Evolving Knowledge Distillation with Large Language Models and Active Learning

Chengyuan Liu[1,2,†] , Yangyang Kang[2] , Fubang Zhao[2] , Kun Kuang[1,*] , Zhuoren Jiang[3,*] , Changlong Sun[2] , Fei Wu[1,4]
liucy1@zju.edu.cn, {yangyang.kangyy, fubang.zfb}@alibaba-inc.com, kunkuang@zju.edu.cn
jiangzhuoren@zju.edu.cn, changlong.scl@taobao.com, wufei@zju.edu.cn
[1]College of Computer Science and Technology, Zhejiang University
[2]Institute for Intelligent Computing, Alibaba Group
[3]School of Public Affairs, Zhejiang University
[4]Shanghai Institute for Advanced Study of Zhejiang University

Abstract

Large language models (LLMs) have demonstrated remarkable capabilities across various NLP tasks. However, their computational costs are prohibitively high. To address this issue, previous research has attempted to distill the knowledge of LLMs into smaller models by generating annotated data. Nonetheless, these works have mainly focused on the direct use of LLMs for text generation and labeling, without fully exploring their potential to comprehend the target task and acquire valuable knowledge. In this paper, we propose EvoKD: Evolving Knowledge Distillation, which leverages the concept of active learning to interactively enhance the process of data generation using large language models, simultaneously improving the task capabilities of small domain model (student model). Different from previous work, we actively analyze the student model's weaknesses, and then synthesize labeled samples based on the analysis. In addition, we provide iterative feedback to the LLMs regarding the student model's performance to continuously construct diversified and challenging samples. Experiments and analysis on different NLP tasks, namely, text classification and named entity recognition show the effectiveness of EvoKD.

Keywords: LLM, Active Learning, Few-Shot Learning

1. Introduction

Although large language models (LLMs) achieve considerable performance with limited task-specific annotated data (Zhang et al., 2023; Dathathri et al., 2020; Brown et al., 2020; Touvron et al., 2023a,c; Yang et al., 2023), they suffer from the disadvantages of high cost and low speed during inference. Besides, the models of some professional systems are required to perform on a high level for the practical applications, such as coding, math, poem writing, rather than solving diverse tasks. Thus, it is very important to study on efficiently teaching a cheap and small model to learn the professionality of the LLMs (Ho et al., 2023; Wu et al., 2023b). We consider Knowledge Distillation (KD) as a feasible technique.

Hinton et al. (2015) firstly distilled the specific knowledge in an ensemble of models into a single model. Traditional KD methods require to train a teacher model with high-quality annotated data. Facilitated with LLMs, the cost of KD naturally decreases when adopting LLMs as teacher models.

Gu et al. (2023) summarized the two commonly applied categories of KD: *black-box KD*, where only the teacher predictions are accessible, and *white-box KD* (Gou et al., 2021), where the teacher parameters are available to use. Recently, *black-box KD* has shown promising results in fine-tuning small models on the prompt-response pairs generated by LLM APIs (Taori et al., 2023; Peng et al., 2023; Wu et al., 2023a). Comparing with *white-box KD*, *black-box KD*: (1) is less restrictive in terms of structural requirements for teacher models and student model, (2) uses stronger teacher model (such as ChatGPT API), (3) doesn't require the private deployment of teacher model. In *black-box KD*, data serves as the carrier of knowledge since the parameters of teacher model is not available. Ye et al. (2022) firstly introduced ZeroGen from the perspective of data-free model-agnostic knowledge distillation in the manner of data generation (DG). Based on DG, Gao et al. (2023) proposed the noise-robust re-weighting framework. With the superiority of ChatGPT[1], Dai et al. (2023) adopted ChatGPT to do Data Augmentation(DA) for text classification, called AugGPT. Figure 1 illustrates the relationship between DG, DA and KD. The difference between DA and DG is the initial amount of supervised data, which is negligible under 1-shot setting. In this paper, we adopt the concept of "Knowledge Distillation" (KD), because we utilize the LLM not only as a data generator but also to distill knowledge about learning the task, understanding the input text, labeling and evaluating the predictions made

[†]This work was done when Chengyuan Liu interned at Alibaba.

*Corresponding author.

[1]https://chat.openai.com/

LREC-COLING 2024, pages 6717–6731
20-25 May, 2024.

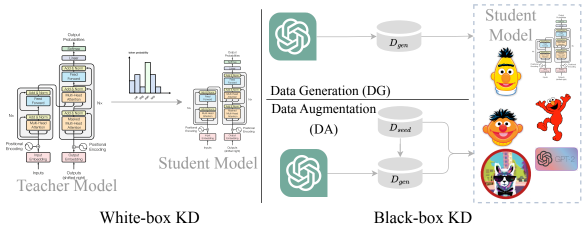

Figure 1: Relationship between KD, DA and DG.

by the student model. Currently, the best LLMs are black-box models(such as GPT-4). Considering the cost and efficiency, we adopt *black-box KD* as our distillation approach.

There are two notable limitations in prior works when conducting the process of *black-box KD*: **Under-utilization**, previous studies have regarded LLMs as mere text generators and sentence rewriters, solely relying on their capabilities for text generation and labeling. However, they have neglected the knowledge embedded in the downstream task and the powerful comprehensive ability of LLMs, which may resulting in a hindrance to the quality of the generated text. **Inflexibility**, prior KD studies have primarily been conducted in an offline and static manner. They construct the entire training data in one go, without considering any dynamic changes that may arise in the status and weaknesses of the student model. Consequently, the generated data often lacks specificity and diversity, limiting its effectiveness in improving the performance of the student model.

In this paper, we propose **EvoKD**: **Evo**lving **K**nowledge **D**istillation with Large Language Models and Active Learning, in order to address the above limitations. The objective of Active Learning (Cohn et al., 1996; Olsson, 2009; Gentile et al., 2022) is to optimize the effectiveness of model training by prioritizing the annotation of the most valuable samples. In line with this, Evolving Knowledge Distillation with Active Learning aims to distil the most informative knowledge that effectively compensate for the weakness of the student model. Moreover, a dynamic teaching strategy is adopted, where the generation of the samples is based on the status of the student model. This dynamic strategy stands in contrast to static strategies that disregard changes in variables. The name draws inspiration from biology, where evolution refers to the change in the characteristics of a species over generations. Changing environmental conditions lead to evolutionary shifts in populations. Similarly, the idea behind Evolving Knowledge Distillation emphasizes the changes of teaching strategy according to the feedback from student, treating it as a dynamic environment over several iterations.

