CORAL: Collaborative Automatic Labeling System based on Large Language Models

Zhen Zhu[1], Yibo Wang[1], Shouqing Yang[1], Lin Long[1], Runze Wu[2], Xiu Tang[1], Junbo Zhao[1], Haobo Wang[1]
[1]Zhejiang University, Hangzhou, China
[2]NetEase Fuxi AI Lab, Hangzhou, China
hzzhuzhen@yeah.net,wanghaobo@zju.edu.cn

ABSTRACT

In the era of big data, data annotation is integral to numerous applications. However, it is widely acknowledged as a laborious and time-consuming process, significantly impeding the scalability and efficiency of data-driven applications. To reduce the human cost, we demonstrate CORAL, a collaborative automatic labeling system driven by large language models (LLMs), which achieves high-quality annotation with the least human effort. Firstly, CORAL employs LLM to automatically annotate vast datasets, generating coarse-grained labels. Subsequently, a weakly-supervised learning module trains small language models (SLMs) using noisy label learning techniques to distill accurate labels from LLM's annotations. It also allows statistical analysis of model outcomes to identify potentially erroneous labels, reducing the human cost of error detection. Furthermore, CORAL supports iterative refinement by LLMs and SLMs using manually corrected labels, thereby ensuring continual enhancement in annotation quality and model performance. A visual interface enables annotation process monitoring and result analysis.

PVLDB Reference Format:
Zhen Zhu, Yibo Wang, Shouqing Yang, Lin Long, Runze Wu, Xiu Tang, Junbo Zhao, and Haobo Wang. CORAL: Collaborative Automatic Labeling System based on Large Language Models. PVLDB, 17(12): 4401 - 4404, 2024.
doi:10.14778/3685800.3685885

PVLDB Artifact Availability:
The source code, data, and other artifacts have been made available at https://github.com/hzzhuzhen/CORAL.

1 INTRODUCTION

In the era of big data, data annotation plays a pivotal role in many applications, such as data integration and machine learning. However, it is notoriously a laborious and time-consuming process that largely hinders the scalability and efficiency of data-driven applications. In Figure 1, we illustrate the most common pipeline of data annotation that comprises three iterative stages: data annotation, error detection, and label refinement. Notably, each stage requires massive human efforts, resulting in high annotation costs.

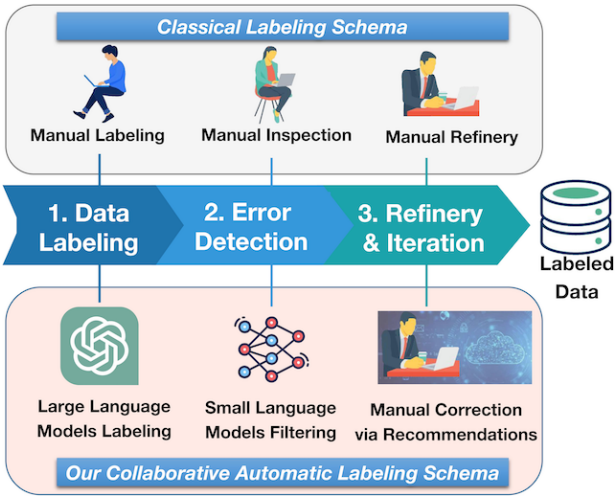

Figure 1: Comparison between classic labeling approaches and our collaborative labeling framework. Our CORAL framework consists of three core modules: (i) LLMs for cheap annotation; (ii) weakly supervised small models for error filtering; and (iii) a user-friendly UI for manual correction.

To remedy this problem, researchers have explored diverse solutions. Among these, weakly-supervised learning (WSL) [2, 3, 8] stands out as an effective approach that attempts to reduce the necessity for precise supervision. For example, active learning [3] only annotates a subset of the most important samples to achieve superior performance; semi-supervised learning [8] allows training models from a small set of labeled samples along with a huge amount of unlabeled data; noisy label learning (NLL) [2] allows the training corpus to contain many wrong labels. However, despite advancements, WSL still necessitates significant annotation costs to ensure decent performance.

In recent years, the emergence of large language models (LLMs) has introduced another promising paradigm, showcasing remarkable zero-shot/few-shot capabilities, which hold promise for alleviating annotation expenses. Even more inspiringly, they emerge with in-context learning (ICL) [11] ability that demonstrates impressive performance on domain-specific tasks given a few task-related labeled samples. However, some recent works [1] have observed that LLMs tend to lag behind fine-tuned small language models (SLMs) when confronted with challenging tasks. To date, it remains

*Haobo Wang is the corresponding author.

