Hide and Seek in Noise Labels: Noise-Robust Collaborative Active Learning with LLM-Powered Assistance

Bo Yuan[1], **Yulin Chen**[1], **Yin Zhang**[1*], **Wei Jiang**[2]
[1]Zhejiang University, Hangzhou, China
[2]Ant Group, Hangzhou, China
{byuan,yulinchen,yinzh}@zju.edu.cn
shouzhi.jw@antgroup.com

Abstract

Learning from noisy labels (LNL) is a challenge that arises in many real-world scenarios where collected training data can contain incorrect or corrupted labels. Most existing solutions identify noisy labels and adopt active learning to query human experts on them for denoising. In the era of large language models (LLMs), although we can reduce the human effort to improve these methods, their performances are still subject to accurately separating the clean and noisy samples from noisy data. In this paper, we propose an innovative collaborative learning framework NoiseAL based on active learning to combine LLMs and small models (SMs) for learning from noisy labels. During collaborative training, we first adopt two SMs to form a co-prediction network and propose a dynamic-enhanced threshold strategy to divide the noisy data into different subsets, then select the clean and noisy samples from these subsets to feed the active annotator LLMs to rectify noisy samples. Finally, we employ different optimization objectives to conquer subsets with different degrees of label noises. Extensive experiments on synthetic and real-world noise datasets further demonstrate the superiority of our framework over state-of-the-art baselines. The code is available at https://github.com/byuan186/NoiseAL.

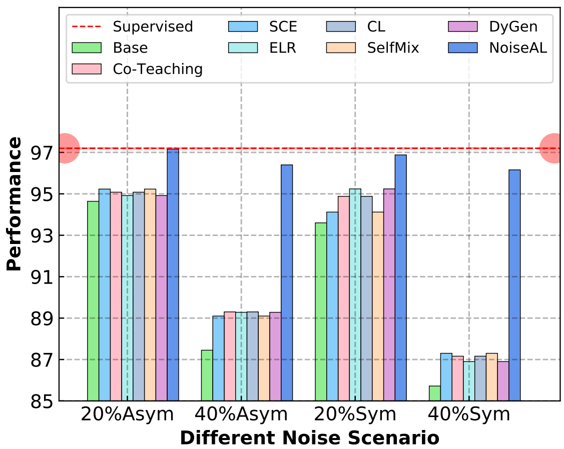


Figure 1: Comparisons of our proposed NoiseAL with previous LNL methods on the Trec dataset under different noise scenarios. NoiseAL surpasses all other baselines and under some scenarios near the performance supervised on ground truth labels.

1 Introduction

The core of deep learning models' success lies in the precision of large amounts of annotated data, which can be a complex and labor-intensive process. In practical scenarios, some researchers often collect datasets by web-crawling (Li et al., 2017) or crowd-sourcing (Yan et al., 2014) to reduce the burden of annotation. However, the obtained datasets frequently suffer from the presence of noisy labels, which will mislead the learning patterns and subsequently result in incorrect predictions. Existing LNL methods include designing robust loss functions (Zhang et al., 2018; Ma et al., 2020), estimating noise transition matrices (Patrini et al., 2017; Zhang et al., 2021c), etc. Among them, active learning (Zhao et al., 2011; Younesian et al., 2021) is a popular solution that provides more accurate labels for noisy samples by querying experts, but it still requires human effort in the annotation.

Recently, large language models (LLMs), such as ChatGPT, have exhibited strong zero-shot learning ability, avoiding human costs for text annotation tasks. Some studies indicate that zero-shot ChatGPT classifications outperform crowd workers in some domains (Shu et al., 2019; Gilardi et al., 2023). Despite the promise, further studies (Bang et al., 2023; Xiao et al., 2023) observe that LLMs tend to underperform compared to BERT supervised on complex datasets. This observation is further supported by our own empirical studies (Table 1). Inspiringly, LLMs emerge with the in-context learning (ICL) ability to learn from a few labeled

*Corresponding Author

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 10977–11011
August 11-16, 2024

samples for boosted performance. However, ICL is sensitive to the input prompt, where acquiring ground-truth labeling of the input demonstrations is important for good ICL performance (Mavromatis et al., 2023). So, we study the following problem: *Given a noisy dataset, how can we accurately identify noisy samples for active querying LLMs and successfully select clean samples as ICL examples?*

Before the era of LLMs, the traditional methods (Han et al., 2018; Shu et al., 2019; Qiao et al., 2022) to divide the noisy data depending on the loss value. The key idea is to set a fixed threshold for loss value, such that the clean samples are associated with a smaller mean loss value and noisy ones with bigger values. However, a recent study (Gao et al., 2023) argues that these methods fail to select clean samples under some synthetic noisy scenarios. Our empirical studies verify this opinion and further observe the mean value of samples with clean labels and noisy labels gradually decreases during training (see Appendix A). So it is not ideal to set a fixed threshold, particularly when we consider SMs'(e.g., BERT, BiLSTM) inherent memory effect that the memorization strength of samples increases during training (See Figure 4 (a-c)). Besides, some memory effect-based work indicates that the confirmation bias (Li et al., 2020) makes SM memorize itself mistakes and hard to discern noisy labels by itself (Xiao et al., 2023). While previous methods could partition datasets, they often struggle to accurately select clean samples in different LNL scenarios, thus failing to construct high-quality ICL for querying LLMs.

To overcome the above limitations of traditional selection methods, we present a novel collaborative learning paradigm, NoiseAL, that utilizes active learning to marry LLMs with SMs by filtering noisy data with the help of SMs and distilling the related knowledge from the LLMs. Specifically, we first adopt two SMs to form a co-prediction network, focusing on different predictive capabilities and producing multiple predictions for each text to mitigate confirmation bias. Our intuition is that the presence of noisy labels may make it difficult for the two SMs to arrive at a consensus on the outputs. In this scenario, one SM may start fitting the noise before another, leading to divergent co-predictions. With two prediction results, we then propose *a dynamic-enhanced threshold strategy following memory effect to select clean and noisy samples to LLMs*. To integrate SMs and LLMs as a whole, we design a collaborative training framework where SMs operate as filters to divide the noisy dataset into different subsets and the LLM acts as an active annotator to correct noisy samples from subsets. During collaborative training, SMs can learn the knowledge of LLMs to boost their performance, while LLMs can also benefit from the divided clean samples to boost their ICL ability. As shown in Figure 1, NoiseAL achieves competitive results compared with its counterparts supervised on the ground truth label under some scenarios. Overall, our main contributions are:

- Based on the memory effect, we innovatively utilize a co-prediction network, combined with a dynamic-enhanced threshold strategy, to select clean samples and noisy samples from noisy data.
- We propose a novel collaborative learning framework termed NoiseAL, which employs SMs as filters to segment noisy data and LLMs as active annotators without any human effort.
- We conduct experiments on diverse text classification datasets under varied noise conditions, revealing the superiority of our proposed NoiseAL against current baselines.

2 Related Work

2.1 Memory Effect of Small Models (SMs)

Carlini et al. (2019) demonstrated that BiLSTM models are able to consistently memorize examples during the very first phase of training. Tänzer et al. (2021) show that BERT forgets examples at a slower rate than BiLSTM and other non-pretrained models. Inspired by the above work, we believe that BiLSTM (weak model) starts fitting the noise before BERT (strong model) and adopt these two SMs to compose a co-prediction network producing different predictions, which helps to mitigate confirmation bias and separate noisy data. Moreover, Li et al. (2023) also observes the memorization strength of SMs for given labels towards individual samples improves during training, which resembles our observations and inspires us to provide a dynamic-enhanced threshold strategy.

2.2 Learning with Noisy Labels

Previous LNL studies can be categorized into three groups: (1) *Sample Selection* approaches (Han et al., 2018; Shu et al., 2019; Qiao et al., 2022) based on loss values that rely on the assumption that clean samples tend to have a smaller mean loss value. These methods require manually setting

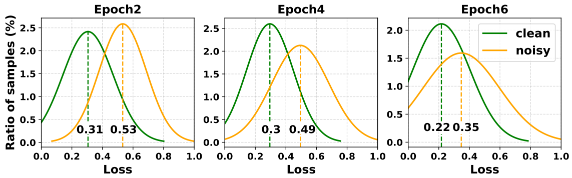


Figure 2: The loss distributions of Bert on Trec dataset under 40% asymmetric noise in different training stages. The solid line represents the loss distributions, and the dashed line points out the mean value of loss distributions. During training, the clean samples tend to have a smaller loss value and the noisy samples tend to have a bigger loss value. And the loss distributions of clean and noisy samples are becoming more consistent.

a fixed threshold for the loss value, which makes their performance questionable in some scenarios. Based on our observation that the loss distributions dynamically evolve during training, we propose a dynamic-enhanced threshold strategy to improve these methods. (2) *Label Correction* methods aim to correct the noisy labels, which adopt noise transition matrix estimation (Sohn et al., 2020; Zhang et al., 2021c) or *AL*. While traditional *AL* relies on expensive expert labeling, we explore the rich knowledge of LLMs to generate labels for noisy samples without human effort. (3) *Regularization* methods (Wang et al., 2019; Liu et al., 2020; Northcutt et al., 2021b; Zhuang et al., 2023) enhance model resistance to label noise by crafting robust loss functions or adopting regularized strategies. Although DyGen (Zhuang et al., 2023) also considers dynamic training, it uses dynamic patterns in the embedding space, which is different from ours.

3 Background

Let $\mathcal{D}$=$\{(x_i, y_i)\}_{i=1}^{N}$ denote the training data with noisy labels, where x is the text, y is its label, and N is the data size. Denote $f_\theta(x) \in \mathbb{R}^{\mathcal{K}}$ as the output of the model f with a linear layer (i.e., the classifier), where θ is its parameters and $\mathcal{K}$ is the number of classes. The confidence of x for each label $k \in [1, \mathcal{K}]$ can be represented as follows: $p(k; x) = \frac{e^{f_\theta(k;x)}}{\sum_{k=1}^{\mathcal{K}} e^{f_\theta(k;x)}}$. Generally, as the SM's memory strength regarding a text increases, the prediction confidence also correspondingly rises. Based on this association, we quantify the SM's memorization strength through a confidence metric. Thus, we regard $p(k; x)$ as the value of memorization strength. For the text classification, if a model memorizes a text, its confidences $p(k; x)$ of k exceed a certain threshold or reach the maximum.

4 Method

In this section, we introduce our proposed framework NoiseAL which explores the opportunity for active learning to address LNL problems in the LLMs era. Although LLMs can generate new labels for noisy samples directly, we still need SMs to help separate noisy data. In each training loop, we alternate the following steps: (1) Training a co-prediction network consisting of two SMs to divide the noisy data into different subsets. (2) Selecting clean and noisy samples from subsets, the noisy samples are corrected by active querying LLMs, and the clean samples are used to prompt ICL. Figure 3 shows the framework of NoiseAL. And its overall pipeline is shown in Algorithm 1. Following that, we will provide a detailed explanation of our proposed NoiseAL framework.

4.1 Dynamic-Enhanced Threshold Strategy

For clean samples and noisy samples, we observed that their mean loss value decreases and loss distribution gradually tends to be consistent during training, as shown in Figure 2. The observed phenomena could potentially be attributed to the memory effect of SMs. Specifically, in the early epochs, noisy samples tend to have a higher loss and can be selected by setting a fixed threshold. However, as training continues, these noisy samples will be memorized gradually, and their loss accordingly decreases. In this situation, it is inappropriate to set a fixed threshold. So, we propose a novel dynamic-enhanced threshold strategy to provide both dynamic and fixed thresholds.

Dynamic Threshold. To capture the increasing memorization strength, we introduce a dynamic threshold $\tau(t)$ for each sample x: $\tau(t) = \lambda p(t) + (1 - \lambda)\tau(t-1)$, $\tau(0) = 0$, where $p(t) = max(p(k; x))$, $p(t)$ is the maximum confidence of the current training epoch t, λ is a hyperparameter controlling threshold stability. The idea of $\tau(t)$ is that the threshold for determining whether a model memorizes a sample should also increase accordingly with the increase of historical confidence. However, the confidence may be unstable when SM begins to overfit noise samples. Hence, we utilize the momentum $p(t)$ attained by each sample across all prior epochs as the threshold value.