Different from Moon and Carbonell (2019); Diao et al. (2023), EvoKD employs LLMs to adapt strategies for generating valuable samples incorporating the concept of Evolving Knowledge Distillation. This approach can mitigating the potential impact of human annotators' diverse preferences, and providing a stable, cost-effective, and flexible framework.

Specifically, EvoKD uses the student model's past performance on selected samples as inputs and prompts the LLM to identify weakness of the student model, based on which, the LLM generates a batch of new sentences, consisting of both challenging and easy samples, along with their corresponding labels. The student model is evaluated and subsequently trained on the batch data. The evaluation output provides iterative feedback to the LLM. This batch distillation process is repetitive. The LLM is effectively explored during Weakness Analysis, which demands a meticulous investigation of the data distribution.

We performed experiments on five text classification tasks and two NER tasks mainly under 1-shot settings to evaluate the effectiveness of EvoKD. The experiment results demonstrate that EvoKD significantly outperformed the baseline methods. Notably, on text classification datasets, **EvoKD achieved up to 90% of the full-shot performance with only 1-shot**. We will release our code for further studies.

Our contributions are three folds, which can be summarized as follows:

1. We introduce the concept of Evolving Knowledge Distillation, which uses dynamically teaching strategy to distill the knowledge about learning the task, understanding the input texts, labeling and evaluating the predictions of student model.

2. A novel approach called EvoKD is proposed in this paper incorporating the evolving KD and Active Learning, which leverages LLM's potential to comprehend the target task and acquire valuable knowledge.

3. Experiments on text classification and NER tasks are conducted under few-shot settings comparing EvoKD with other baselines. EvoKD significantly outperformed all baseline approaches. Notably, EvoKD **achieved up to 90% of the full-shot text classification performance with only 1-shot**.

2. Related Work

2.1. Large Language Model

Large Language Models (LLMs) have recently demonstrated remarkable superiority in multiple traditional NLP tasks. Brown et al. (2020) proposed GPT-3, increasing the size of model parameters to

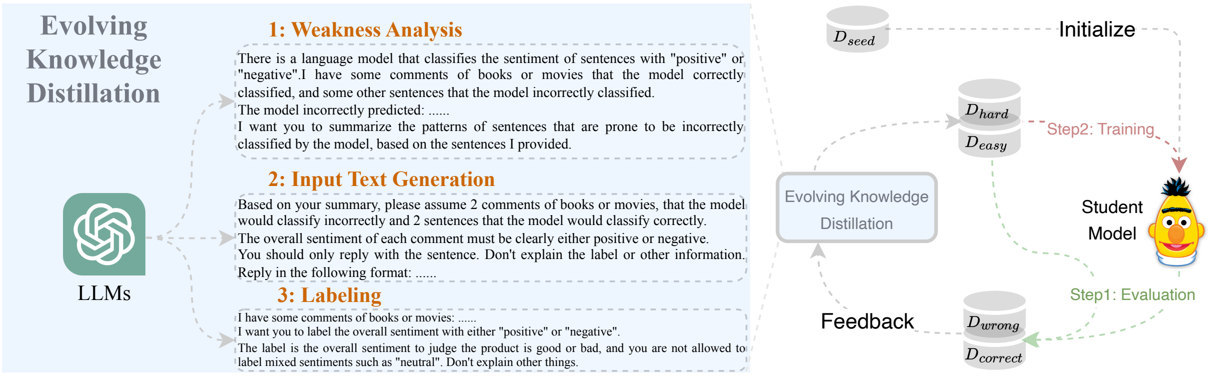

Figure 2: Framework of EvoKD. The initial student model is trained using the few-shot training data. Then, both correct and wrong samples are identified via "Evaluation" step. Iteratively, the identification results are used to distill the new samples. For Evolving Knowledge Distillation, the process begins by prompting the LLM to analyse the weakness of the student model, given the correct and wrong samples. Based on the weakness, the LLM is required to generate a set of challenging and easy samples, which are collected to construct a batch data. The batch data firstly evaluates the student model to obtain the next feedback, then the student model is trained on the batch data.

175B. Chowdhery et al. (2022) trained a 540B parameter, densely activated Transformer language model, named as PaLM. PaLM achieved SOTA few-shot learning results on hundreds of NLU and NLG taks. Chung et al. (2022) released Flan-T5, they found that instruction fine-tuning with the scaling of the number of tasks, the model size and fine-tuning on chain-of-thought data dramatically improves performance.

2.2. Black-Box KD

Few-shot learning is considered as a more practical setting than full-shot, although some methods are developed to improve the performance of models (Wang et al., 2017). Data augmentation is a widely employed approach to improve the performance by expanding the size of the training dataset and increasing the diversity of samples. Common data augmentation approaches for NLP tasks include deleting, inserting random characters, and substituting words with synonyms. However, these methods have obvious limitations such as reduced text fluency and limited word diversity.

Model-based methods are more efficient currently. Edwards et al. (2023) used GPT-2 to generate artificial training instances with domain expert selection in order to improve classification performance. Wei et al. (2021) explored a technique particularly suitable for few-shot, highly-multiclass text classification setting. To further boost performance, they also presented a simple training strategy called curriculum data augmentation, which leverages curriculum learning by first training on only original examples and then introducing augmented data as training progresses. Dai et al. (2023) proposed a text data augmentation approach based on ChatGPT, named AugGPT. AugGPT rephrases each sentence in the training samples into multiple conceptually similar but semantically different samples.

Data Generation is slightly different with DA, as it has no initial seed data. ZeroGen (Ye et al., 2022) is a flexible and efficient zero-short learning method, also provides insights from the perspective of data-free model-agnostic knowledge distillation. Gao et al. (2023) proposed a novel noise-robust re-weighting framework SunGen to automatically construct high-quality data for zero-shot classification problems. Ubani et al. (2023) investigated the use of data obtained from prompting a large generative language model, to generate synthetic training data for few-shot learning. Tang et al. (2023) proposed to generate a vast quantity of high-quality synthetic data with labels utilizing ChatGPT and fine-tuning a local model for the downstream task. They prompted ChatGPT to extract structured information from unstructured healthcare texts, with a focus on biological named entity recognition and relation extraction.