Proceedings of the VLDB Endowment, Vol. 17, No. 12 ISSN 2150-8097.
doi:10.14778/3685800.3685885

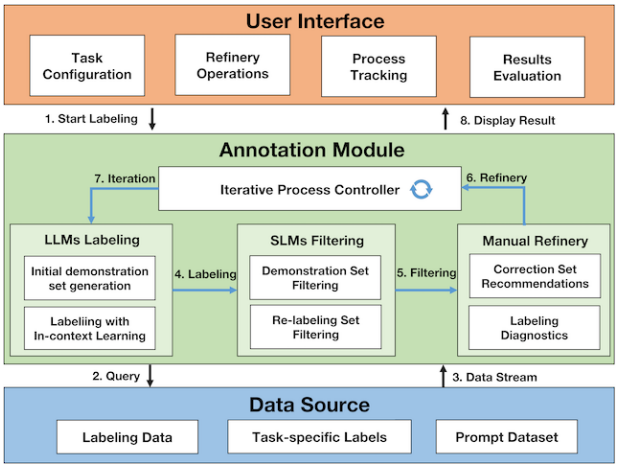

Figure 2: The architecture of CORAL.

an unresolved question of how we can generalize to downstream tasks with minimal human efforts in the era of LLMs.

To address the problems, we demonstrate the **CO**llabo**R**ative **A**utomatic **L**abeling (CORAL) system, designed to streamline the data annotation process while maintaining label accuracy based on our previous work [9]. CORAL comprises three core components: (a) a large language model annotation module capable of automatically annotating vast corpora, generating coarse-grained labels; (b) a weakly supervised learning module that trains small models using noisy label learning techniques to distill accurate labels from the coarse-grained annotations; and (c) a label refinement mechanism that recommend potentially erroneous labels based on statistical analysis of model outcomes. Finally, CORAL also supports iterative refinement by LLMs and SLMs using manually corrected labels, thus ensuring continual improvement in annotation quality and model performance.

Based on CORAL, users can quickly acquire huge amounts of high-quality annotated data at minimal human resource costs, expediting the development of big data applications.

2 SYSTEM OVERVIEW

In this section, we present the details of our CORAL system. As depicted in Figure 2, CORAL comprises three key parts:

- **User-Friendly Interface** makes the labeling process transparent to users and provides a friendly interaction interface to users for the refinery including take configuration, refinery operations, process tracking, and results revaluation.
- **Annotation Module** is the core module of CORAL that performs annotation with the least human costs. It contains LLM-driven labeling, SLM-driven filtering, refinery components, and iterative process controller components.
- **Data Management Module** handles and organizes data within our system, including data storage preprocessing, and integration. It first writes user-input unlabeled data, defined labels, and task configurations into the database. Then it is responsible for managing data interaction between different modules and updating new annotation results.

Table 1: An example of prompt for sentiment analysis task.

Prompt Template for the Sentiment Analysis Task
`[Task Description]` You are a helpful assistant for the task of sentiment analysis. You reply with brief, to-the-point answers with no elaboration as truthfully as possible. Your task is to a binary classification to classify content as positive or negative according to their overall sentiment polarity. The category is divided into two types: 'positive' and 'negative'. `[In-Context Demonstration]` You can refer to the following labeled samples for prediction: ["content": "enigma is well-made, but it's just too dry and too placid.", "label": "negative"]... `[Output Control]` Given a content: <QUERY>. How do you feel about the sentiment polarity of the given content? Is this positive or negative? please answer in a single line with 'positive' or 'negative'.

The most innovative feature of CORAL is to support automated data annotation and error filtering, making annotation less human-dependent. Hence, in what follows, we focus on our core annotation modules, which comprise four components:

(1) LLM-driven labeling component. In this component, we leverage the great zero-shot/few-shot learning ability of LLMs to replace human annotators. Given any labeling task, we can design task-specific prompts for LLM to produce a vast amount of weak annotations on given unlabeled corpora. Notably, one can modify the prompts to address different tasks, such as various text classification [1] and tabular data analysis [6]. In Table 1, we demonstrate a prompt template for the sentiment analysis task.