Fixed Threshold. Preliminary studies indicate that the loss distributions of both clean and noisy samples during training appear to follow two Gaussian distributions (Qiao et al., 2022). Exploiting

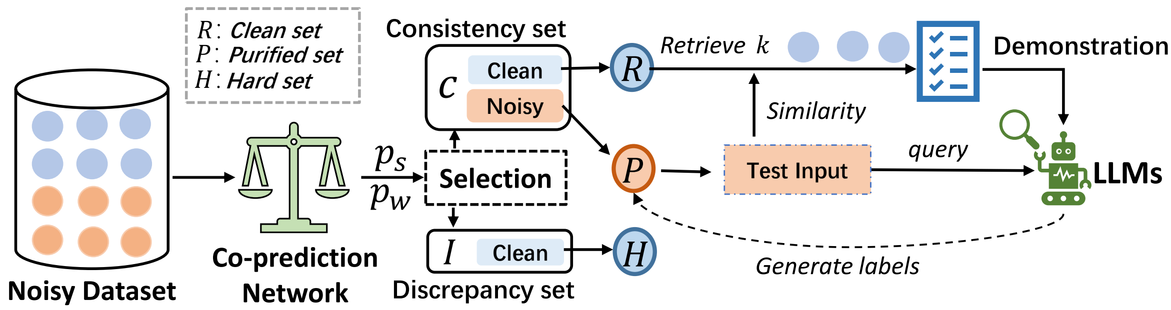


Figure 3: The overview of NoiseAL. During collaborative training, the SMs serve as a filter: (1) employs a co-prediction network (strong model and weak model) to obtain different confidences (p_s and p_w). Based on p_s and p_w, (2) the dynamic-enhanced selection module first divides the noisy data into consistency set $\mathcal{C}$ and discrepancy set $\mathcal{I}$, then groups these two sets into the clean set $\mathcal{R}$, hard set $\mathcal{H}$, and purified set $\mathcal{P}$. Meanwhile, the LLMs serve as active annotators: (1) construct demonstrations by selecting clean samples from $\mathcal{R}$, which can prompt its ICL performance; (2) query the noisy sample from $\mathcal{P}$ and generate labels to imbue its knowledge to SMs.

these findings, we compute the loss $\mathcal{L}$ (Section. 4.4) on the training samples and fit $\mathcal{L}$ to a two-component Gaussian Mixture Model (GMM) (Li et al., 2020) using the Expectation-Maximization algorithm. For sample x, let $l(x)$ represent its loss, and $o(x) = p(g|l(x))$ represent its clean probability $o(x)$, where g is the Gaussian component with a smaller mean value (smaller loss). Then, we set a fixed threshold ϕ for $o(x)$ to distinguish whether x belongs to clean or noisy.

4.2 Dynamic-Enhanced Selection

We combine dynamic thresholds $\tau(t)$ and fixed thresholds ϕ to present our dynamic selection.

Co-prediction Network. Recent research found that training a model using the data divided by itself could lead to confirmation bias (Li et al., 2020), as noisy samples would keep having lower losses due to the model overfitting their labels. To solve this problem, we propose a co-prediction network, in which two models diverge from each other due to different (random) parameter initialization, sequence lengths, and training text features. Being diverged offers the two models diverse evidence (confidence) to memorize samples, making the whole network more robust than a single model.

We first use the diverse confidence of the co-prediction network and dynamic threshold $\tau(t)$ to divide the noisy data into two subsets (see Figure 3), *i.e.*, a consistency set $\mathcal{C}$ and a discrepancy set $\mathcal{I}$ which are obtained according to the consistency and discrepancy of confidences. Next, we finely group these two sets into three subsets according to the fixed threshold ϕ, *i.e.*, a clean set $\mathcal{R}$ obtained through selecting clean samples from consistency set, a hard set $\mathcal{H}$ obtained through selecting clean samples from discrepancy set, and a purified set $\mathcal{P}$ obtained through selecting noisy samples from consistency set followed by LLMs to query. As a result, these subsets are utilized to train the co-prediction network improving its performance. Meanwhile, the co-prediction network with better performance also prompts the precise partitioning of subsets.

Algorithm 1 Pipeline of NoiseAL

Input: $\mathcal{D}$=$\{(x_i, y_i)\}_{i=1}^{N}$, $y_i \in [1, \mathcal{K}]$; small models S_s and S_w; Large Language Model L

Set epoch = 1;
while epoch $\leq$ 6 **do**
  Get two confidence P_s and P_w from S_s and S_w for all $i = 1, \ldots, N$;
  Calculate dynamic threshold $\tau(t)$;
  Divide the noisy dataset into Consistency set C and Discrepancy set I based on $\tau(t)$, P_w and P_s;
  Calculate fixed threshold ϕ;
  Based on the ϕ, further divide the C and I into Clean set R, Hard set H, and Purified set P;
  Using L to generate new labels for each sample from P;
  Train S_s and S_w with a combination of different loss functions based on the different subsets (R, H, P) as Eq.(4) and Eq.(6)
  epoch = epoch + 1
end while

Selection. For a text x_i given label y_i, the confidence from the weak and strong model can be denoted as $p_w(y; x)$ and $p_s(y; x)$, respectively. If both confidences in co-prediction network are higher than $\tau(t)$, we put x_i into the consistency set

$\mathcal{C}$:

$$\mathcal{C} = \{(x_i, y_i)|p_w(y_i; x_i) > \tau_w(t)\} \cap \{(x_i, y_i)|p_s(y_i; x_i) > \tau_s(t)\}. \quad (1)$$

That is to say, if a sample can be consistently memorized by two models with high confidence, we regard it as a consistent sample. Besides, if only one model's confidence surpass $\tau(t)$, we can include it in the discrepancy set $\mathcal{I}$:

$$\mathcal{I} = \{(x_i, y_i)|p_w(y_i; x_i) > \tau_w(t)\} \cup \{(x_i, y_i)|p_s(y_i; x_i) > \tau_s(t)\} - \mathcal{C}. \quad (2)$$

However, strong and weak models eventually memorize the noisy label, which means both models may memorize a sample with noisy labels during training, we need to distinguish further whether the sample in the consistent set $\mathcal{C}$ may be clean or noisy. To achieve it, we apply ϕ for $\mathcal{C}$ to distinguish the clean samples from noisy samples. Specifically, for a sample in consistency set $\mathcal{C}$, if its clean probability is higher than ϕ, we put it into the clean set $\mathcal{R}$: $\mathcal{R} = \{(x_i, y_i)|o(x_i) \geq \phi\} \cap \mathcal{C}$.

In short, if a sample is consistently memorized by two models and meanwhile has a high clean probability, we regard it as a clean sample. On the contrary, if a consistent sample's clean probability is lower than ϕ, we put it into $\mathcal{P}$: $\mathcal{P} = \{(x_i, y_i)|o(x_i) < \phi\} \cap \mathcal{C}$. That is to say, if a sample that is consistently memorized by two models has a low clean probability, this sample is highly likely to be a noisy sample.

Qiao et al. (2022) found that pretrained language models (PLMs) BERT memorize noisy samples at a dramatically lower rate compared to BiLSTM in noisy scenarios. When BiLSTM starts to memorize the noisy samples, its confidence for clean samples will decrease, but BERT still maintains high confidence for these clean samples. Thus, we think some samples in the remaining discrepancy set $\mathcal{I}$ may be clean and have the potential to improve our method's generalization ability. We put them into the hard set $\mathcal{H}$: $\mathcal{H} = \{(x_i, y_i)|o(x_i) \geq \phi\} \cap \mathcal{I}$.

4.3 Active Querying by LLMs

In this part, we leverage the strong ICL ability of LLMs to generate new labels for noisy samples. In particular, the core challenge lies in constructing a proper prompt containing demonstration samples. To perform active querying, we first provide an assumption as the justification:

Assumption. *Samples in $\mathcal{R}$ and $\mathcal{P}$ can be considered almost clean and noisy respectively.*

Note that the above *Assumption* is empirically verified in Table 2. The underlying intuition is: that the noisy ratio in $\mathcal{R}$ is significantly low and thus negligible, while the noisy ratio in $\mathcal{P}$ is significantly high and needs to be corrected by querying LLMs.

Demonstration Construction. We devise a feature-aware example sampling strategy. Given an input $x_i \in \mathcal{P}$, we can obtain its text feature $f(x_i)$ and calculate the cosine similarity of the text feature between x_i and other training text $x_j \in \mathcal{R}$, then sample top-*K* nearest neighbors to form demonstration examples:

$$\mathcal{E} = \underset{j \in \{1,2,\ldots,|\mathcal{R}|\}}{\text{argTopN}} \frac{f(x_i)^T f(x_j)}{\|f(x_i)\|\|f(x_j)\|}, \quad (3)$$

where $\mathcal{E}$ is an index set of top-*K* similar samples in $\mathcal{R}$. Note that text features of datasets are computed and stored beforehand, allowing efficient sampling.

Querying. Our LLMs prompt consists of the following three components: (1) *Task description*, which describes the task. (2) *Demonstration*, which consists of a sequence of samples from $\mathcal{R}$. (3) *Input*, which is the test text from $\mathcal{P}$ to classify. Our initial investigation indicates that although this prompt design can already yield reasonable results, the prediction is unstable. Specifically, we observe variations in the results when we randomly rearrange the order of the demonstrations. This indicates that the model faces difficulty comprehending the user's viewpoint accurately. To address this instability, we incorporate *Chain-of-Thought (COT)* reasoning into our setting. The details of prompts are provided in Appendix J. By prompting LLMs to generate high-quality labels for noisy samples, the high noise rate of $\mathcal{P}$ is greatly reduced.

4.4 Training Co-prediction Network (SMs)

Learning From the clean Set $\mathcal{R}$. For the $\mathcal{R}$, we directly utilize the cross-entropy loss for the co-prediction network (SMs):

$$\mathcal{L}_{\mathcal{R}} = -\frac{1}{N}\sum_{i=1}^{N_{\mathcal{R}}} \log p_s(y_i; x_i) + \log p_w(y_i; x_i) \quad (4)$$

where $N_{\mathcal{R}}$ denotes the size of $\mathcal{R}$, N denotes the size of the entire dataset.

Learning From the Purified Set $\mathcal{P}$. After generating the labels by LLMs for samples in $\mathcal{P}$, the number of noisy samples in $\mathcal{P}$ is greatly reduced. However, for LLMs, even the powerful GPT-4, cannot generate right labels for every sample in $\mathcal{P}$. Our solution comes from rethinking the way of learning from the $\mathcal{P}$: *can we design an objective such that our model can be optimized with access to the $\mathcal{P}$ with a lower noise ratio?* To this end, we resort to a family of noise-robust loss functions (ℓ_{robust}) (Ghosh et al., 2017; Zhang and Sabuncu, 2018). Some loss functions has the following property:

$$\sum_{k=1}^{\mathcal{K}} \ell_{\text{robust}}(f_\theta(x), k) = C, \forall x, f \tag{5}$$

where x is the input, and C is a constant. Previous research has shown that these loss functions have a consistent minimum under label noise. This means that the minimizer (θ^*) of ℓ_{robust} is the same when most of the training samples are correctly labeled, no matter if the training data is noisy or clean. The theoretical proof will be given in Appendix E. This robust property lets us optimize the model given a dataset with a lower noise ratio. Specifically, we utilize the reversed cross-entropy loss for sample (x_i, y_i) in $\mathcal{P}$:

$$\mathcal{L}_{\mathcal{P}} = -\frac{1}{N}\sum_{i=1}^{N_{\mathcal{P}}}\sum_{k=1}^{\mathcal{K}}(p_s(k;x_i) + p_w(k;x_i))\log q(k|x_i), \tag{6}$$

where $q(k|\boldsymbol{x})$ is the ground-truth distribution over labels, $N_{\mathcal{P}}$ denotes the size of $\mathcal{P}$.

Remark. *The reversed cross-entropy loss satisfies Property 5 and is theoretically noise-tolerant.*

The proof of *Remark* is given in Appendix F.

Learning From the Hard Set $\mathcal{H}$. Since the sample in hard set $\mathcal{H}$ cannot be memorized by two models at the same time, if we directly use cross-entropy for supervision, the model is prone to accumulate errors. Inspired by EmbMix (Qiao et al., 2022), which is an effective regularization technique that applies the [CLS] embedding encoded by PLMs in noise-robust training for text classification. Based on the [CLS] embedding encoded by PLMs, we also apply interpolations to it. To be specific, we randomly choose two samples (x_i, y_i), (x_j, y_j) and the mixed sample (e'_i, y'_i) can be defined as $e'_i = \lambda' e_i + (1-\lambda')e_j$, $y'_i = \lambda' y_i + (1-\lambda')y_j$, $e_i =$PLMs(x_i), $e_j =$PLMs(x_j), $\lambda \sim$ Beta(α, α), $\lambda' = \max(\lambda, 1-\lambda)$. Then, we perform EmbMix on $\mathcal{H}$ to obtain the mixed set $\widetilde{\mathcal{H}}$: $\widetilde{\mathcal{H}} = \{(e'_i, y'_i)|(x_i, y_i), (x_j, y_j) \in \mathcal{H}\}$. For $\widetilde{\mathcal{H}}$, we compute the loss: $\mathcal{L}_{\mathcal{H}} = -\frac{1}{N}\sum_{i=1}^{N_{\widetilde{\mathcal{H}}}} \log p_s(y'_i; e'_i)$, where $N_{\widetilde{\mathcal{H}}}$ is the size of $\widetilde{\mathcal{H}}$.