2.3. Active Learning

Active Learning involves reducing the amount of labeled data needed to learn a target concept by strategically querying the annotator for labels of the most informative examples (Yuan et al., 2020; Angluin, 1988; Sener and Savarese, 2017; Settles, 2009). Diao et al. (2023) proposed Active-Prompt, to adapt LLMs to different tasks with task-specific example prompts which are annotated with human-designed CoT reasoning, and they determined which questions are the most important and helpful ones to annotate from a pool of task-specific queries. Wang et al. (2021) proposed an active labeling strategy to have humans re-annotate data labeled by GPT-3 (Brown et al., 2020) with the lowest confidence scores, to reduce the noise in the labeled data from GPT-3. There are also other approaches to select the instances to be labeled (Schumann and Rehbein, 2019; Ren et al., 2021).

Standard Active Learning operates by utilizing a pool of unlabeled data, from which annotators select the most informative samples for annotation, thereby reducing labeling costs. However, our experimental setup differs significantly. Our primary focus revolves around the task of knowledge distillation, wherein we compare our approach against other baselines about KD. Unlike traditional Active Learning scenarios, our framework does not involve human annotators or rely on an unlabeled data pool. Instead, we only draw inspiration from the core motivation of Active Learning, which involves identifying the most informative instances. In our framework, these instances are generated by LLM, further distinguishing it from standard Active Learning tasks.

3. EvoKD

In this Section, we introduce the framework of EvoKD, as shown in Figure 2.

In few-shot learning, only a limited amount of training data is available initially. We begin with m training samples and use them to train a student model, denoted as $model^0$. Some samples are predicted incorrectly by $model^0$ and are denoted as D^0_{wrong}, while others are predicted correctly and are denoted as $D^0_{correct}$.

We use the LLM to perform knowledge distillation on the incorrectly predicted or classified samples to investigate the student model's weaknesses on specific tasks, which is an active approach. As the parameters of the student model changes, its performance and weakness both change accordingly.

EvoKD performs knowledge distillation iteratively. For the i-th iteration, D^{i-1}_{wrong}, $D^{i-1}_{correct}$ and $model^{i-1}$ are given, where D^{i-1}_{wrong} is the subset of samples where $model^{i-1}$ has the worst performance, and $D^{i-1}_{correct}$ is the subset of samples with the best performance. The online LLM is fed the student model's performance and it is asked to generate several samples in a conversational manner. Formally, the LLM is asked to propose $\lfloor \frac{b}{2} \rfloor$ easy samples, denoted as D^i_{easy}, and $\lceil \frac{b}{2} \rceil$ hard samples, denoted as D^i_{hard}, given D^{i-1}_{wrong} and $D^{i-1}_{correct}$, where b is the batch size. Then D^i_{easy} and D^i_{hard} are concatenated to construct the i-th batch D^i, with which, $model^{i-1}$ is updated to $model^i$. Additionally, the current student model is evaluated with D^i. The teacher LLM is then instructed to generate the new batch data D^{i+1} given the status of current model performance.

There are reasons for the inclusions of $D_{correct}$ in the chatting inputs and D_{easy} in the chatting outputs respectively:

- **Why include $D_{correct}$ to construct inputs?** The correctly predicted samples help the LLM analyse the student model's weaknesses. Including the correctly predicted samples makes the weakness more apparent than using only the incorrectly predicted samples.
- **Why use D_{easy} to train?** If we only use the challenging samples as training data, the sample distribution learned will be biased, leading to the problem of catastrophic forgetting. The student model may perform increasingly poorly on originally easy samples. Therefore, we enable LLM to generate both D_{easy} and D_{hard} based on weakness, thereby preventing the base model from forgetting previous knowledge and falling into local optimum caused by biased distributions.

In fact, the above operations indeed improve performance, as demonstrated by the ablation study (in SubSection 4.9).

The batch D^i is used in two ways: 1) The previous checkpoint $model^{i-1}$ is evaluated on D^i , and the real incorrect samples D^i_{wrong} and correct samples $D^i_{correct}$ are identified based on the metric. The performance is then described in the prompt, which is fed to the LLM in the next iteration. 2) D^i also serves as the knowledge carrier to update the student model from $model^{i-1}$ to $model^i$. The model checkpoint $model^n$ after n iterations is the final objective.

3.1. Evolving Knowledge Distillation with LLM

Evolving Knowledge Distillation with LLM aims to dynamically provide the most informative knowledge for the student model. In EvoKD, the LLM is utilized to adapt teaching strategies to generate the beneficial samples. To enhance the results of Evolving Knowledge Distillation, we have subdivided the batch generation process into three sub-steps:

1. **Weakness Analysis.** The LLM analyses the weakness of the student model by identifying the pattern of the sentences that are likely to be incorrectly predicted by the student model. The pattern string is used both in generating new samples and explaining the generated samples. If required, EvoKD also enables human intervention to adjust the pattern string, thus influencing subsequent generations.

Algorithm 1 EvoKD

Require: $D^0, num\_steps, chat, review$

Initialize $model^0$
$D^0_{wrong}, D^0_{correct} \leftarrow \mathbf{Identify}(model^0, D^0)$
$model^0 \leftarrow$ train $model^0$ on D^0
$i \leftarrow 0$
$step \leftarrow 0$
$history \leftarrow \{D^0\}$
while $step < num\_steps$ **do**
  $step \leftarrow step + 1$
  if $step \; \% \; review = 0$ **then**
    train $model^i$ on $history$
    update $step$
  else if $step$ % $chat = 0$ **then**
    $i \leftarrow i + 1$
    $D^i_{hard}, D^i_{easy} \leftarrow \mathbf{LLM}(D^{i-1}_{wrong}, D^{i-1}_{correct})$
    $D^i \leftarrow \{D^i_{hard}, D^i_{easy}\}$
    Add D^i to $history$
    $D^i_{wrong}, D^i_{correct} \leftarrow \mathbf{Identify}(model^{i-1}, D^i)$
    $model^i \leftarrow$ train $model^{i-1}$ on D^i
  else
    train $model^i$ on D^i
  end if
end while
return $model^i$

2. **Input Text Generation.** The LLM is prompted to create input texts according to the identified weakness. LLMs exhibit advanced reasoning capabilities, enabling them to generate text based on attributes inferred from weakness.

3. **Labeling.** The LLM is prompted to label each generated sentence in a new conversation. We separate the text generation and labeling processes to mitigate the risk of intentional mislabeling by the LLM. As in the conversation of Input Text Generation, we requested the LLM to create challenging samples, if the LLM simultaneously generates text and labels, it may deliberately mislabel the samples to induce the student model to make mistakes. We conducted ablation study to prove the effectiveness of separating text generation and labeling, which is shown in SubSection 4.9.

