in its step 3

Question: Rong has been saving 20 coins in his piggy bank every month. Neil has been saving 2/5 times more coins in his piggy bank per month than Rong. How many coins are they having ten years after they started their savings? **Response from Small Model:** Rong has been saving 20 coins in his piggy bank every month. Neil has been saving 2/5 times more coins in his piggy bank per month than Rong. How many coins are they having ten years after they started their savings? Let's think step by step. Rong has been saving 20 coins per month for 10 years, so he has saved a total of 20 x 10 = 200 coins. Neil has been saving 2/5 times more coins than Rong, so he has saved 2/5 x 20 = 40 coins per month. He has saved a total of 40 x 10 = 400 coins. Therefore, Rong has 200 coins and Neil has 400 coins after 10 years of saving."
Prompt: Rong has been saving 20 coins in his piggy bank every month. Neil has been saving 2/5 times more coins in his piggy bank per month than Rong. How many coins are they having ten years after they started their savings? Let's think step by step. Can we start with this step: Rong has been saving 20 coins per month for 10 years, so he has saved a total of 20 x 10 = 200 coins? If yes, give me your answer on this step. If no, give me the rest steps and final answer.
Response: × No, we cannot start with that step because Neil has been saving a different amount of coins per month than Rong. We need to calculate Neil's savings separately. Let's break it down step by step: Step 1: Calculate how many coins Rong has saved in 10 years. Rong saves 20 coins per month, so in a year, he saves 20 coins x 12 months = 240 coins. In 10 years, Rong will have saved 240 coins/year x 10 years = 2,400 coins. Step 2: Calculate how many coins Neil has saved in 10 years. Neil has been saving 2/5 times more coins than Rong per month. 2/5 of 20 coins = (2/5) * 20 = 8 coins. So, Neil saves an extra 8 coins per month compared to Rong. Neil's monthly savings = 20 coins (Rong's savings) + 8 coins (extra savings) = 28 coins per month. In a year, Neil saves 28 coins x 12 months = 336 coins. In 10 years, Neil will have saved 336 coins/year x 10 years = 3,360 coins. Step 3: Calculate the total number of coins they both have together. Rong saved 2,400 coins, and Neil saved 3,360 coins. So, the total number of coins they have saved together is 2,400 coins + 3,360 coins = 5,760 coins. So, after 10 years of saving, they will have a total of 5,760 coins.

Table 5: Example that small model makes incorrect reasoning in its step 1