Finally, our overall training objective $\mathcal{L}$ can be calculated by: $\mathcal{L} = \mathcal{L}_{\mathcal{R}} + \mathcal{L}_{\mathcal{P}} + \mathcal{L}_{\mathcal{H}}$.

5 Experiments

5.1 Experimental Settings

Datasets. We first experiment with five datasets: 20ng (Lang, 1995), SST-2 (Socher et al., 2013), Trec (Li and Roth, 2002), AGNews (Gulli, 2005), and IMDB (Maas et al., 2011). Three different types of synthetic label noise are generated and injected into these datasets following the setups of existing LNL works (Qiao et al., 2022; Zhu et al., 2022): (1) **Symmetric Noise (S)** flips labels uniformly to other classes (van Rooyen et al., 2015) (2) **Asymmetric Noise (A)** flips their labels to the corresponding class according to the asymmetric noise transition matrix. (Chen et al., 2019; Zhu et al., 2022) (3) **Instance-dependent Noise (I)** flips origin labels to the class with the highest prediction probability (other models as the feature extractor) among other classes (Algan and Ulusoy, 2020). Then, we conduct experiments on datasets with (4) **Real-world Noise**: TREC (Awasthi et al., 2020), ChemProt (Krallinger et al., 2017), SemEval (Zhou et al., 2020), and so on. Details are in Appendix B.

Baselines. We compare NoiseAL with the most relevant LNL baselines as follows: (1) *Basic models* without Noise-handling (Devlin et al., 2019); (2) *Regularization Technology*, including **SCE** (Wang et al., 2019), **ELR** (Liu et al., 2020), **CL** (Northcutt et al., 2021b), **DyGen** (Zhuang et al., 2023); (3) *Sample Selection*, including **Co-Teaching** (Han et al., 2018), **SelfMix** (Qiao et al., 2022), **LAFT** (Wang et al., 2023). See Appendix C for details. For these baselines, we perform their public code (except **LAFT**) to implement them.

LLMs and Prompts We use GPT-3.5-Tubor-0613 API (e.g., ChatGPT), and run the generation 5 times with a temperature of 0.5 to produce different reasoning paths and predictions. Then we use majority voting to get the final prediction results.

The implementation details are in Appendix D.

5.2 Main Results

Table 1 and Table 3 show the main results for three synthetic and real-world noisy datasets (See Appendix G for more datasets' results). From these results, we have the following observations: (1) NoiseAL significantly outperforms all baselines on synthetic datasets with varying noise types and ratios. (2) Instance-dependent Noise (I) is feature-dependent noise, which is more challenging than other synthetic noise. The results in Table 1 show that NoiseAL still outperforms other methods un-

Dataset	Trec				AGNews						IMDB					
Method(↓) / Noise(→)	**20%S**	**40%S**	**20%A**	**40%A**	**20%S**	**40%S**	**20%A**	**40%A**	**20%I**	**40%I**	**20%S**	**40%S**	**20%A**	**40%A**	**20%I**	**40%I**
BERT	94.64	87.45	93.60	85.72	90.68	84.43	90.27	84.30	88.24	85.72	84.44	64.92	84.83	63.78	86.28	74.66
Co-Teaching	95.08	89.30	94.88	87.16	92.03	88.41	92.12	89.38	89.53	88.72	90.04	84.48	89.93	84.64	88.94	77.62
SCE	95.23	89.10	94.12	87.30	91.66	88.22	91.88	89.52	89.68	89.37	90.66	83.58	90.89	82.81	88.32	77.05
ELR	94.92	89.28	95.24	86.90	92.01	88.22	91.88	89.52	89.68	89.37	90.81	82.67	90.64	82.90	88.42	76.42
CL	95.64	89.72	95.52	86.24	92.17	88.45	92.30	89.13	89.94	87.03	86.85	84.66	86.74	84.39	89.08	77.99
SelfMix	95.20	89.80	95.16	89.00	91.37	89.28	91.21	87.80	88.32	87.45	89.10	87.12	89.13	86.11	87.31	82.44
DyGen	95.88	89.00	94.96	88.56	91.61	89.88	91.59	86.62	89.15	87.72	86.53	71.18	86.58	72.23	86.46	75.56
Supervised GT (0% Noise)	97.20				94.05						92.98					
ChatGPT (Zero-shot)	61.60				82.92						90.76					
NoiseAL (Ours)	**97.16**	**96.40**	**96.80**	**95.80**	**93.92**	**93.05**	**93.85**	**93.07**	**93.68**	**92.70**	**92.78**	**91.71**	**92.78**	**91.73**	**92.76**	**90.23**

Table 1: Performance (accuracy %) comparison of NoisyAL with other LNL basslines on synthetic noise datasets. Moreover, we also compare NoisyAL with the zero-shot and supervised counterparts on the test dataset. Supervised GT refers to BERT trained on ground truth data. **Bold** means the best score for each dataset.

Dataset	Subsets	epoch3	epoch4	epoch5	epoch6
Trec	Clean set ($\mathcal{R}$)	2933/ 2919	2817/ 2798	3112/ 3095	3167/ 3145
	Purified set ($\mathcal{P}$)	889/ 159 ⇒ 889/ 567 (63.78%)	916/ 204 ⇒ 916/ 816 (89.08%)	929/ 148 ⇒ 929/ 893 (96.12%)	915/ 133⇒915/ 907 (99.13%)
	Hard set ($\mathcal{H}$)	1345/ 1077	1236/ 989	1229/ 996	1219/ 982
STT-2	Clean set ($\mathcal{R}$)	2316/ 2210	2505/ 2397	2589/ 2479	2632/ 2517
	Purified set ($\mathcal{P}$)	852/ 346 ⇒ 852/ 834 (97.89%)	770/ 257 ⇒ 770/ 719 (93.38%)	739/ 115 ⇒ 739/712 (96.35%)	682/ 170⇒ 682/ 673 (98.68%)
	Hard set ($\mathcal{H}$)	2803/ 2647	2796/ 2662	2854/ 2688	2906/ 2739

Table 2: The data statistical distribution (the number of all samples / the number of samples with correct labels) of different subsets on the Trec dataset and SST-2 dataset under 20% asymmetric label noise. The left part of ⇒ represents the original data distribution of the purified set, and the right part of ⇒ represents the data distribution after querying LLMs. The value (%) in brackets represents the ratio of correct labels in the subset.

Method	ChemProt	TREC	SEMEVAL
Noise Ratio	22.88%	38.56%	16.00%
Base	$64.84_{\pm 0.28}$	$67.33_{\pm 0.83}$	$71.44_{\pm 0.10}$
Co-Teaching	$65.98_{\pm 0.63}$	$66.61_{\pm 0.35}$	$72.07_{\pm 0.76}$
SCE	$65.91_{\pm 0.24}$	$68.12_{\pm 0.61}$	$74.83_{\pm 2.29}$
ELR	$65.90_{\pm 0.26}$	$70.32_{\pm 1.16}$	$71.53_{\pm 2.08}$
CL	$65.95_{\pm 0.28}$	$71.16_{\pm 0.61}$	$73.63_{\pm 0.58}$
SelfMix	$65.44_{\pm 0.55}$	$69.96_{\pm 2.16}$	$74.24_{\pm 3.01}$
DyGen	$69.07_{\pm 0.38}$	$72.39_{\pm 0.82}$	$73.17_{\pm 0.29}$
Ours	$\mathbf{70.98}_{\pm 0.46}$	$\mathbf{78.84}_{\pm 1.82}$	$\mathbf{81.47}_{\pm 0.62}$

Table 3: Main results on real-world noise datasets

Modules					Trec			
CN	**DS**	$\mathcal{H}$	$\mathcal{P}$	**R**	**20%S**	**40%S**	**20%A**	**40%A**
✔	✔	✔	✔	✔	**97.16**	**96.40**	**96.80**	**95.80**
✗	✔	✔	✔	✔	95.20	90.00	84.84	72.36
✔	✗	✔	✔	✔	94.80	87.80	83.96	67.14
✔	✔	✗	✔	✔	95.80	94.60	86.43	82.64
✔	✔	✔	✗	✔	95.20	93.40	88.30	79.89
✔	✔	✔	✔	✗	95.80	95.40	90.22	88.11

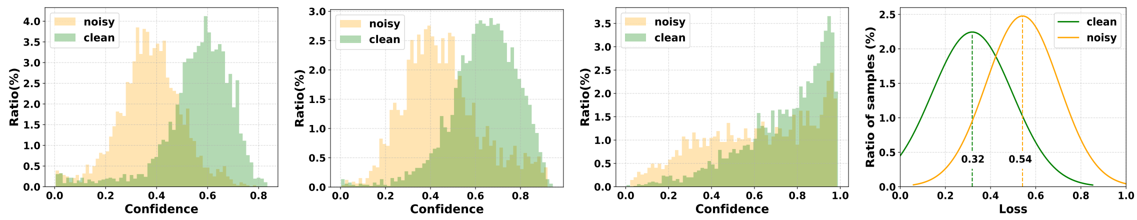
Table 4: Ablation study on the Trec dataset.

der this label noise setting, which shows NoiseAL has stronger robustness and generalization ability compared with others. (3) We also provide the base model's performance on the ground truth data (upper bounds) and the performance of ChatGPT (zero-shot). On simple datasets with fewer label categories, ChatGPT performs better than certain baselines, yet our method still maintains a certain advantage. For upper bounds, the results of NoiseAL are closest to it compared to others.

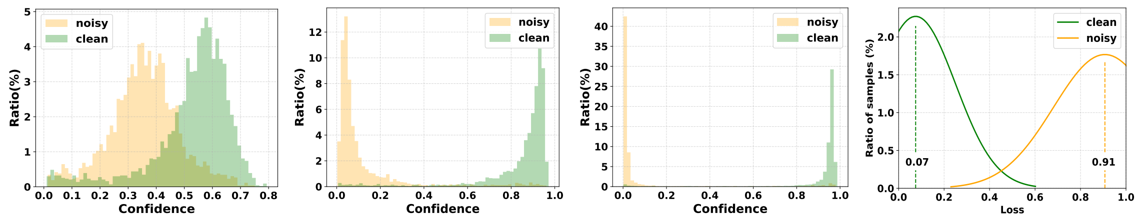
5.3 Ablation Studies

To evaluate the contribution of each component in our NoiseAL, we conduct ablation studies on the Trec dataset (refer to Table 4). More ablation studies on other datasets are shown in Appendix H.

Co-prediction Network. Co-prediction Network (CN) provides different memorization strengths to fit datasets, which can alleviate the overfitting of noisy labels and confirmation bias during training. The performance of NoiseAL will decrease greatly when we remove the CN. These results indicate that utilizing a co-prediction network is crucial, particularly in scenarios with substantial label noise. The details for the selection of strong



(a) Epoch2 (Warmup/Base) (b) Epoch4 (Base) (c) Epoch6 (Base) (d) Epoch2 (Loss/Ours)



(e) Epoch2 (Warmup/Ours) (f) Epoch4 (Ours) (g) Epoch6 (Ours) (h) Epoch6 (Loss/Ours)

Figure 4: The confidence distributions (a-c, e-g) and loss distributions (d,h) of Base/Ours on Trec under 40% asymmetric noise in different stages. We observe that the base model (a-c) gradually overfits the noisy sample, while our method (e-g) keeps learning from clean samples effectively and eventually avoids fitting noisy samples.

and weak models are in Appendix H.2.

Dynamic-enhanced Selection. Dynamic-enhanced Selection (DS) utilizes the distinct hints provided by the dynamic-enhanced threshold strategy to partition the noisy dataset. If we remove DS and only rely on a fixed threshold strategy for selection, the performance of NoiseAL will decrease by a large margin. From the overall results, we further found that the performance degradation is the most significant if DS or CN are removed, which proves these two modules contribute the most to performance improvement.

Different Subsets. To better exploit the useful information in noisy data, we divide noisy data into $\mathcal{R}$, $\mathcal{P}$, and $\mathcal{H}$. The results in Table 4 demonstrate that these subsets are all important for improving the performance of our NoiseAL. Failure to employ them leads to a decline in the results. For $\mathcal{P}$, we feed its noisy samples to LLMs for query and generate new labels, and then we adopt the reversed cross-entropy loss to learn from them. More ablation studies in Appendix H.1.

5.4 Analysis

Effect of In-Context Examples. We explore the effect of different numbers N of examples during the process of ICL on the Trec datasets. Moreover, we conduct comparison experiments by replacing the ICL with much simpler approaches such as Knn. As shown in Figure 5, our approach performs better than KNN over a wide range of N from 1 to 20, which verifies the effectiveness of NoisyAL. In our experiments, we adopt N=5 for fair comparisons.