3.2. Strategies to Improve Effectiveness

We suggest two strategies to reuse the samples generated by the LLM.

Repeat Batch We train the student model on the same batch for several steps. Because training on a batch of samples for a single step has little impact to the parameters and performance of the student model, and its prediction will likely be similar to the previous iteration.

Review History We store all the generated samples in a global cache. Then, at regular intervals, batches of samples are fetched from the cache to train the model. We find that training on the historical samples enables the model to recall previous knowledge and avoid making mistakes on the same patterns.

The pseudocode of EvoKD is shown in Algorithm 1.

3.3. Initialization

If we execute EvoKD with a randomly initialized student model, the initial pattern may be arbitrary and unnecessary to analyze, or it could potentially hinder the student model's ability to comprehend texts with diverse attributes. To address this issue, we introduce an optional strategy: AugGPT is incorporated in the first few epochs, before running EvoKD in the subsequent epochs.

4. Experiments

Task	Dataset	Language	Label Num	Train Num	Test Num
Classification	Amazon	English	2	50000	50000
	IMDB	English	2	25000	25000
	Inshorts	English	5	2999	407
	Toutiao News	Chinese	14	10000	950
	CAIL2019 divorce	Chinese	3	9876	1200
NER	CoNLL03	English	3	14041	3453
	CoNLL04	English	3	922	288

Table 1: Details of the datasets.

In this Section, we conduct experiments on 5 text classification datasets under 1-shot setting (SubSection 4.4), and investigate the performance when the size of training data increases (SubSection 4.7). We also explore the effectiveness of EvoKD on NER tasks (SubSection 4.8). Finally, we conduct the ablation study (SubSection 4.9).

4.1. Implementation Details

We utilize gpt-3.5-turbo-0301 to implement the conversation with the LLMs, and report the mean F1 value and standard deviation with random seeds ranging from 1 to 5 for each setting. We adopt BERT-base as the student model for English datasets and Chinese-BERT-base for Chinese datasets. We set the learning rate to 2e-5 and batch size to 8. We set the clip gradient norm to 2. For few-shot experiments, we train the student model with 10 epochs, and the total number of steps in each epoch is set to 1250. As we have no available validation set in the practical few-shot scenario, we directly test the final checkpoint of

Method	English			Chinese		AVG
	Amazon	IMDB	Inshorts	TouTiao	CAIL2019	
Full Shot	0.9480	0.9495	0.9705	0.8495	0.9683	0.9372
No Augment	0.6030 $\pm$ 0.0880	0.5833 $\pm$ 0.0853	0.6408 $\pm$ 0.1528	0.3638 $\pm$ 0.0737	0.4422 $\pm$ 0.0604	0.5266
EDA	0.6314 $\pm$ 0.0838	0.6189 $\pm$ 0.0599	0.6776 $\pm$ 0.1669	0.3848 $\pm$ 0.0843	0.6120 $\pm$ 0.0319	0.5849
ZeroGen	0.7054 $\pm$ 0.1134	0.5087 $\pm$ 0.2204	0.8334 $\pm$ 0.0429	0.6442 $\pm$ 0.0369	0.7620 $\pm$ 0.1118	0.6907
SunGen	0.6257 $\pm$ 0.1288	0.5769 $\pm$ 0.0521	0.8103 $\pm$ 0.0456	0.2533 $\pm$ 0.0827	0.8305 $\pm$ 0.1020	0.6193
Gradual	0.5826 $\pm$ 0.0771	0.6857 $\pm$ 0.0109	0.7608 $\pm$ 0.0144	-	-	-
AugGPT	0.6234 $\pm$ 0.1712	0.6903 $\pm$ 0.0788	0.7902$\pm$ 0.0759	0.6514 $\pm$ 0.0459	0.7122 $\pm$ 0.1117	0.6935
EvoKD	**0.8425** $\pm$ **0.0317**	<u>0.7982 $\pm$ 0.0565</u>	<u>0.8516 $\pm$ 0.0257</u>	<u>0.6874 $\pm$ 0.0199</u>	**0.9148** $\pm$ **0.0411**	<u>0.8189</u>
+Init	<u>0.8403 $\pm$ 0.0240</u>	**0.8359** $\pm$ **0.0272**	**0.8688** $\pm$ **0.0167**	**0.7112** $\pm$ **0.0237**	<u>0.9137 $\pm$ 0.0355</u>	**0.8340**

Table 2: Experiment results under 1-shot text classification. We use "No Augment" to denote the 1-shot performance without knowledge distillation, and "+Init" means that in the initial epochs, using AugGPT to initialize the student model. The best results are highlighted in bold, and the second best results are underlined. "AVG" denotes the average performance over all datasets.

the student model. For few-shot sampling from full training data, we follow (Dai et al., 2023), Lu et al. (2022) and Lou et al. (2023). To train EvoKD, we set $chat$ in Algorithm 1 to 40, and $review$ to 50. And the threshold to identify the correct and wrong cases is set to 0.95.

4.2. Datasets

Our experiments encompass two tasks: Text Classification and Named Entity Recognition (NER). For Text Classification, we use Amazon (Keung et al., 2020), IMDB (Maas et al., 2011) and Inshorts-News[2], TouTiao-News[3] and CAIL2019[4]. Amazon and IMDB are datasets for product reviews, which express either a positive or negative sentiment. Inshorts and TouTiao are about news categories classification. We remove the ambiguous news categories. The track we choose from CAIL2019 involves identifying attributes from sentences relative to divorce events. We only keep the categories "have children after marriage", "joint debts" and "joint property" in order to clearly distinguish between the attributes. For NER, we use CoNLL03 (Tjong Kim Sang and De Meulder, 2003) and CoNLL04 (Carreras and Màrquez, 2004). We remove the entity types "other" and "miscellaneous" as they have no specific meanings.

We list the details of the datasets in Table 1.

4.3. Baselines

As we discussed in Section 1, both of DG and DA can be applied in *black-box KD*. Several previous studies of DG and DA are included as baselines:

EDA We adopt several rule-based approaches for Easy Data Augmentation(EDA), including 1) Swap Char (Belinkov and Bisk, 2018), 2) Delete Char, 3) Insert Char, 4) Replace Word (Ma, 2019), 5) Insert Word. We employ nlpaug (Ma, 2019) for English and nlpcda[5] for Chinese to implement the rule-based methods. The baselines "Swap Word" and "Replace Word" both use synonym words, which are generated by stronger, model-based methods. For details, we adopt bert-base-uncased from HuggingFace as the model for synonyms. We set $change\_rate$ as 0.3 in nlpcda, and the others are default. The highest F1 score among above methods is reported.