To further improve the annotation quality, we also include the in-context learning [11] algorithm such that LLM can mimic the pattern of labeled samples in [In-Context Demonstration]. Here, the demonstration set can not necessarily be human-annotated or even fully precise, which will be discussed later.

(2) SLMs filtering component. It is worth noting that LLM can inevitably introduce erroneous labels and it is difficult for the LLM itself to distinguish its own errors. To this end, we train a robust small language model by noisy label learning (NLL) [4] to detect noisy samples during the training procedure. After that, its outcomes, including feature representations and prediction confidences, can be employed for designing indicators for detecting error labels. For example, those wrong-labeled samples typically demonstrate low confidence. Moreover, the SLM itself can produce much more accurate predictions than the LLM's outputs, achieving a human-free label refinery.

Notably, for non-language tasks, one can also replace the small language model with another task-specific model, e.g., multi-layered perceptions for tabular data analysis.

(3) Manual refinery component. It is unfortunate that relying solely on machine models for labeling is unable to address some challenging samples. For real-world applications, human knowledge is indispensable in controlling the quality of annotations. In this component, we highlight those potentially wrong samples through the user interfaces, e.g., data with very low prediction confidence. Then, users can manually correct these recommended

samples with the help of displayed information such as LLM/SLM predictions, confidences, and so on. This component can largely relieve the burden of human correction costs, transferring human expertise to solving the most difficult labeling problems.

(4) Iterative process controller component. Inspired by the active learning algorithms, CORAL also enables iterative annotation to maximally improve the annotation quality. This controller component implements two main functionalities. Firstly, to enhance the proficiency of the LLM, this component gathers highly probable clean samples based on model outcomes and identifies the most representative samples to serve as in-context demonstration examples. With the help of precise demonstration samples, LLM is able to re-label the training corpora with improved accuracy. Secondly, it aggregates the machine/human labels to retrain a more robust small model. One can optionally rerun (1) and (2) components many times through our controller component.

3 IMPLEMENTATION DETAILS

To better understand how labeling works in CORAL, this section will explain the implementation of three key issues.

SLMs Distillation Strategy: While any off-the-shelf NLL algorithm can be used, we choose the most popular selection-based technique [4] for our implementation. The intuition is that those clean samples typically demonstrate smaller loss values because deep networks tend to fit easy patterns during early training stages [10]. Therefore, we can separate clean samples for robust training according to loss values. Following [4], after a few warm-up epochs of standard training on noisy labels, we fit a two-component Gaussian Mixture Model (GMM) to the loss function to identify clean samples belonging to the Gaussian component with a smaller mean, indicative of their clean probability. To maximize the robustness, we further develop a semi-supervised learning (SSL) procedure by regarding clean samples as labeled and the remaining data as unlabeled. In our implementation, we follow the pioneering FixMatch algorithm for SSL training. We perform these two steps at the beginning of each training epoch, until the convergence.

In-Context Demonstration for LLM: The quality of LLM's annotation depends on two key factors, i.e., a good prompt and in-context learning (ICL). Though we provided few templates for specific tasks, the prompt design largely depends on human knowledge. Thus, this section focuses on improving the ICL ability in our CORAL system. Through our experiments, we find that even noisy in-context demonstration samples still produce can bring large performance improvement. Consequently, we develop two strategies to obtain in-context samples. At the initial round of annotation, we perform **Self-Labeling**, which randomly annotates a few samples by LLM and integrates them into the prompt. After the subsequent rounds where SLM are well-trained, we perform **Distilled Selection**. Specifically, we first collect all clean samples detected by SLM as well as those manually corrected. Then, we run a simple k-medoids clustering algorithm on SLM's features to gather the most representative medoids of k-clusters for ICL. As the annotation procedure proceeds, the ICL demonstrates data becomes increasingly accurate, fully unleashing the great power of LLM for refined annotations.

Performance Evaluation: Due to space limitations, we illustrate the model's performance specifically on the MR [7] dataset for sentiment classification. Table 2 demonstrates that CORAL significantly enhances unsupervised performance with the free LLM's iterator and SLM's refinery. In Table 2, we adopt Llama3-8B[1] as LLM and we use RoBERTa-Base[5] as the downstream SLM. CORAL achieves competitive results and substantially reduces low-confidence samples at minimal cost associated with previous human labeling efforts. Further analyses and more results with GPT3.5[2] can be found in our previous work[9].

Table 2: Comparisons of different processes on MR dataset.