Effect of Collaborative Training. From a more nuanced perspective, we report the statistical distribution of the different subsets during training. After warming up the model for 2 epochs, the SMs begin to divide the noisy data according to the previously generated confidences and loss distributions. From the Table 2, we find the number of samples with correct labels in purified sets $\mathcal{P}$ has greatly increased after querying LLMs, which verifies the effectiveness of querying LLMs for label denoising. During training, we observe that: (1) for $\mathcal{R}$ and $\mathcal{H}$, the number of samples with correct labels in these two subsets is gradually increasing; (2) for $\mathcal{P}$, the ratio of correct labels provided by LLMs is also gradually increasing. This observation proves our collaborative training can enable SMs and LLMs to promote and improve their performance mutually.

5.5 Capability of Distinguishing Noisy Samples

For clean samples and noisy samples, we demonstrate their confidence distributions of the base model (Figure 4(a-c)) and strong model (Figure 4(e-g)) on Trec under 40% asymmetric noise. During training, the confidence generated by the strong model is getting more polarized while the base model has already overfitted the wrong labels. Furthermore, we demonstrate the loss distributions of the strong model (Figure 4(d,h)). During training, the loss output by the strong model is also getting more polarized while the loss of the base model (as

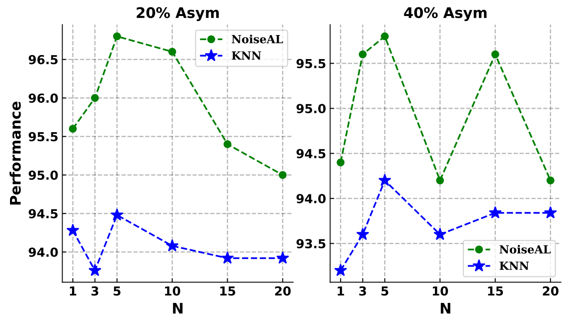


Figure 5: Comparisions of NoiseAL with KNN on Trec dataset under 20% and 40% asymmetric label noise.

shown in Figure 2) is gradually decreasing (loss distributions of clean and noisy samples are becoming consistent). That is to say, the base model eventually memorizes the noisy samples. These fine-grained experiments and visualization show that NoiseAL can better isolate noisy samples and clean samples, which can help the strong model better perform fine-tuning on clean samples and avoid memorizing (overfitting) noisy samples.

5.6 Comparison with Human Annotations

We discuss human and LLM annotations from two perspectives: expense and label quality. In terms of the expense of human annotations and LLM annotations, human annotation tasks can be conducted by crowd-workers on platforms such as MTurk and trained annotators, such as research assistants. However, LLM annotations are generally less expensive than human annotations when dealing with large datasets (For instance, AGNews, which has 127600 samples; and IMDB, which has 50000 samples). This is primarily because LLMs can generate annotations quickly and at scale, without the need for extensive human labor. Moreover, the per-annotation cost of ChatGPT is less than \$0.003—about twenty times cheaper than MTurk. But, this cost advantage must be weighed against the potential decrease in label quality. While human annotations can potentially provide a cleaner set of labels, the costs associated can be significantly higher, especially for large-scale datasets. Human annotation requires time, financial resources for compensating annotators, and sometimes additional rounds of validation to ensure quality. However, the benefits include higher accuracy and potentially fewer errors, which can be crucial for certain tasks.

Since our goal is to reduce the noise rate in the original noisy subset $\mathcal{P}$, that is, to generate a subset with a lower noise rate rather than to produce completely clean data. Therefore, the LLM annotations could be considered a more cost-effective approach, even though they might introduce some noise. For these noises, we utilize the robust loss function to learn from them and, to some extent, mitigate the negative impact of noises. Moreover, some previous work shows that the zero-shot accuracy of ChatGPT exceeds that of crowd-workers for some tasks (Gilardi et al., 2023; Törnberg, 2023).

Overall, in terms of cost and accuracy, as well as our goals, we believe that LLM annotation has more advantages than human annotation.

5.7 Case Analyse for Cost Efficiency

When it comes to leveraging pretrained knowledge, it's also essential to analyze cost efficiency (Aguda et al., 2024). In this paper, we incur costs by using ChatGPT. Thus, we provide a single case to calculate how many dollars need to be expensed by multiplying the total number of consumed tokens with the ChatGPT model price (\$0.003 per thousand tokens). Specifically, for the Trec dataset with 20% symmetric label noise (20%S), all samples (Task descriptions+Demonstration+Inputs) to query ChatGPT were tokenized into 19652 tokens, and ChatGPT generated 6338 tokens, resulting in a total of 25990 tokens. Since we run each case using 5 random seeds and report the average performance, the final tokens are 25990*5=129950 tokens, so we need to spend \$0.38985.

6 Conclusion

In this paper, we focus on solving a challenging text classification problem with noisy labels. We propose a novel framework called NoisyAL that introduces active learning to combine the non-pretrained model (BiLSTM), PLMs (BERT), and LLMs for learning from noisy labels. The key idea of NoisyAL is to separate noisy datasets by SMs (BiLSTM and BERT) and distill related knowledge from LLMs with a collaborative framework, where SMs act as filters to select clean and noisy samples for LLMs and the LLM is employed as an active annotator. The results of our experiments demonstrate the versatility of our method, as it significantly improves the accuracy of benchmark datasets with both synthetic and real-world noise. We hope that our work can inspire people's interest in developing new active denoising algorithms by collaboratively utilizing LLMs and SMs.

Limitations

We proposed a collaborative framework NoiseAL to handle label noise for multi-class text classification. Although the proposed approach outperforms the baselines by a large margin, there is still much room to improve. One promising direction is to generalize the NoiseAL to multi-label classification or hierarchical text classification problems, which needs to consider the dependence relation and hierarchical structure of labels. To capture such label information and obtain more fruitful label features, we could introduce a powerful label encoder, for example, by applying GCN, GAT, or Graphormer in NoiseAL. However, when we introduce these complex network structures, the computational cost will increase accordingly, which is also our main obstacle to pursuing this front. We plan to conduct such experiments in the future when we have access to better computing resources.

Ethics Statement

This paper will not pose ethical problems or negative social consequences. The datasets used in our paper are all publicly available and are widely adopted by researchers to evaluate models. We do not introduce extra bias or potential bias in the data compared to vanilla/other methods.

Acknowledgments

This work is supported by the China Knowledge Centre for Engineering Sciences and Technology (CKCEST-2022-1-7).

References

Toyin D. Aguda, Suchetha Siddagangappa, Elena Kochkina, Simerjot Kaur, Dongsheng Wang, and Charese Smiley. 2024. Large language models as financial data annotators: A study on effectiveness and efficiency. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 10124–10145. ELRA and ICCL.

Görkem Algan and Ilkay Ulusoy. 2020. Label noise types and their effects on deep learning. *CoRR*, abs/2003.10471.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

HeeSun Bae, Seungjae Shin, Byeonghu Na, JoonHo Jang, Kyungwoo Song, and Il-Chul Moon. 2022. From noisy prediction to true label: Noisy prediction calibration via generative model. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 1277–1297. PMLR.

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. *CoRR*, abs/2302.04023.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium, USENIX Security 2019, Santa Clara, CA, USA, August 14-16, 2019*, pages 267–284. USENIX Association.

Pengfei Chen, Benben Liao, Guangyong Chen, and Shengyu Zhang. 2019. Understanding and utilizing deep neural networks trained with noisy labels. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 1062–1070. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

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. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Aritra Ghosh, Himanshu Kumar, and P. S. Sastry. 2017. Robust loss functions under label noise for deep neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 1919–1925. AAAI Press.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. 2023. Chatgpt outperforms crowd-workers for text-annotation tasks. *CoRR*, abs/2303.15056.

Antonio Gulli. 2005. The anatomy of a news search engine. In *Proceedings of the 14th international conference on World Wide Web, WWW 2005, Chiba, Japan, May 10-14, 2005 - Special interest tracks and posters*, pages 880–881. ACM.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8536–8546.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Martin Krallinger, Obdulia Rabal, Saber A Akhondi, Martın Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio López, Umesh Nandal, et al. 2017. Overview of the biocreative vi chemical-protein interaction track. In *Proceedings of the sixth BioCreative challenge evaluation workshop*, volume 1, pages 141–146.

Ken Lang. 1995. Newsweeder: Learning to filter netnews. In *Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, California, USA, July 9-12, 1995*, pages 331–339. Morgan Kaufmann.

Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. Dividemix: Learning with noisy labels as semi-supervised learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. 2017. Webvision database: Visual learning and understanding from web data. *CoRR*, abs/1708.02862.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *19th International Conference on Computational Linguistics, COLING 2002, Howard International House and Academia Sinica, Taipei, Taiwan, August 24 - September 1, 2002*.

Yifan Li, Hu Han, Shiguang Shan, and Xilin Chen. 2023. Disc: Learning from noisy labels via dynamic instance-specific selection and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24070–24079.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. 2020. Early-learning regularization prevents memorization of noisy labels. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xingjun Ma, Hanxun Huang, Yisen Wang, Simone Romano, Sarah M. Erfani, and James Bailey. 2020. Normalized loss functions for deep learning with noisy labels. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 6543–6553. PMLR.

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 *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 142–150. The Association for Computer Linguistics.

Costas Mavromatis, Balasubramaniam Srinivasan, Zhengyuan Shen, Jiani Zhang, Huzefa Rangwala, Christos Faloutsos, and George Karypis. 2023. Which examples to annotate for in-context learning? towards effective and efficient selection. *CoRR*, abs/2310.20046.

Tom Mitchell. 1999. Twenty Newsgroups. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5C323.

Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. 2021a. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. 2021b. Confident learning: Estimating uncertainty in dataset labels. *J. Artif. Intell. Res.*, 70:1373–1411.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2233–2241. IEEE Computer Society.

Dan Qiao, Chenchen Dai, Yuyang Ding, Juntao Li, Qiang Chen, Wenliang Chen, and Min Zhang. 2022. Selfmix: Robust learning against textual label noise with self-mixup training. In *Proceedings of the 29th International Conference on Computational Linguistics, COLING 2022, Gyeongju, Republic of Korea, October 12-17, 2022*, pages 960–970. International Committee on Computational Linguistics.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. 2019. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing*

Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada, pages 1917–1928.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. ACL.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Xiaofei Sun, Xiaoya Li, Jiwei Li, Fei Wu, Shangwei Guo, Tianwei Zhang, and Guoyin Wang. 2023. Text classification via large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 8990–9005. Association for Computational Linguistics.

Michael Tänzer, Sebastian Ruder, and Marek Rei. 2021. BERT memorisation and pitfalls in low-resource scenarios. *CoRR*, abs/2105.00828.

Petter Törnberg. 2023. Chatgpt-4 outperforms experts and crowd workers in annotating political twitter messages with zero-shot learning. *CoRR*, abs/2304.06588.

Brendan van Rooyen, Aditya Krishna Menon, and Robert C. Williamson. 2015. Learning with symmetric label noise: The importance of being unhinged. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 10–18.

Song Wang, Zhen Tan, Ruocheng Guo, and Jundong Li. 2023. Noise-robust fine-tuning of pretrained language models via external guidance. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 12528–12540. Association for Computational Linguistics.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019. Symmetric cross entropy for robust learning with noisy labels. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 322–330. IEEE.

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 *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023, Singapore, December 6-10, 2023*, pages 14520–14535. Association for Computational Linguistics.

Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. 2019. L_dmi: A novel information-theoretic loss function for training deep nets robust to label noise. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 6222–6233.

Yan Yan, Rómer Rosales, Glenn Fung, Subramanian Ramanathan, and Jennifer G. Dy. 2014. Learning from multiple annotators with varying expertise. *Mach. Learn.*, 95(3):291–327.

Taraneh Younesian, Zilong Zhao, Amirmasoud Ghiassi, Robert Birke, and Lydia Y. Chen. 2021. Qactor: Active learning on noisy labels. In *Asian Conference on Machine Learning, ACML 2021, 17-19 November 2021, Virtual Event*, volume 157 of *Proceedings of Machine Learning Research*, pages 548–563. PMLR.

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

Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. 2018. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. 2021b. WRENCH: A comprehensive benchmark for weak supervision. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*.

Yivan Zhang, Gang Niu, and Masashi Sugiyama. 2021c. Learning noise transition matrix from only noisy labels via total variation regularization. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 12501–12512. PMLR.

Zhilu Zhang and Mert R. Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8792–8802.

Liyue Zhao, Gita Sukthankar, and Rahul Sukthankar. 2011. Incremental relabeling for active learning with noisy crowdsourced annotations. In *PASSAT/SocialCom 2011, Privacy, Security, Risk and Trust (PASSAT), 2011 IEEE Third International Conference on and 2011 IEEE Third International Conference on Social Computing (SocialCom), Boston, MA, USA, 9-11 Oct., 2011*, pages 728–733. IEEE Computer Society.