AugGPT Dai et al. (2023) introduced AugGPT for few-shot learning to solve text classification. As the code and prompt is publicly unavailable at the time of writing, we write customized prompt for each dataset.

ZeroGen ZeroGen (Ye et al., 2022) provides insights from the perspective of data-free model-agnostic knowledge distillation. For fairness, we set the same augmented data size for all methods under 1-shot, and adopt gpt-3.5-turbo-0301 to do augmentation with seed samples as context.

SunGen Gao et al. (2023) proposed a novel noise-robust re-weighting framework SunGen. SunGen shares the same data setting with ZeroGen.

Gradual Curriculum Wei et al. (2021) explored a technique particularly suitable for few-shot, highly-multiclass text classification setting. As the official implementation depends on EDA package which doesn't supports Chinese, so we only report the results on English datasets. This baseline is denoted as **Gradual** in the following experiments.

[2]https://github.com/kishanpython/Inshorts-News-Data-Collection

[3]https://github.com/BenDerPan/toutiao-text-classfication-dataset

[4]https://github.com/china-ai-law-challenge/CAIL2019/

[5]https://github.com/425776024/nlpcdaf

4.4. 1-Shot Classification

We can observe from Table 2 that, model-based methods exhibits notable superiority over EDA. SunGen has a clear disadvantage in the news classification of TouTiao. We observed its sample weights and found that SunGen gives a relatively large weight to the $story$ and $travel$ categories, while the weights for other categories are small. Through analysis, we discovered that due to the low noise in the samples generated by GPT-3.5, SunGen's advantage cannot be fully utilized. Under this setting, when there are multiple label categories, SunGen's performance can be unstable and unsatisfactory.

Our proposed method, EvoKD, exhibits **better performance** than all baselines on all classification datasets. Compared with AugGPT, EvoKD achieved the highest absolute improvement in Amazon sentiment classification at 21.91% and the lowest in TouTiao news classification at 3.60%. It is worth mentioning that EvoKD demonstrates a **higher stability** compared with AugGPT. For instance, in the case of Amazon, EvoKD reduces the deviation from 0.1712 to 0.0317, resulting in a relative reduction of 81.48%. The stability of EvoKD could be attributed to its active analysis of the student model, which effectively mitigating the impact caused by the distribution randomness of the seed samples.

"EvoKD + Init" involves distilling the training data in the first few epochs using AugGPT for initialization, then training the student model with EvoKD in the subsequent epochs. The results are shown in the last row of Table 2. "EvoKD + Init" **achieved as high as 90% of the full-shot performance with only 1-shot** on most of the text classification datasets, such as CAIL2019 and Inshorts news classification. In the case of TouTiao news classification, which involves 14 categories, EvoKD with initialization achieved 84% of the full-shot F1 under the 1-shot setting. These results highlight EvoKD's ability to effectively employ LLMs for actively analyzing and generating difficult samples in 1-shot scenarios. By harnessing the capabilities and knowledge of LLMs, EvoKD showcases its effectiveness in knowledge distillation.

We also notice that on the legal dataset, CAIL2019, EvoKD outperforms the baselines by at least 8 percents, which indicates the effectiveness of EvoKD under professional domains.

4.5. Efficiency of EvoKD

Figure 3 illustrates the performance versus the number of tokens used through OpenAI API, comparing EvoKD with AugGPT. The dataset is TouTiao news classification, under 1-shot.

It can be concluded that from the trending, our method has more advantages from a long-term perspective. In a single interaction, our method will consume more tokens than AugGPT because analyzing weaknesses and task explanations both require a significant number of tokens. However, as the number of interactions increases, the benefits brought by our method far outweigh the disadvantage of token consumption. After 10,000 tokens of interaction, the growth of AugGPT becomes very slow, while EvoKD is still growing. In conclusion, AugGPT has a little superiority over EvoKD at the begining, while EvoKD ties AugGPT then and significantly outperforms AugGPT after about 8000-tokens interaction.

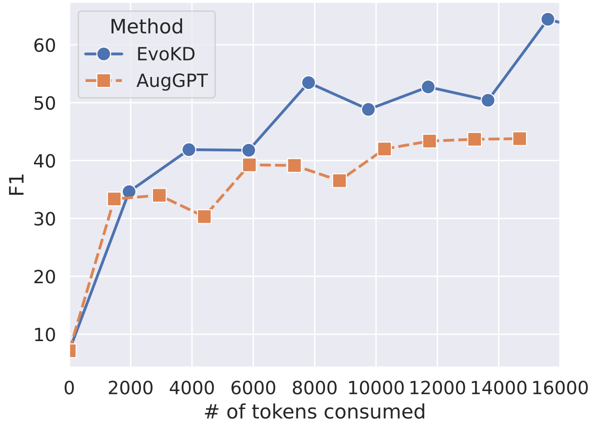

Figure 3: F1 versus the number of tokens used during training.

4.6. Adaptability Analysis

We believe that the outstanding performance may come from the adaptability of EvoKD. In different environments, EvoKD should be able to adapt and adjust the strategy. To verify this, we take the category as the indicator, and study the performance changes of the student model on different categories during the training process. We plot the number of samples and F1 of each category on TouTiao news classification under 1-shot in Figure 4. We smooth the line of total number of samples. For better visualization and effectively highlighting the trends, two typical categories are chosen for comparison.

It is obvious that the performance on category "game" has lower F1 at the beginning, leading to an increase in the number of samples. With the increasing number of samples for "game", its F1 score eventually reaches approximately 80% after 400 steps. In contrast, "car" maintains a consistently high level of performance, especially after 150 steps, resulting in a relatively stable number of instances. The increasing number of samples indicates that the LLMs concentrates on improving the performance on this category at that time point, thus generating more samples to teach the