Model	Step	Process	Accuracy
LLMs	1	Llama3-8B zero-shot labeling	87.69
SLMs	2	RoBERTa first distillation	90.94
LLMs	3	Llama3-8B labeling refinery	91.03
SLMs	4	RoBERTa second distillation	92.44

4 DEMONSTRATION SCENARIOS

Figure 3 is the screenshot of the CORAL user interface, which allows users to work on our novel labeling processes in CORAL through the following steps:

Step 1 (Task configuration). The configuration interface, as illustrated in Figure 3-1, guides users through the initial setup. Users begin by selecting the type of labeling task with current options including sentiment analysis, topic classification, and tabular prediction. The system is designed with a flexible strategy schema, allowing for easy extension to accommodate additional task types. Following task selection, users upload the target unlabeled data via the "Upload" button. Additionally, CORAL offers prompt template configuration for specific tasks but allows the users to tailor prompts to their specific needs. Another crucial configuration step involves defining customized label names for CORAL labeling tasks. Basic task information such as LLM and SLM selection, as well as epochs for SLMs, must also be configured. In our screenshot, we use GPT-Turbo-3.5 and RoBERTa respectively. Upon submission of the task configuration, the task transitions to the annotation module for the primary labeling process.

Step 2 (LLMs labeling). Next, the LLM starts to perform data annotation according to the pre-defined prompts. As depicted in Figure 3-2, users can track the progress of the LLM labeling through the progress bar at the bottom left corner. The table on the right side displaying LLM labeling results will be updated continuously. Upon completion of LLM's coarse annotation, users can proceed by clicking the "Next" button, transitioning CORAL to Step 3: SLM filtering.

Step 3 (SLMs filtering). The SLM filtering is exactly the procedure for CORAL auditing of the result of the preceding work. The system presents confidence levels and Gaussian loss rankings for each sample in the table on the right for analysis. SLMs iterates continuously to refine predictions based on cross-entropy loss.

Figure 3-3 provides a visualization of this process, depicting the current epoch and progress bar at the bottom left, while the scatter