Wenxuan Zhou, Hongtao Lin, Bill Yuchen Lin, Ziqi Wang, Junyi Du, Leonardo Neves, and Xiang Ren. 2020. NERO: A neural rule grounding framework for label-efficient relation extraction. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2166–2176. ACM / IW3C2.

Dawei Zhu, Michael A. Hedderich, Fangzhou Zhai, David Ifeoluwa Adelani, and Dietrich Klakow. 2022. Is BERT robust to label noise? A study on learning with noisy labels in text classification. In *Proceedings of the Third Workshop on Insights from Negative Results in NLP, Insights@ACL 2022, Dublin, Ireland, May 26, 2022*, pages 62–67. Association for Computational Linguistics.

Yuchen Zhuang, Yue Yu, Lingkai Kong, Xiang Chen, and Chao Zhang. 2023. Dygen: Learning from noisy labels via dynamics-enhanced generative modeling. *CoRR*, abs/2305.19395.

A Analysis for Loss-Value-Based Method

Recent studies (Zhang et al., 2021a) have illuminated that during SMs (such as non-pretrained neural network models and pretrained language models) training, the loss distributions of clean and noisy samples typically adhere to two Gaussian Distributions. Notably, clean samples tend to have a smaller mean loss value, as shown in Figure 2. While some research based on SM ($e.g.$ Bert), such as SelfMix, has shown promise in leveraging this insight to separate noisy datasets(Qiao et al., 2022), a recent study (Gao et al., 2023) indicates that these fixed-loss-value-based methods fail to help select clean samples in some scenarios. We further empirically analyze why the popular fixed-loss-value-based methods, such as SelfMix, do not perform well in noisy scenarios (Qiao et al., 2022). For this experiment, we generate two types of synthetic label noise with different ratios (20% asymmetric, 40% asymmetric, 20% symmetric, 40% symmetric), and inject them into SST-2 datasets. As shown in Figure 8, the loss values of the correctly labeled data and the mislabelled data still coincided together during the training process. Then, we observe that the mean loss value gradually decreases during training, as shown in Figure 9.

From the above observation, it can be inferred that the adoption of a uniform fixed threshold throughout training is less than ideal, particularly when we consider SMs' inherent memorization effect that the memorization strength of samples increases during training (See Figure 4 (a-c)). Our stance is that an intricate interplay exists between memory effects and loss distributions, and that these loss distributions should dynamically evolve during training. Therefore, it is not appropriate to set a fixed loss threshold.

A.1 Comparison of Dividing Dataset Capabilities

From a more nuanced perspective, we assess the capability of SelfMix and our NoiseAL to divide noisy datasets. Specifically, we report the statistical distribution of the different subsets on the Trec dataset under 20% asymmetric label noise (Table 2). According to the results in Table 2, it was observed that NoiseAL outperforms SelfMix in accurately selecting clean samples with correct labels. This ability of NoiseAL is particularly beneficial in constructing demonstrations for LLMs by utilizing clean subsets. Moreover, the subsets divided by SelfMix are coarse-grained sets, indicating that SelfMix fails to separate clean and noisy samples accurately. Therefore, these fixed-loss-value-based methods are not applicable in our scenario.

B Dataset Details

In this work, we select **20ng**, **SST-2**, **Trec**, **AG-News** and **IMDB** for experiments on synthetic noise datasets. For these datasets, we explain the details of synthetic noise generation processes in the following:

Asymmetric noise (Asym) Asymmetric noise attempts to simulate the incorrect classification of classes. Modeling such noise can be achieved by flipping the labels of the samples according to a pre-defined noise level $\varepsilon \in [0, 1)$ (Zhu et al., 2022):

$$p_{flip}(\hat{y} = j|y = i) = \begin{cases} 1 - \varepsilon, & i = j \\ \varepsilon, & i \neq j \end{cases} .$$

Due to these noise generation processes are feature independent ($i.e.$ $p(\cdot|y = i, x) = p(\cdot|y = i)$), we describe them by an asymmetric noise transition matrix, which can be used to generate noisy labels.

Symmetric noise (Sym) Modeling such noise can be achieved by uniformly flipping the labels of the samples to other classes according to a predefined noise level $\varepsilon \in [0, 1)$ (van Rooyen et al., 2015):

$$p_{flip}(\hat{y} = j|y = i) = \begin{cases} 1 - \varepsilon, & i = j \\ \frac{\varepsilon}{\mathcal{K}-1}, & i \neq j \end{cases} ,$$

where $\mathcal{K}$ is the number of classes.

Instance-dependent noise (IDN) We follow the noise generation process in existing literature (Bae et al., 2022; Qiao et al., 2022) for IDN generation in our experiments. The **Trec** dataset comprises only 5452 training samples and exhibits a significant class imbalance. Consequently, when considering a high noise ratio, there's a possibility that the count of clean samples might be lower than that of generated noisy samples within the long-tailed class. This circumstance renders the classification task meaningless. As a result, we exclusively generate IDN on other datasets expect Trec. The detailed algorithm of IDN noisy label generation is summarized in Algorithm 2.

Regarding the **real-world noise**, we follow the work in DyGen (Zhuang et al., 2023) to carry out extensive experiments on three real-world noisy datasets: **TREC**, **ChemProt**, and **SemEval**. Specifically, TREC is a question classification

Models	Subsets	epoch3	epoch4	epoch5	epoch6
NoiseAL	Clean	2933/2919(99.52%)	2817/2798(99.33%)	3112/3095(99.45%)	3167/3145(99.31%)
	Purified	889/159(17.89%)	916/204(22.27%)	929/148(15.93%)	915/133(14.54%)
	Hard	1345/1077(80.07%)	1236/ 989(80.02%)	1229/996(81.04%)	1219/982(80.56%)
SelfMix	Clean	3349/2709(80.89%)	3165/2559(80.85%)	3178/2577(81.09%)	3191/2587(81.07%)
	Unclean	2103/1675(79.65%)	2287/1825(79.80%)	2274/1807(79.46%)	2261/1797(79.48%)

Table 5: The data statistical distribution (the number of all samples / the number of samples with correct labels) of different subsets on the Trec dataset under 20% asymmetric label noise. The value (%) in brackets represents the ratio of correct labels in the subset.

with 6 classes in the weak supervision benchmark (Zhang et al., 2021b); ChemProt is a chemical-protein interaction dataset with 10 classes; and SemEval is a relation extraction dataset with 9 classes. For these three datasets, we use the pre-defined heuristic rules from prior work DyGen (Zhuang et al., 2023) as weak supervision to obtain noisy labels.

Then, we conduct experiments on "real data". Specifically, a work (Northcutt et al., 2021a) identifies label errors in the *test* sets of 10 of the most commonly-used computer vision, natural language, and audio datasets. Further, we viewed [1] the test set errors across the natural language datasets (20news and IMDB datasets) and reproduced the label errors of IMDB datasets and 20news dataset (Mitchell, 1999) by their public code [2]. For these label errors, we remove them from the *test* set and place them in the original training set, constructing a dataset that naturally contains noise (**real-noisy datasets**). The noisy labels in real-noisy datasets are human-validated via crowdsourcing, which is different from those generated or label-flipped label errors. Strictly speaking, the label errors in real-noisy datasets are the noise of the real world, without any flips and rules. We compare our method NoiseAL with other baselines on real-noisy 20news dataset and IMDB dataset. From the results in Table 7, we found that there isn't a significant difference in performance among the various methods, which is due to the low proportion of noise (the label errors ratio in 20new test sets is 1.09% and the label errors ratio in IMDB test sets is 3.90%). However, our method still outperforms the others, demonstrating its effectiveness in scenarios with real noise.

Table 6 introduces detailed statistics about all datasets used in our experiments.

#Dataset	#Class	#Training	#Validation	#Test
Trec	6	4952	500	500
20ng	20	9051	7527	2263
AGNews	4	112400	7600	7600
SST-2	2	5099	1820	1820
IMDB	2	40000	5000	5000
TREC	6	4965	500	500
SemEval	9	1749	692	200
ChemProt	10	12861	1607	1607

Table 6: The detailed statistics of all datasets used in our experiments.

[1] https://labelerrors.com

[2] https://github.com/cleanlab/label-errors

C Baselines Details

We compare with the most relevant state-of-the-art baselines on learning with noisy labels, including: (1) **BERT/Base** (Devlin et al., 2019) We train the BERT (base model) model fine-tuned only with standard cross-entropy loss without noise-handling; (2) **Co-Teaching** (Han et al., 2018) concurrently develop two deep neural networks, and allow them to instruct one another using each mini-batch; (3) **SCE** (Wang et al., 2019) propose Reverse Cross Entropy to boost Cross Entropy symmetrically with a noise-robust learning; (4) **ELR** (Liu et al., 2020) designs a regularization term that steers the model implicitly forget the false labels; (5) **CL** (Northcutt et al., 2021b) employ confidence learning to quantify ontological class overlap and moderately increase model accuracy by cleaning data prior to training; (6) **SelfMix** (Qiao et al., 2022) separates samples via GMM and leverages semi-supervised learning to handle label noise; (7) **DyGen** (Zhuang et al., 2023) uses the variational auto-encoding framework to infer the posterior distributions of true labels from noisy labels to improve noisy label predictions; (8) **LAFT** (Wang et al., 2023) also segregates all training samples into different subsets by generating confidences for each sample of training

Algorithm 2 Instance Dependent Noise Generation

Input: Clean samples $(x_i, y_i)_{i=1}^n$, $y_i \in [1, \mathcal{K}]$; Noisy ratio τ;

Train an LSTM classifier f;
Get output from an LSTM classifier $f_{x_i} \in \mathbb{R}^{\mathcal{K}}$ for all $i = 1, \ldots, n$;
Set $N_{noisy} = 0$;
while $N_{noisy} < n \times \tau$ **do**
  Randomly choose a sample x_i, $argmax(softmax(f_{x_i})) \neq y_i$;
  set its noisy label $\bar{y}_i = argmax(softmax(f_{x_i}))$;
  $N_{noisy} = N_{noisy} + 1$;
end while

Output: Noise samples $(x_i, \bar{y}_i)_{i=1}^n$;

Noise	**real-noisy**							
Datasets(↓) / **Models**(→)	BERT	Co-Teaching	SCE	ELR	CL	SelfMix	DyGen	Ours
20news	80.06	80.43	80.42	80.71	80.83	80.16	80.50	**81.37**
IMDB	93.24	93.94	93.74	93.52	93.56	93.62	94.05	**94.52**

Table 7: Comparisions (accuracy %) of NoiseAL with other baselines on 20news datasets and IMDB dataset under real label noise. **Bold** means the best score for each dataset.

datasets, which is a way that introduces the external guidance from LLMs. Although the segregation method based on confidence is similar to ours, LAFT ignores the inaccurate of LLM-generated confidences. Compared to LAFT, our methods only utilize the LLM on one subset, which can reduce the cost of LLM expenses. Then, we apply the noise-robust loss functions on LLM-generated labels, which can avoid the additional biases introduced by inaccurate results from LLMs. So our method is more efficient and effective than LAFT.

D Implementation Details

All experiments are evaluated using accuracy on a clean test set, and the reported test performance is selected according to the performance on a clean development set. This applies to both NoiseAL and all baselines. We report the average performance as well as standard deviations using 5 random seeds. We implement our framework with Python 3.7, PyTorch 1.13, and HuggingFace, and train our framework on Nvidia RTX 3090 and Nvidia A100 GPU. In addition, we use Adam (Kingma and Ba, 2015) as an optimizer. In the main experiments, we choose the BERT as the backbone model for NoiseAL and all baseline methods. Further, we verify the generalization of NoiseAL across different PLMs in Appendix I.

Table 8 shows the detailed hyperparameter configuration. The selection of hyperparameters, especially λ, is not trivial. In our experiment, for the small dataset (trec and TREC), we select λ from $[0.95, 0.96, \ldots, 0.99]$ for strong model and select λ from $[0.1, 0.2, \ldots, 0.9]$ for the weak model; for other datasets, we select λ from $[0.95, 0.96, \ldots, 0.99]$ for strong model and select λ from $[0.5, 0.6, \ldots, 0.9]$ for weak model.

To see how λ affects the final performance of NoiseAL. The experimental results of adjusting the value of parameter λ for strong and weak models are shown in Figure 6 (Trec datasets) and Figure 7 (AGNews datasets). From these results, we find that the appropriate values λ are not entirely consistent for different datasets. This is because the data distribution of datasets of different types is complex and inconsistent. Hence, we should comprehensively consider various situations and carefully tune the value of λ according to actual datasets. Based on our experience with parameter tuning during experiments, we have the following suggestions:

Determine the range for parameter adjustment: Establishing an approximate range for parameter adjustments based on the size of the dataset. Specifically, in our experiment, for the small dataset (trec and TREC), we select λ from $[0.95, 0.96, \ldots, 0.99]$ for BERT and select λ from $[0.1, 0.2, \ldots, 0.9]$ for BiLSTM; for other datasets, we select λ from $[0.95, 0.96, \ldots, 0.99]$ for BERT and select λ from $[0.5, 0.6, \ldots, 0.9]$ for BiLSTM.