Method	Shot	English			Chinese		AVG
		Amazon	IMDB	Inshorts	TouTiao	CAIL2019	
Full Shot	Full	0.9480	0.9495	0.9705	0.8495	0.9683	0.9372
No Augment	3	0.6704 $\pm$ 0.0446	0.6269 $\pm$ 0.0720	0.8182 $\pm$ 0.0429	0.6623 $\pm$ 0.0254	0.7792 $\pm$ 0.0403	0.7114
AugGPT	3	0.7574 $\pm$ 0.0808	0.7266 $\pm$ 0.1479	**0.8639 $\pm$ 0.0189**	0.6985 $\pm$ 0.0300	0.9000 $\pm$ 0.0606	0.7893
EvoKD	3	**0.8392 $\pm$ 0.0066**	**0.7873 $\pm$ 0.0574**	0.8501 $\pm$ 0.0335	**0.7328 $\pm$ 0.0120**	0.9263 $\pm$ 0.0150	**0.8271**
+Init	3	0.8270 $\pm$ 0.0472	0.7808 $\pm$ 0.0681	0.8561 $\pm$ 0.0359	0.7259 $\pm$ 0.0122	**0.9383 $\pm$ 0.0124**	0.8256
No Augment	5	0.7159 $\pm$ 0.0421	0.7536 $\pm$ 0.0388	0.8693 $\pm$ 0.0292	0.7352 $\pm$ 0.0096	0.8937 $\pm$ 0.0245	0.7935
AugGPT	5	0.8333 $\pm$ 0.0215	0.7858 $\pm$ 0.0728	0.8717 $\pm$ 0.0249	0.7248 $\pm$ 0.0256	0.9413 $\pm$ 0.0067	0.8314
EvoKD	5	0.8228 $\pm$ 0.0301	**0.8485 $\pm$ 0.0463**	**0.8826 $\pm$ 0.0179**	**0.7474 $\pm$ 0.0196**	0.9307 $\pm$ 0.0138	**0.8464**
+Init	5	**0.8682 $\pm$ 0.0130**	0.7729 $\pm$ 0.1607	0.8767 $\pm$ 0.0081	0.7445 $\pm$ 0.0118	**0.9460 $\pm$ 0.0099**	0.8417

Table 3: Experiment results under Few-Shot text classification.

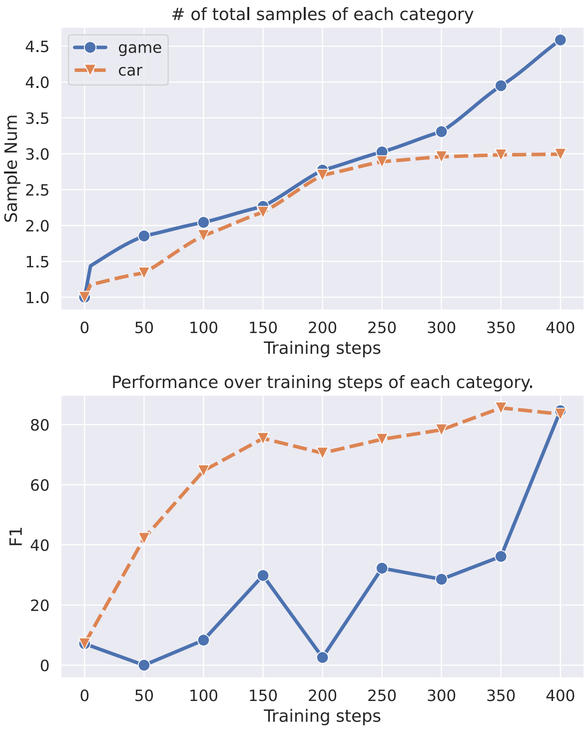

Figure 4: EvoKD concentrates on the samples with lower performance. Note that the upper sub-figure shows the accumulate number of samples of each category. A rising trending means the LLM generates more samples of the category, while a stage of horizontal line indicates that the category is absent in the generated data.

student. The active analysis can contribute to the F1 improvement of "game".

4.7. Few-Shot Classification

We conducted 3-shot and 5-shot text classification experiments and the results are shown in Table 3.

In comparison to AugGPT, EvoKD exhibits higher F1 results and lower deviations across most of the datasets. In the case of TouTiao, EvoKD achieves respective scores of 0.7328 and 0.7474 under the 3-shot and 5-shot settings, which are higher than AugGPT by 3.4 percents and 2.3 percents. EvoKD with initialization exhibits unstable superiority over bare EvoKD. For instance, EvoKD with initialization outperforms EvoKD by 4.5 percents on Amazon, but underperforms it by 7.6 percents on IMDB.

4.8. 1-Shot NER

Method	CoNLL03	CoNLL04	AVG
Full Shot	0.9322	0.8766	0.9044
No Augment	0.3143	0.4929	0.4036
EDA	0.3062	0.5058	0.4060
AugGPT	0.6315	0.6683	0.6499
EvoKD	0.6538	**0.6848**	**0.6693**
+Init	**0.6629**	0.6628	0.6629

Table 4: 1-shot results for NER datasets, where AVG denotes the average F1 performance over CoNLL03 and CoNLL04.

We consider a prediction to be correct only if both the entity type and the entity text align with the ground truth. Based on the metric, we report the F1 results in Table 4.

Generally, the 1-shot performances without KD or EDA are unsatisfactory. Leveraging the generation ability, AugGPT significantly enhances the average result to 0.65, outperforming EDA by a considerable margin. EvoKD achieves the best perfomance. It outperforms AugGPT on CoNLL04 by approximately 2 percents, while EvoKD with AugGPT initialization achieves the highest F1 on CoNLL03, surpassing AugGPT by around 3 percents.

4.9. Ablation Study

We conduct ablation study on our pipeline and the prompt, the results are shown in Table 5. We find that both easy samples and correct samples enhance the quality of the generated texts. Overall, the former is more crucial. The risk associated with removing the easy samples generated by the LLM is more substantial, resulting in a decrease of

Method	Amazon	Inshorts	AVG
EvoKD	**0.8425**	**0.8516**	**0.8471**
w/o Easy	0.5976	0.6491	0.6234
w/o Correct	0.8132	0.8334	0.8233
w/o Review	0.7324	0.8152	0.7738
w/o Separating	0.7112	0.8359	0.7736

Table 5: Ablation Study on 1-shot text classification. We remove D_{easy} and $D_{correct}$ respectively, denoted as "w/o Easy" and "w/o Correct" respectively. In addition, we drop the strategy of "Review History", which is denoted as "w/o Review". And we also merge sub-steps "Input Text Generation" and "Labeling" discussed in SubSection 3.1 together, which is denoted as "w/o Separating".

0.2237 on average F1. As mentioned in Section 3, training only on D_{hard} would cause the problem of catastrophic forgetting. The student model would learn the biased distribution.

Across the pipeline, we remove 1) the strategy "Review History" and 2) merge the sub-steps ""Input Text Generation" and "Labeling". Their performance degradation are roughly equal, about 7.3 percents, which indicates that they are equally important.