[1] https://llama.meta.com/llama3
[2] https://chat.openai.com

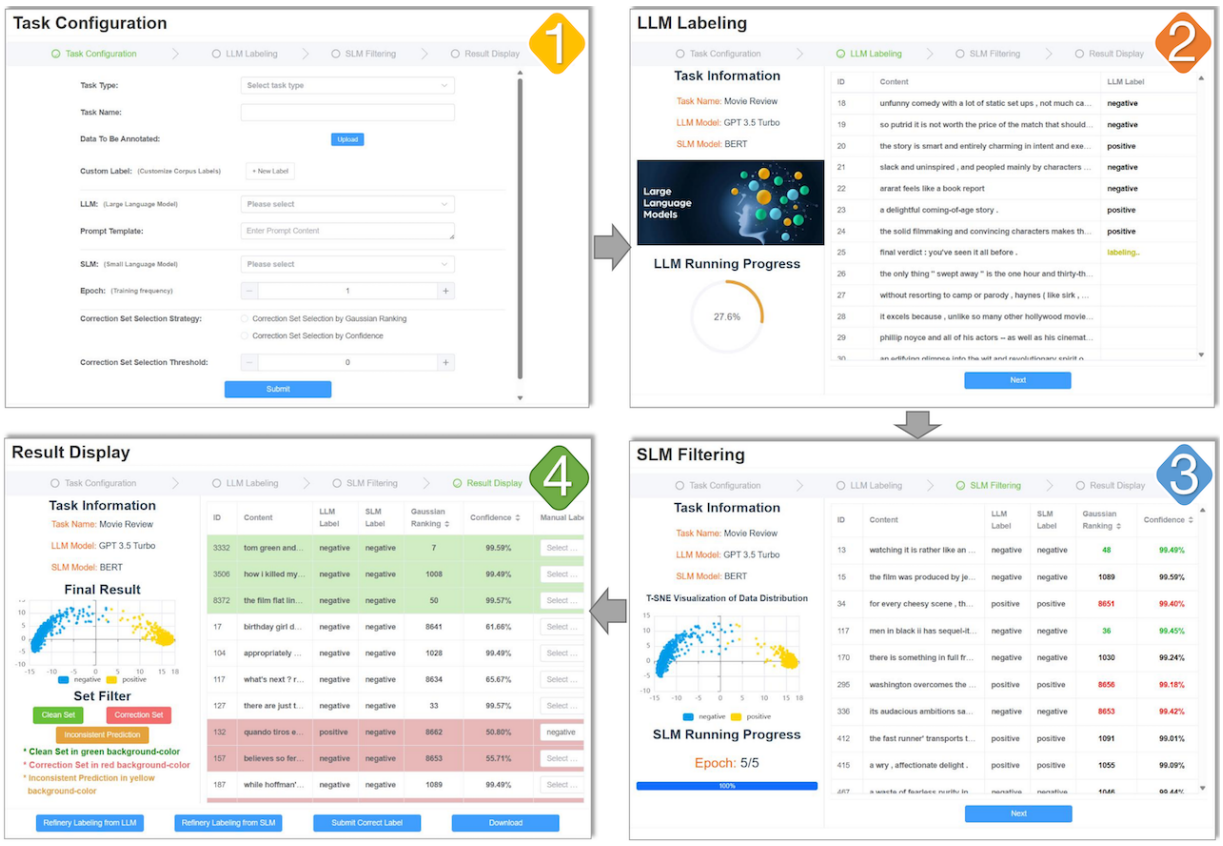

Figure 3: A Screenshot of CORAL with an example annotation task of sentiment analysis.

graph on the left illustrates the distribution of data based on SLM's features. The table on the right shows the process data filtered by the SLMs. Once filtering work is finished, we supplement several new columns showing SLM's predicted labels, confidence values, and the ranking of GMM's probability. We also highlight those Data that are likely to be clean or noisy in green and red, respectively. After that, users can advance to the next step via the "Next" button.

Step 4 (Results display and manual correction). In this step, CORAL displays the final results of labeling and filtering. As shown in Figure 3-4, the right side table exhibits labeling results with an additional manual correction column. Three buttons on the left allow the users to highlight those samples with a high probability of being *clean* (high confidence), *noisy* (low confidence), and *inconsistent* (different SLM/LLM's predictions). Those clean samples are marked with a green background, while noisy and inconsistent data are in red. We also allow the users to sort their data according to the SLMs' confidence or the GMM probability. All these UI designs allow the user to better perceive the degree of annotation error. Accordingly, by clicking the "Submit Correct Label" button, users can manually update labels in the correction column.

Optional Step (Iterative annotation). Finally, CORAL supports looped iterative annotation, allowing users to proceed to the next iteration from LLMs labeling or SLMs filtering by clicking "Refinery labeling from LLMs" or "Refinery labeling from SLMs" respectively. Then, the users will turn back to the pages of Steps 2-4 with basically the same UI interfaces. On each page, the system can utilize the last manual correction set to inform LLMs and SLMs, respectively. Finally, users can download an Excel file containing labeled data by clicking on the "Download" button.

REFERENCES

[1] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. In *IJCNLP*. ACL, 675–718.

[2] Mohamad Dolatshah, Mathew Teoh, Jiannan Wang, and Jian Pei. 2018. Cleaning crowdsourced labels using oracles for statistical classification. *Proc. VLDB Endow.* 12, 4 (dec 2018), 376–389.

[3] Arjit Jain, Sunita Sarawagi, and Prithviraj Sen. 2021. Deep indexed active learning for matching heterogeneous entity representations. *Proc. VLDB Endow.* 15, 1 (sep 2021), 31–45.

[4] Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. ICLR, 1–14.

[5] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019).

[6] Weizheng Lu, Jiaming Zhang, Jing Zhang, and Yueguo Chen. 2024. Large Language Model for Table Processing: A Survey. arXiv:2402.05121 [cs.AI]

[7] Bo Pang and Lillian Lee. 2005. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *ACL*. 115–124.

[8] Paroma Varma and Christopher Ré. 2018. Snuba: automating weak supervision to label training data. *Proc. VLDB Endow.* 12, 3 (nov 2018), 223–236.

[9] Ruixuan Xiao, Yiwen Dong, Junbo Zhao, Runze Wu, Minmin Lin, Gang Chen, and Haobo Wang. 2023. FreeAL: Towards Human-Free Active Learning in the Era of Large Language Models. In *EMNLP*. ACL, 14520–14535.

[10] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2021. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM* 64, 3 (feb 2021), 107–115.

[11] Yuanhan Zhang, Kaiyang Zhou, and Ziwei Liu. 2023. What Makes Good Examples for Visual In-Context Learning?. In *NeurIPS*.