Determine the direction for parameter adjustment: After determining the range of parameter adjustments, we typically set the parameters to start experimenting from the median value of the range interval. Then, for datasets with a smaller sample size, we typically adjust the parameters in the direction of decrease, while for datasets with a larger sample size, we usually adjust the parameters in the direction of increase.

Taking BERT as an example, we first determine the adjustment range for the parameter λ to be $[0.95, 0.96, 0.97, 0.98, 0.99]$, and then we start experimenting with parameter λ at 0.97 and make adjustments from there. For datasets with a smaller sample size (such as SST-2 and Trec), we adjust the parameter λ in the direction of $[0.97, 0.96, 0.95]$; for datasets with a larger sample size (such as AgNews, 20ng and ChemProt), we adjust the parameter λ in the direction of $[0.97, 0.98, 0.99]$.

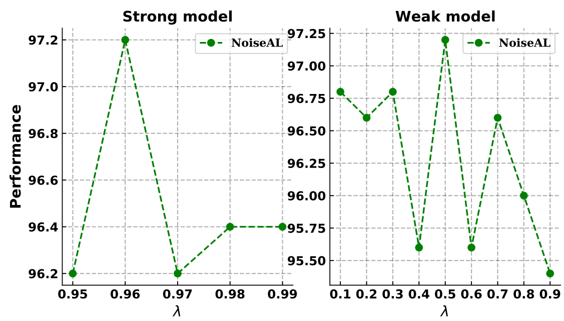


Figure 6: Performance of our NoiseAL on Trec dataset under 20% asymmetric label noise when λ is ranging from 0.95 to 0.99 for strong model and ranging from 0.1 to 0.9 for weak model

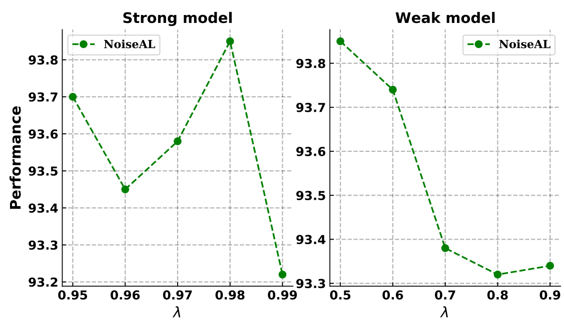


Figure 7: Performance of our NoiseAL on AGNews dataset under 20% asymmetric label noise when λ is ranging from 0.95 to 0.99 for strong model and ranging from 0.5 to 0.9 for weak model

E Theoretical Analysis

E.1 Risk Minimization problem for losses

Generally, for a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with data size N, given any loss function $\mathcal{L}$ and classifier f_θ, we define the loss on $\mathcal{D}$:

$$\begin{aligned}\mathcal{L}(f_\theta, \mathcal{D}) &= \mathbb{E}_{\mathcal{D}}[\mathcal{L}(f_\theta(x), y)] \\ &= \mathbb{E}_{x,y}[\mathcal{L}(f_\theta(x), y)].\end{aligned} \tag{7}$$

Under the risk minimization problem for losses, our object is to learn a classifier f, which is a global minimizer of $\mathcal{D}$ depending on the loss function $\mathcal{L}$. That is to say, we want to obtain the optimal parameters θ^* of f with loss function $\mathcal{L}$ over dataset $\mathcal{D}$, *i.e.*, $\theta^* = \arg\min_\theta \mathcal{L}_{robust}(f_\theta, \mathcal{D})$.

E.2 Noise Robustness of Loss Functions

Let $\mathcal{D}_{clean} = \{(x_i, y_i)\}$ represent the clean training dataset, and $\mathcal{D}_{noisy} = \{(x_i, \hat{y}_i)\}$ represent the noisy training dataset with noise rate ε, where

$$\hat{y}_i = \begin{cases} y_i, & 1-\varepsilon \\ others, & \varepsilon \end{cases}.$$

Previous work (Ghosh et al., 2017; Zhang and Sabuncu, 2018; Xu et al., 2019; Gao et al., 2023) on noise-robust loss functions has shown that the loss function satisfying formula 5 is a robust loss $\mathcal{L}_{robust}$, which has the below noise-tolerant property (Gao et al., 2023):

$$\arg\min_\theta \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{clean}) = \arg\min_\theta \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}). \tag{8}$$

E.3 Proof For Noise-tolerant Property

We include the aforementioned work here to ensure comprehensiveness. More precisely, we consider three scenarios of label noise: asymmetric noise, symmetric noise, and instance-dependent noise as described in the following.

Symmetric Noise. In a multi-class classification task with $\mathcal{K}$ classes, given a loss function $\mathcal{L}_{robust}$ satisfying property 5. Then $\mathcal{L}_{robust}$ is noise-tolerant under symmetric label noise if noise rate $\varepsilon < 1 - \frac{1}{\mathcal{K}}$, the proof as follows:

Hyperparameter	Specification	SST-2	Trec	20ng	AGNews	IMDB	TREC	ChemProt	SemEval
λ	BERT	0.96	0.96	0.97	0.98	0.97	0.97	0.99	0.99
	BiLSTM	0.8	0.5	0.6	0.5	0.9	0.3	0.8	0.9
ϕ	-	0.2	0.1	0.1	0.2	0.6	0.1	0.1	0.1
Batch Size	-	32	32	32	32	16	32	32	32
Max Length	-	256	256	256	256	512	256	256	256
Learning Rate	BERT	1e-5	1e-5	4e-5	1e-5	1e-5	1e-5	1e-5	1e-5
	BiLSTM	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02

Table 8: Main hyperparameter settings of our models in this paper.

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}) \\
&= \mathbb{E}_{x,\hat{y}}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}\mathbb{E}_{\hat{y}|y,x}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}[(1-\varepsilon)\mathcal{L}_{robust}(f_\theta(x), y) + \\
&\quad \frac{\varepsilon}{\mathcal{K}-1}\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}[(1-\varepsilon+\frac{\varepsilon}{\mathcal{K}-1}-\frac{\varepsilon}{\mathcal{K}-1}) \\
&\quad \mathcal{L}_{robust}(f_\theta(x), y) + \frac{\varepsilon}{\mathcal{K}-1}\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{x,y}[(1-\varepsilon+\frac{\varepsilon}{\mathcal{K}-1}-\frac{\varepsilon}{\mathcal{K}-1}) \\
&\quad \mathcal{L}_{robust}(f_\theta(x), y) + \frac{\varepsilon}{\mathcal{K}-1}\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{\boldsymbol{x},y}[\frac{\mathcal{K}-1-\mathcal{K}\varepsilon}{\mathcal{K}-1}\mathcal{L}_{robust}(f_\theta(x), y)] + \frac{\varepsilon C}{\mathcal{K}-1} \\
&= \frac{\mathcal{K}-1-\mathcal{K}\varepsilon}{\mathcal{K}-1}\mathcal{L}_{robust}(f_\theta, \mathcal{D}_{clean}) + \frac{\varepsilon C}{\mathcal{K}-1}
\end{aligned}
\tag{9}
$$

where C is a constant due to the property 5. Suppose θ^* is the optimal parameter of f over the clean dataset $\mathcal{D}_{clean}$, then for any θ:

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_{\theta^*}, \mathcal{D}_{noisy}) - \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}) \\
&= \frac{\mathcal{K}-1-\mathcal{K}\varepsilon}{\mathcal{K}-1} \\
&\quad (\mathcal{L}_{robust}(f_\theta{}^*, \mathcal{D}_{clean}) - \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{clean})) \\
&\leq 0.
\end{aligned}
\tag{10}
$$

Thus, when $\varepsilon < 1 - \frac{1}{\mathcal{K}}$, θ^* is also the optimal parameter of f over the noisy dataset $\mathcal{D}_{noisy}$.

Asymmetric Noise. For a loss function $\mathcal{L}_{robust}$ satisfying property 5. Then $\mathcal{L}_{robust}$ is noise-tolerant under asymmetric label noise if noise rate $\varepsilon < \frac{1}{2}$, the proof as follows:

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}) \\
&= \mathbb{E}_{x,\hat{y}}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}\mathbb{E}_{\hat{y}|y,x}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}[(1-\varepsilon)\mathcal{L}_{robust}(f_\theta(x), y) + \\
&\quad \varepsilon\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_x\mathbb{E}_{y|x}[(1-\varepsilon+\varepsilon-\varepsilon \\
&\quad \mathcal{L}_{robust}(f_\theta(x), y) + \varepsilon\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{x,y}[(1-\varepsilon+\varepsilon-\varepsilon) \\
&\quad \mathcal{L}_{robust}(f_\theta(x), y) + \varepsilon\sum_{j\neq y}\mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{\boldsymbol{x},y}[(1-2\varepsilon)\mathcal{L}_{robust}(f_\theta(x), y)] + \varepsilon C \\
&= (1-2\varepsilon)\mathcal{L}_{robust}(f_\theta, \mathcal{D}_{clean}) + \varepsilon C
\end{aligned}
\tag{11}
$$

where C is a constant due to the property 5. Suppose θ^* is the optimal parameter of f over the clean dataset $\mathcal{D}_{clean}$, then for any θ:

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_{\theta^*}, \mathcal{D}_{noisy}) - \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}) \\
&= (1-2\varepsilon) \\
&\quad (\mathcal{L}_{robust}(f_\theta{}^*, \mathcal{D}_{clean}) - \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{clean})) \\
&\leq 0.
\end{aligned}
\tag{12}
$$

Thus, when $\varepsilon < \frac{1}{2}$, θ^* is also the optimal parameter of f over the noisy dataset $\mathcal{D}_{noisy}$.

Instance-dependent Noise. For a loss function $\mathcal{L}_{robust}$ satisfying property 5 and $0 \leq \mathcal{L}_{robust}(f_\theta(x), i) \leq \frac{C}{\mathcal{K}-1}, \forall i \in [\mathcal{K}]$. Suppose θ^* is the optimal parameter of f over the clean dataset $\mathcal{D}_{clean}$ and $\mathcal{L}_{robust}(f_{\theta^*}, \mathcal{D}_{clean}) = 0$. Then $\mathcal{L}_{robust}$ is noise-tolerant under instance-dependent noise label if noise rate $\varepsilon_j < 1 - \varepsilon_{ij}, \forall j \neq i, \forall i, j \in [\mathcal{K}]$, ε_{ij} represents the probability of class i mislabeled into class j. For instance-dependent noise, we have:

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy}) \\
&= \mathbb{E}_{x,\hat{y}}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x \mathbb{E}_{y|x} \mathbb{E}_{\hat{y}|y,x}[\mathcal{L}_{robust}(f_\theta(x), \hat{y})] \\
&= \mathbb{E}_x \mathbb{E}_{y|x}[(1-\varepsilon_y)\mathcal{L}_{robust}(f_\theta(x), y) + \\
&\quad \sum_{j \neq y} \varepsilon_{yj} \mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_x \mathbb{E}_{y|x}[(1-\varepsilon_y)(C - \sum_{j \neq y} \mathcal{L}_{robust}(f_\theta(x), j)]) \\
&\quad + \sum_{j \neq y} \varepsilon_{yj} \mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{x,y}[(1-\varepsilon_y)(C - \sum_{j \neq y} \mathcal{L}_{robust}(f_\theta(x), j)]) \\
&\quad + \sum_{j \neq y} \varepsilon_{yj} \mathcal{L}_{robust}(f_\theta(x), j)] \\
&= \mathbb{E}_{x,y}(C(1-\varepsilon_y) - (1-\varepsilon_y) \sum_{j \neq y} \mathcal{L}_{robust}(f_\theta(x), j) \\
&\quad + \sum_{j \neq y} \varepsilon_{yj} \mathcal{L}_{robust}(f_\theta(x), j)) \\
&= C\mathbb{E}_{x,y}(1-\varepsilon_y) \\
&\quad - \mathbb{E}_{x,y} \sum_{j \neq y} (1 - \varepsilon_y - \varepsilon_{yj}) \mathcal{L}_{robust}(f_\theta(x), j)
\end{aligned} \tag{13}
$$

where C is a constant due to the property 5. Suppose $\theta^\dagger$ is the optimal parameter of f over the noisy dataset $\mathcal{D}_{noisy}$ and $\theta^\dagger = \arg\min_\theta \mathcal{L}_{robust}(f_\theta, \mathcal{D}_{noisy})$, then:

$$
\begin{aligned}
&\mathcal{L}_{robust}(f_{\theta^\dagger}, \mathcal{D}_{noisy}) - \mathcal{L}_{robust}(f_{\theta^*}, \mathcal{D}_{noisy}) \\
&= \mathbb{E}_{x,y} \sum_{j \neq y} (1 - \varepsilon_y - \varepsilon_{yj})(\mathcal{L}_{robust}(f_{\theta^*}(x), j) \\
&\quad - \mathcal{L}_{robust}(f_{\theta^\dagger}(x), j)) \\
&\leq 0.
\end{aligned} \tag{14}
$$

Since we are given $\mathcal{L}_{robust}(f_{\theta^*}, \mathcal{D}_{clean}) = 0$, we have $\mathcal{L}_{robust}(f_{\theta^*}(x), y) = 0$. Considering the conditions stated before, we can get that $\mathcal{L}_{robust}(f_{\theta^*}(x), i) = \frac{C}{\mathcal{K}-1}, \forall i \neq y$. If we assume that $1 - \varepsilon_y - \varepsilon_{yj} > 0$, in order the Eq. 14 to hold, we must have $\mathcal{L}_{robust}(f_{\theta^\dagger}(x), i) = \frac{C}{\mathcal{K}-1}, \forall i \neq y$, which implies $\mathcal{L}_{robust}(f_{\theta^\dagger}(x), y) = 0$ due to the symmetric property of $\mathcal{L}_{robust}$. Thus, when $\varepsilon_j < 1 - \varepsilon_{ij}$, $\theta^\dagger$ is also the optimal parameter of f over the clean dataset $\mathcal{D}_{clean}$.