5. Conclusion

In this paper, we propose EvoKD: Evolving Knowledge Distillation with LLM and Active Learning, which is an effective framework especially for few-shot setting. We prompt ChatGPT to analyse the weakness of the student model, and subsequently generate samples based on the analysis. The experimental results demonstrate the effectiveness of EvoKD, particularly on most text classification datasets where we achieve 90% of the full-shot performance with only 1-shot.

6. Limitations

We notice that the advantage of our proposed method becomes less as the number of shots increases. This phenomenon could be expected, since an increase in the number of samples leads to a larger pool of data available, and the quality of the generated data is not substantially affected by different KD approaches.

7. Ethics Statement

We introduce EvoKD in this paper. Our design of the prompt texts and samples generated by ChatGPT honors the ethical code. In our experiment, ChatGPT is applied, which is a large language model pre-trained on a large-scale corpus. We encourage researchers to explore whether ChatGPT-generated data is biased and discriminatory before deploying EvoKD to ethically improve the performance of student models.

8. Acknowledgements

This work was supported in part by National Key Research and Development Program of China (2022YFC3340900), National Natural Science Foundation of China (62376243, 62037001, U20A20387), the StarryNight Science Fund of Zhejiang University Shanghai Institute for Advanced Study (SN-ZJU-SIAS-0010), Alibaba Group through Alibaba Research Intern Program, Project by Shanghai AI Laboratory (P22KS00111), Program of Zhejiang Province Science and Technology (2022C01044).

9. Bibliographical References

Dana Angluin. 1988. Queries and concept learning. *Machine learning*, 2:319–342.

Yonatan Belinkov and Yonatan Bisk. 2018. Synthetic and natural noise both break neural machine translation.

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

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

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. 1996. Active learning with statistical models.

Haixing Dai, Zhengliang Liu, Wenxiong Liao, Xiaoke Huang, Yihan Cao, Zihao Wu, Lin Zhao, Shaochen Xu, Wei Liu, Ninghao Liu, Sheng Li, Dajiang Zhu, Hongmin Cai, Lichao Sun, Quanzheng Li, Dinggang Shen, Tianming Liu, and Xiang Li. 2023. Auggpt: Leveraging chatgpt for text data augmentation.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2020. Plug and play language models: A simple approach to controlled text generation.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active prompting with chain-of-thought for large language models.

Aleksandra Edwards, Asahi Ushio, Jose Camacho-Collados, Hélène de Ribaupierre, and Alun Preece. 2023. Guiding generative language models for data augmentation in few-shot text classification.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. 2018. Learning to teach.

Jiahui Gao, Renjie Pi, Yong Lin, Hang Xu, Jiacheng Ye, Zhiyong Wu, Weizhong Zhang, Xiaodan Liang, Zhenguo Li, and Lingpeng Kong. 2023. Self-guided noise-free data generation for efficient zero-shot learning.

Claudio Gentile, Zhilei Wang, and Tong Zhang. 2022. Fast rates in pool-based batch active learning.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Jianping Gou, Baosheng Yu, Stephen J. Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2023. Knowledge distillation of large language models.

Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. 2021. XL-sum: Large-scale multilingual abstractive summarization for 44 languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4693–4703, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

Namgyu Ho, Laura Schmid, and Se-Young Yun. 2023. Large language models are reasoning teachers.

Nathan Lambert, Louis Castricato, Leandro von Werra, and Alex Havrilla. 2022. Illustrating reinforcement learning from human feedback (rlhf). *Hugging Face Blog*. Https://huggingface.co/blog/rlhf.

Jie Lou, Yaojie Lu, Dai Dai, Wei Jia, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2023. Universal information extraction as unified semantic matching.

Yaojie Lu, Qing Liu, Dai Dai, Xinyan Xiao, Hongyu Lin, Xianpei Han, Le Sun, and Hua Wu. 2022. Unified structure generation for universal information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5755–5772, Dublin, Ireland. Association for Computational Linguistics.

Edward Ma. 2019. Nlp augmentation. https://github.com/makcedward/nlpaug.

George A. Miller. 1994. WordNet: A lexical database for English. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Seungwhan Moon and Jaime G. Carbonell. 2019. Learn to active learn: Dynamic active learning with attention-based strategies selection.

Fredrik Olsson. 2009. A literature survey of active machine learning in the context of natural language processing.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

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.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Raphael Schumann and Ines Rehbein. 2019. Active learning via membership query synthesis for semi-supervised sentence classification. In *Proceedings of the 23rd conference on computational natural language learning (CoNLL)*, pages 472–481.

Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*.

Burr Settles. 2009. Active learning literature survey.

Ruixiang Tang, Xiaotian Han, Xiaoqian Jiang, and Xia Hu. 2023. Does synthetic data generation of llms help clinical text mining?

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023b. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023c. Llama 2: Open foundation and fine-tuned chat models.

Solomon Ubani, Suleyman Olcay Polat, and Rodney Nielsen. 2023. Zeroshotdataaug: Generating and augmenting training data with chatgpt.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.

Shuohang Wang, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021. Want to reduce labeling cost? GPT-3 can help. In *Findings* *of the Association for Computational Linguistics: EMNLP 2021*, pages 4195–4205, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. 2017. Community preserving network embedding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2022. Self-instruct: Aligning language model with self generated instructions.

Jason Wei, Chengyu Huang, Soroush Vosoughi, Yu Cheng, and Shiqi Xu. 2021. Few-shot text classification with triplet networks, data augmentation, and curriculum learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5493–5500, Online. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2023a. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *CoRR*, abs/2304.14402.

Yiquan Wu, Siying Zhou, Yifei Liu, Weiming Lu, Xiaozhong Liu, Yating Zhang, Changlong Sun, Fei Wu, and Kun Kuang. 2023b. Precedent-enhanced legal judgment prediction with llm and domain-model collaboration.

Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Ce Bian, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, Fei Deng, Feng Wang, Feng Liu, Guangwei Ai, Guosheng Dong, Haizhou Zhao, Hang Xu, Haoze Sun, Hongda Zhang, Hui Liu, Jiaming Ji, Jian Xie, JunTao Dai, Kun Fang, Lei Su, Liang Song, Lifeng Liu, Liyun Ru, Luyao Ma, Mang Wang, Mickel Liu, MingAn Lin, Nuolan Nie, Peidong Guo, Ruiyang Sun, Tao Zhang, Tianpeng Li, Tianyu Li, Wei Cheng, Weipeng Chen, Xiangrong Zeng, Xiaochuan Wang, Xiaoxi Chen, Xin Men, Xin Yu, Xuehai Pan, Yanjun Shen, Yiding Wang, Yiyu Li, Youxin Jiang, Yuchen Gao, Yupeng Zhang, Zenan Zhou, and Zhiying Wu. 2023. Baichuan 2: Open large-scale language models.