F Proof For Reversed Cross-entropy Loss

Theorem. *The reversed cross-entropy loss function satisfies formula 5 and has the noisy-tolerant property 8.*

Proof. For the input x and its label y, the predicted probability of x for each label $k \in [1, \mathcal{K}]$ can be represented as $p(k; x) = \frac{e^{f_\theta(k;x)}}{\sum_{k=1}^{\mathcal{K}} e^{f_\theta(k;x)}}$. $q(k|x)$ is the ground-truth distribution over labels, and $\sum_{k=1}^{\mathcal{K}} q(k|x) = 1$. If the ground-truth label is y, then $q(y|x) = 1$ and $q(k|x) = 0$ for all $k \neq y$. Based on it, we can obtain the reversed cross-entropy loss function $\mathcal{L}_{rce}$:

$$
\begin{aligned}
&\mathcal{L}_{rce}(f_\theta(x), y) = -\sum_{k=1}^{\mathcal{K}} p(k; x) \log q(k|\boldsymbol{x}) \\
&= -p(y; x) \log q(y|\boldsymbol{x}) - \sum_{k \neq y}^{\mathcal{K}} p(k; x) \log q(k|\boldsymbol{x}) \\
&= -\sum_{k \neq y}^{\mathcal{K}} p(k; x) \log q(k|\boldsymbol{x}) \\
&= -\sum_{k \neq y}^{\mathcal{K}} p(k; x) \log(0).
\end{aligned} \tag{15}
$$

We approximate the $log(0)$ as a constant A, then $\sum_{k=1}^{\mathcal{K}} \mathcal{L}_{rce}(f_\theta(x), y) = -(\mathcal{K}-1)A$, which satisfies formula 5 and C=$-(\mathcal{K}-1)A$.

G More detailed Results

We report the detailed performance (accuracy with standard deviation %) on Trec (refer to Table 9), AGNews (refer to Table 10), IMDB (refer to Table 11), SST-2 (refer to Table 12), 20ng (refer to Table 13).

H More Ablation Experiments

To evaluate the contribution of each component in our NoiseAL, we conduct ablation studies on all datasets: Trec (refer to Table 17), IMDB (refer to Table 21), SST-2 (refer to Table 18), AGNews (refer to Table 19), 20ng (refer to Table 20).

H.1 Effect of Robust Loss Function

Due to LLMs being unable to generate correct labels for each sample in the subset $\mathcal{P}$, we utilize the reversed cross-entropy loss functions to better learn from $\mathcal{P}$ with a certain noise ratio. We conduct an ablation experiment (refer to Table 22) to verify the effectiveness of this robust loss function by replacing it with cross-entropy loss functions.

Dataset	**Trec**			
Method(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**
BERT	94.64$\pm_{0.81}$	87.45$\pm_{0.74}$	93.60$\pm_{1.30}$	85.72$\pm_{0.97}$
Co-Teaching	95.08$\pm_{0.57}$	89.30$\pm_{1.50}$	94.88$\pm_{0.53}$	87.16$\pm_{0.36}$
SCE	95.23$\pm_{0.34}$	89.10$\pm_{0.10}$	94.12$\pm_{0.79}$	87.30$\pm_{0.30}$
ELR	94.92$\pm_{0.47}$	89.28$\pm_{1.14}$	95.24$\pm_{0.45}$	86.90$\pm_{0.50}$
CL	95.64$\pm_{0.15}$	89.72$\pm_{0.81}$	95.52$\pm_{0.24}$	86.24$\pm_{4.94}$
SelfMix	95.20$\pm_{0.89}$	89.80$\pm_{1.15}$	95.16$\pm_{1.23}$	89.00$\pm_{0.86}$
DyGen	95.88$\pm_{0.32}$	89.00$\pm_{0.82}$	94.96$\pm_{0.57}$	88.56$\pm_{1.16}$
Supervised GT (0% Noise)	97.20			
ChatGPT (Zero-shot)	61.60			
Ours	**97.16**$\pm_{0.08}$	**96.40**$\pm_{0.18}$	**96.80**$\pm_{0.24}$	**95.80**$\pm_{0.15}$

Table 9: The detailed results (accuracy with standard deviation %) on Trec datasets. **Bold** means the best score.

Dataset	**AGNews**					
Method(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
BERT	90.68$\pm_{0.15}$	84.43$\pm_{1.36}$	90.27$\pm_{0.65}$	84.30$\pm_{1.90}$	88.24$\pm_{0.49}$	85.72$\pm_{0.97}$
Co-Teaching	92.03$\pm_{0.12}$	88.41$\pm_{0.26}$	92.12$\pm_{0.09}$	89.38$\pm_{1.11}$	89.53$\pm_{0.86}$	88.72$\pm_{0.14}$
SCE	91.66$\pm_{0.08}$	88.44$\pm_{0.12}$	91.76$\pm_{0.15}$	86.18$\pm_{0.42}$	89.74$\pm_{0.66}$	89.44$\pm_{0.08}$
ELR	92.01$\pm_{0.12}$	88.22$\pm_{0.24}$	91.88$\pm_{0.11}$	89.52$\pm_{0.26}$	89.68$\pm_{0.77}$	89.37$\pm_{0.11}$
CL	92.17$\pm_{0.11}$	88.45$\pm_{0.10}$	92.30$\pm_{0.11}$	89.13$\pm_{1.27}$	89.94$\pm_{0.15}$	87.03$\pm_{0.31}$
SelfMix	91.37$\pm_{0.59}$	89.28$\pm_{0.90}$	91.21$\pm_{1.26}$	87.80$\pm_{0.40}$	88.32$\pm_{0.34}$	87.45$\pm_{0.74}$
DyGen	91.61$\pm_{0.20}$	89.88$\pm_{0.31}$	91.59$\pm_{0.25}$	86.62$\pm_{0.78}$	89.15$\pm_{0.24}$	87.72$\pm_{4.95}$
Supervised GT (0% Noise)	94.05					
ChatGPT (Zero-shot)	82.92					
Ours	**93.92**$\pm_{0.07}$	**93.05**$\pm_{0.06}$	**93.85**$\pm_{0.05}$	**93.07**$\pm_{0.09}$	**93.68**$\pm_{0.08}$	**92.70**$\pm_{0.05}$

Table 10: The detailed results (accuracy with standard deviation %) on AGNews datasets. **Bold** means the best score.

H.2 Further Analysis for Co-prediction Network

In our paper, we design a co-prediction network consisting of a strong and weak model. For a strong model, pretrained language models (PLMs) might be a better choice since the whole training process can be divided into two stages, the wrong labels do not corrupt the pre-training process, which makes PLMs more robust again label noise (Qiao et al., 2022) than other traditional networks (such as BiLSTM, and Text-CNN). Therefore, we select BERT as the strong model and BiLSTM as the weak model. These two models have different fitting speeds for label noise, which can provide different predictions to avoid overfitting.

To evaluate the impact of strong model or weak model selection on performance, we conduct other approaches to reduce overfitting. To be specific, we first utilize two BERT with different dropout ratios to construct the co-prediction Network, then we select the Text-CNN as the weak model. From the

Dataset	IMDB					
Method(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
BERT	84.44$\pm_{1.69}$	64.92$\pm_{1.57}$	84.83$\pm_{0.50}$	63.78$\pm_{2.89}$	86.28$\pm_{0.84}$	74.66$\pm_{0.84}$
Co-Teaching	90.04$\pm_{0.30}$	84.48$\pm_{0.32}$	89.93$\pm_{0.44}$	84.64$\pm_{0.38}$	88.94$\pm_{0.86}$	77.62$\pm_{2.07}$
SCE	90.66$\pm_{0.55}$	83.58$\pm_{2.06}$	90.89$\pm_{0.35}$	82.81$\pm_{0.40}$	88.32$\pm_{0.55}$	77.05$\pm_{0.68}$
ELR	90.81$\pm_{0.34}$	82.67$\pm_{0.95}$	90.64$\pm_{0.67}$	82.90$\pm_{1.18}$	88.42$\pm_{0.63}$	76.42$\pm_{0.68}$
CL	86.85$\pm_{1.06}$	84.66$\pm_{0.66}$	86.74$\pm_{0.91}$	84.39$\pm_{0.53}$	89.08$\pm_{0.50}$	77.99$\pm_{1.20}$
SelfMix	89.10$\pm_{0.15}$	87.12$\pm_{2.15}$	89.13$\pm_{0.10}$	86.11$\pm_{0.93}$	87.31$\pm_{1.25}$	82.44$\pm_{3.66}$
DyGen	86.53$\pm_{0.40}$	71.18$\pm_{2.70}$	86.58$\pm_{0.60}$	72.23$\pm_{2.25}$	86.46$\pm_{0.29}$	75.56$\pm_{0.74}$
ChatGPT (Zero-shot)	90.76					
Supervised GT (0% Noise)	92.98					
Ours	**92.78**$\pm_{0.22}$	**91.71**$\pm_{0.16}$	**92.78**$\pm_{0.06}$	**91.73**$\pm_{0.10}$	**92.76**$\pm_{0.16}$	**90.23**$\pm_{0.45}$

Table 11: The detailed results (accuracy with standard deviation %) on IMDB datasets. **Bold** means the best score.

Dataset	SST-2					
Method(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
BERT	82.41$\pm_{1.71}$	67.73$\pm_{1.12}$	82.23$\pm_{0.26}$	68.01$\pm_{0.65}$	81.75$\pm_{1.68}$	66.88$\pm_{1.00}$
Co-Teaching	86.98$\pm_{0.73}$	79.94$\pm_{0.17}$	87.85$\pm_{0.41}$	79.40$\pm_{0.70}$	82.09$\pm_{1.45}$	67.57$\pm_{1.78}$
SCE	86.78$\pm_{0.37}$	73.11$\pm_{2.29}$	87.36$\pm_{1.13}$	74.85$\pm_{1.97}$	81.41$\pm_{1.42}$	67.34$\pm_{1.44}$
ELR	86.76$\pm_{0.96}$	70.58$\pm_{3.79}$	87.33$\pm_{1.12}$	74.33$\pm_{1.20}$	80.78$\pm_{1.40}$	68.49$\pm_{1.37}$
CL	86.21$\pm_{0.58}$	79.19$\pm_{0.59}$	88.74$\pm_{0.12}$	78.63$\pm_{1.10}$	82.63$\pm_{1.48}$	67.51$\pm_{1.60}$
SelfMix	83.76$\pm_{3.70}$	76.59$\pm_{0.93}$	85.36$\pm_{1.11}$	79.36$\pm_{1.44}$	82.54$\pm_{4.51}$	67.45$\pm_{3.78}$
DyGen	86.69$\pm_{1.30}$	74.27$\pm_{3.72}$	86.44$\pm_{0.40}$	72.37$\pm_{1.74}$	82.89$\pm_{1.03}$	67.76$\pm_{3.33}$
ChatGPT (Zero-shot)	90.71					
Supervised GT (0% Noise)	91.43					
Ours	**90.95**$\pm_{0.25}$	**89.34**$\pm_{0.46}$	**91.18**$\pm_{0.29}$	**89.26**$\pm_{0.34}$	**91.37**$\pm_{0.18}$	**90.08**$\pm_{0.23}$

Table 12: The detailed results (accuracy with standard deviation %) on SST-2 datasets. **Bold** means the best score.

experimental results in Table 15 (the Trec dataset) and Table 16 (the SST-2 dataset), we have the following observations: (1) the co-prediction Network consisting of BERT (dropout:0.1) and BERT (dropout:0.5) achieves the best result under 20% asymmetric label noise; (2) within the same model architecture (BERT+BERT), the larger the gap in dropout rates (0.1 and 0.9), the better the network performance; (3) eventually, the combination of different model network (BERT+LSTM) outperforms the use of a single model type (BERT+BERT). With these observations in mind, we believe that combining models with more distinct architectures can provide more diversified information to prevent confirmation bias and noise overfitting. On the whole, our selection of strong and weak models is optimal.