Jiacheng Ye, Jiahui Gao, Qintong Li, Hang Xu, Jiangtao Feng, Zhiyong Wu, Tao Yu, and Lingpeng Kong. 2022. ZeroGen: Efficient zero-shot learning via dataset generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11653–11669, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Michelle Yuan, Hsuan-Tien Lin, and Jordan Boyd-Graber. 2020. Cold-start active learning through self-supervised language modeling. *arXiv preprint arXiv:2010.09535*.

Tianjun Zhang, Yi Zhang, Vibhav Vineet, Neel Joshi, and Xin Wang. 2023. Controllable text-to-image generation with gpt-4.

Yue Zhang, Eduardo Coutinho, Zixing Zhang, Caijiao Quan, and Bjoern Schuller. 2015. Dynamic active learning based on agreement and applied to emotion recognition in spoken interactions. In *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, ICMI '15, page 275–278, New York, NY, USA. Association for Computing Machinery.

Zhe Zhao, Hui Chen, Jinbin Zhang, Xin Zhao, Tao Liu, Wei Lu, Xi Chen, Haotang Deng, Qi Ju, and Xiaoyong Du. 2019. Uer: An open-source toolkit for pre-training models. *EMNLP-IJCNLP 2019*, page 241.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. 2020. Fine-tuning language models from human preferences.

10. Language Resource References

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, pages 89–97, Boston, Massachusetts, USA. Association for Computational Linguistics.

Phillip Keung, Yichao Lu, György Szarvas, and Noah A. Smith. 2020. The multilingual amazon reviews corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts.

2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

A. Interaction Example

We take the sentiment classification as an example and demonstrate our Evolving Active Learning pipeline in Figure 5.

B. ChatGPT Augmentation Example

We list some examples by bare ChatGPT augmentation of sentiment classification in Table 6.

User

There is a language model that classifies the sentiment of sentences with "positive" or "negative".I have some comments of books or movies that the model correctly classified, and some other sentences that the model incorrectly classified.
The model incorrectly predicted:
1. "sentence": "I loved the movie! It was such a sad and heartwarming story."; "label": "positive"; "prediction": "negative"
2. "sentence": "The acting in the play was mediocre, but the set design was fantastic."; "label": "positive"; "prediction": "negative"
The model correctly predicted:
1. "sentence": "The sci-fi novel was a thrilling ride, with a unique concept and tense action scenes.", "label": "positive"; "prediction": "positive"
2. "sentence": "The movie had beautiful cinematography, but the acting was terrible.", "label": "negative"; "prediction": "negative"
I want you to summarize the patterns of sentences that are prone to be incorrectly classified by the model, based on the sentences I provided.

ChatGPT

Based on the provided sentences, it appears that the language model is prone to incorrectly classify sentences that contain contrasting statements (such as praising the acting but criticizing the script), and sentences that contain a mixture of positive and negative language. It also seems to struggle with sentences that express a strong negative opinion and provide multiple examples to support it. In general, the model may struggle with accurately interpreting more nuanced or complex opinions about books or movies.

User

Based on your summary, please assume 2 comments of books or movies, that the model would classify incorrectly and 2 sentences that the model would classify correctly.
The overall sentiment of each comment must be clearly either positive or negative.
You should only reply with the sentence. Don't explain the label or other information. Reply in the following format:
The model may incorrectly predict:
1. Sentence: "xxx"
2. ...
The model may correctly predict:
1. Sentence: "xxx"
2. ...

ChatGPT

The model may incorrectly predict:
1. Sentence: "The ending was disappointing, but overall it was a great movie.
2. Sentence: "The beginning of the book was slow and boring, but the heartwarming ending made up for it."
The model may correctly predict:
1. Sentence: "The movie was absolutely hilarious and had me laughing throughout."
2. Sentence: "The plot was boring and poorly developed."

User

I have some comments of books or movies:
1. Sentence: "The ending was disappointing, but overall it was a great movie.
2. Sentence: "The beginning of the book was slow and boring, but the heartwarming ending made up for it."
3. Sentence: "The movie was absolutely hilarious and had me laughing throughout."
4. Sentence: "The plot was boring and poorly developed."
I want you to label the overall sentiment with either "positive" or "negative".'
The label is the overall sentiment to judge the product is good or bad, and you are not allowed to label mixed sentiments such as "neutral". Don\'t explain other things.
Your answer should be like:
1. positive/negative
2. positive/negative
3. ...

ChatGPT

1. positive
2. positive
3. positive
4. negative

Figure 5: Example of evolving active learning of sentiment classification task.

Original Text	Augmented Text
I purchased this item after reading reviews. Using the directions for roasting in the microwave the garlic was burnt to a crisp, the smell lingered for days. I immediately threw this garlic roaster in the trash. I will use foil and bake in the oven. What a waste of money.	The hype around this restaurant was unwarranted - the food was overpriced and underwhelming. I regret spending my money here.
Kylie is back in the States! Though she was never missing from the pop music world always making music... I like it, and I think it is some of the best dance music to come from Kylie in a while.	Kylie's latest single "Fever" is a subtle but catchy dance song with rhythmic beats and a hook that grows on you - some of the best dance music from her in a while, and I like it!
i ordered five of these for my grandkids because I saw an "8 x 10.5" book and the contents are great- it looked wonderful! BUT, when they arrived I was very disappointed to see... The website does not show the size of the book. Who makes childrens books in what must be about a size 6 font? I feel like I wasted my money.For heavens sake put the size on your books!	The product photos looked fantastic, but the actual item was a disappointment, as it was much smaller than expected and its features became hard to see.
I was held completely enthralled through the first half of this book. i loved the setting and it's description. I loved the heroine and the "wee lad" she took under her wing. My only problems with it were that, for me, there weren't enough love scenes, and they arrived too late in the story...and the fact the author decided to throw in another couple into the mix, without fully fleshing out their story. I think Ms. Garwood should have stuck to her main protagonists and left the other couple for another book	I thoroughly enjoyed the storyline and well-rounded characters, but would have preferred a stronger focus on the central romance and less on the supporting characters.

Table 6: Examples augmented by bare ChatGPT.