H.3 Further Analysis for Prompt contents

In Table 14, we ablate the prompt contents in the default settings by: (a) removing the Task description (*r.m.* a); (b) removing the Chain-of-Thought

Dataset	20ng					
Method(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
BERT	$78.79\pm_{2.51}$	$66.55\pm_{2.02}$	$75.28\pm_{2.10}$	$60.15\pm_{3.51}$	$76.07\pm_{0.96}$	$66.32\pm_{1.24}$
Co-Teaching	$76.76\pm_{1.43}$	$68.42\pm_{1.72}$	$77.04\pm_{0.85}$	$58.95\pm_{4.53}$	$77.64\pm_{0.67}$	$66.43\pm_{1.67}$
SCE	$74.61\pm_{1.96}$	$68.64\pm_{4.58}$	$75.80\pm_{2.05}$	$67.56\pm_{2.75}$	$75.03\pm_{1.92}$	$69.89\pm_{1.66}$
ELR	$80.45\pm_{0.13}$	$76.11\pm_{0.47}$	$79.26\pm_{1.22}$	$73.52\pm_{0.83}$	$78.99\pm_{0.69}$	$69.39\pm_{1.16}$
CL	$80.48\pm_{0.85}$	$77.04\pm_{0.90}$	$80.50\pm_{0.75}$	$66.98\pm_{3.93}$	$79.35\pm_{0.81}$	$72.41\pm_{1.44}$
SelfMix	$80.46\pm_{1.28}$	$72.50\pm_{2.32}$	$80.15\pm_{1.98}$	$72.50\pm_{2.32}$	$78.36\pm_{0.41}$	$74.40\pm_{1.24}$
DyGen†	$83.82\pm_{0.04}$	$79.56\pm_{0.93}$	$83.63\pm_{0.23}$	$81.98\pm_{0.80}$	$84.07\pm_{0.17}$	$81.54\pm_{0.44}$
LAFT†	$82.04\pm_{0.11}$	$76.93\pm_{0.63}$	83.70	81.97	83.61	80.49
ChatGPT (Zero-shot)	69.33					
Supervised GT (0% Noise)	85.02					
Ours	$\mathbf{84.75}\pm_{0.09}$	$\mathbf{83.32}\pm_{0.28}$	$\mathbf{84.65}\pm_{0.33}$	$\mathbf{82.78}\pm_{0.28}$	$\mathbf{84.26}\pm_{0.05}$	$\mathbf{82.90}\pm_{0.11}$

Table 13: The detailed results (accuracy %) on 20ng datasets. DyGen and LAFT also perform experiments on the 20ng dataset, so we directly report the results † of their versions. Since LAFT doesn't public their codes and report accuracy with standard deviation only under 20% Symmetric and 40% Symmetric, we can only report their incomplete results in our paper. **Bold** means the best score.

(COT) (*r.m.* b); (c) removing the Demonstration (*r.m.* c); (d) replacing the feature-aware example sampling with random example sampling (*r.p.* d). The results yield the subsequent observations: Firstly, the performance of our NoiseAL is heavily influenced by demonstrations. This is due to the fact that they possess crucial information that enables ChatGPT to comprehend our tasks. Secondly, the COT is necessary for ChatGPT to activate its capability to adapt to our task, and enhance the accuracy of answer generation. Thirdly, the task prompt is of less importance, indicating that ChatGPT is capable of understanding the task directly from the demonstration. Finally, the feature-aware example sampling strategy is important to the performance of our NoiseAL, especially under a high noise ratio.

Noise	default	*r.m.* a	*r.m.* b	*r.m.* c	*r.p.* d
20%A	**96.80**	96.35	95.32	95.10	96.26
40%A	**95.80**	95.36	94.64	94.52	94.60

Table 14: Prompt contents (Accuracy on Trec under 20% and 40% asymmetric label noise). The default settings include precisely the necessary information for prompting.

I Additional Results of More PLMs

To verify the effectiveness of our proposed NoiseAL on other PLMs, we perform experiments on RoBERTa (refer to Table 23, Table 24, Table 25 and Table 26). All experiment results show the improvement brought by NoiseAL is significant.

J Prompts Details

J.1 Prompts Structure

Our LLMs prompt consists of the following three components:

(1) ***Task description***, which describes the task. For different classification tasks, *e.g.*, question classification, sentiment classification, topic classification, *etc*, the descriptions are different.

`Trec:` You are a text classifier and your task is to classify a given text according to candidate categories. The true category must be one of the candidate categories.

`SST-2:` You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative.

Dataset	**Trec**			
Co-prediction Network(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.3)	95.80	93.20	96.40	93.60
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.5)	95.00	94.00	**97.60**	93.00
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.7)	95.00	93.60	96.40	94.00
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.9)	95.20	94.80	97.20	94.80
BERT (dropout ratio:0.1) + TextCNN	93.00	94.40	96.00	93.00
BERT (dropout ratio:0.1) + BiLSTM (Ours)	**97.16**	**96.40**	96.80	**95.80**

Table 15: The results of different Co-prediction networks on Trec datasets. **Bold** means the best score.

Dataset	**Trec**					
Co-prediction Network(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.3)	89.01	86.04	87.25	87.36	89.18	86.65
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.5)	88.02	84.67	86.48	81.26	91.37	87.31
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.7)	85.82	84.95	85.22	86.26	90.60	87.03
BERT (dropout ratio:0.1) + BERT (dropout ratio:0.9)	90.05	86.98	89.12	82.25	91.04	86.59
BERT (dropout ratio:0.1) + TextCNN	89.89	85.16	88.74	84.78	88.74	86.66
BERT (dropout ratio:0.1) + BiLSTM (Ours)	**90.95**	**89.34**	**91.18**	**89.26**	**91.37**	**90.08**

Table 16: The results of different Co-prediction networks on SST-2 datasets. **Bold** means the best score.

Dataset	**Trec**			
Modules(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**
Ours	**97.16**	**96.40**	**96.80**	**95.80**
w/o co-prediction network	95.20	90.00	84.84	72.36
w/o dynamic-enhanced selection	94.80	87.80	83.96	67.14
w/o hard subsets	95.80	94.60	86.43	82.64
w/o purified subsets	95.20	93.40	88.30	79.89
w/o reliable subsets	95.80	95.40	90.22	88.11

Table 17: Ablation study on Trec datasets. **Bold** means the best score.

IMDB: You are a Sentiment classifier and your task is to classify a given text according to candidate labels. Your answer can be either positive or negative.

Agnews: You are a text classifier and your task is to classify a given text according to candidate topics. Your answer must be exactly one of ['World', 'Sports', 'Business', 'Science/Technology'].

Chemprot: You are a text classifier and your task is to classify a given text according to candidate labels. Your answer must be exactly one of ['Part of','Regulator','Upregulator','Downregulator','Agonist','Antagonist','Modulator', 'Cofactor','Substrate/Product','NOT'].

Semeval: You are a text classifier and your task is to classify a given

Dataset	**SST-2**					
Modules(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
Ours	**90.95**	**89.34**	**91.18**	**89.26**	**91.37**	**90.08**
w/o co-prediction network	81.54	71.98	84.84	72.36	83.90	66.87
w/o dynamic-enhanced selection	85.49	66.32	83.96	67.14	78.57	69.62
w/o hard subsets	89.29	79.40	86.43	82.64	91.15	86.26
w/o purified subsets	87.03	83.35	88.30	79.89	84.12	68.90
w/o reliable subsets	90.27	88.24	90.22	88.11	90.82	85.11

Table 18: Ablation study on SST-2 datasets. **Bold** means the best score.

Dataset	**AGNews**					
Modules(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
Ours	**93.92**	**93.05**	**93.85**	**93.07**	**93.68**	**92.70**
w/o co-prediction network	91.75	86.42	91.04	80.47	91.41	87.21
w/o dynamic-enhanced selection	91.50	87.83	91.25	90.28	91.18	88.74
w/o hard subsets	91.09	90.47	91.46	91.59	91.78	90.87
w/o purified subsets	93.47	92.12	93.25	92.46	93.39	92.18
w/o reliable subsets	93.29	92.83	93.24	92.49	93.32	92.43

Table 19: Ablation study on AGNews datasets. **Bold** means the best score.

Dataset	**20ng**					
Modules(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
Ours	**84.75**	**83.32**	**84.65**	**82.78**	**84.26**	**82.90**
w/o co-prediction network	80.67	75.56	75.09	66.03	81.20	77.56
w/o dynamic-enhanced selection	80.82	76.71	80.46	65.52	81.48	81.64
w/o hard subsets	83.85	80.26	75.17	72.38	81.95	81.93
w/o purified subsets	83.12	82.56	83.34	82.03	81.91	81.96
w/o reliable subsets	78.32	81.76	81.15	79.95	81.36	81.15

Table 20: Ablation study on 20ng datasets. **Bold** means the best score.

`text according to candidate labels. Your answer must be exactly one of ['Cause-Effect','Component-Whole', 'Content-Container','Entity-Destination', 'Entity-Origin','Instrument-Agency','Member-Collection','Message-Topic','Product-Producer'].`

`20ng:` `You are a text classifier and your task is to classify a given news according to candidate categories. The true category must be one of the candidate categories.`

(2) *Demonstration*, which consists of a sequence of annotated samples and is only needed for the few-shot learning setup. Two functions of demon-

Dataset	**IMDB**					
Modules(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
Ours	**92.78**	**91.71**	**92.78**	**91.73**	**92.76**	**90.23**
w/o co-prediction network	86.18	72.10	86.90	72.96	86.66	73.02
w/o dynamic-enhanced selection	87.66	71.20	87.38	71.63	86.56	74.04
w/o hard subsets	91.78	88.24	91.84	86.58	84.84	89.40
w/o purified subsets	91.66	87.36	90.54	85.84	89.14	85.12
w/o reliable subsets	91.88	89.82	90.86	89.90	90.26	88.10

Table 21: Ablation study on IMDB datasets. **Bold** means the best score.

Dataset	**Trec**				**AGNews**						**IMDB**					
Loss function(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
cross-entropy loss	96.00	94.24	95.80	93.60	92.20	91.58	91.91	91.86	91.88	90.89	91.28	90.14	90.62	89.48	90.36	88.46
reversed cross-entropy loss	**97.16**	**96.40**	**96.80**	**95.80**	**93.92**	**93.05**	**93.85**	**93.07**	**93.68**	**92.70**	**92.78**	**91.71**	**92.78**	**91.73**	**92.76**	**90.23**

Table 22: Ablation study for loss functions on purified set $\mathcal{P}$. **Bold** means the best score.

stration are as follows: (a) it gives the LLM evidence to refer to when making decisions, which will greatly improve performance; and (b) it establishes an output format that the LLM's outputs must adhere to, allowing the output—which is natural language—to be readily converted into labels (Sun et al., 2023). For the demonstration sampling, there are some common methods including:

Random Sampling Sampling k examples at random from the training dataset.

kNN Sampling An inherent drawback of random sampling is the absence of assurance that the chosen samples possess semantic relevance to the input sequence. An effective approach is to employ kNN search to select samples that closely resemble the test input. Given a test input $x_i \in \mathcal{P}$, we can obtain its text feature $f(x_i)$ and calculate the cosine similarity of the text feature between x_i and other training text $x_j \in \mathcal{R}$, then sample top-K nearest neighbors to form in-context examples:

$$\mathcal{E} = \underset{j \in \{1,2,\ldots,|\mathcal{R}|\}}{\text{argTopN}} \frac{f(x_i)^T f(x_j)}{\|f(x_i)\| \|f(x_j)\|}, \tag{16}$$

where $\mathcal{E}$ is an index set of the top-K similar samples in clean set $\mathcal{R}$. Note that the text features of datasets can be computed and stored beforehand, allowing efficient sampling.

(3) *Input*, which is the test text to classify.

(4) *Chain-of-Thought (COT)*, our initial investigation indicates that although this prompt design can already yield reasonable results, the prediction is unstable. Specifically, we observe variations in the results when we randomly rearrange the order of the demonstrations. This indicates that the model faces difficulty comprehending the user's viewpoint accurately. To address this instability, we incorporate *Chain-of-Thought (COT)* reasoning into our setting.

J.2 Prompts Cases

Here, we present the prompts we design for the LLMs, including **SST-2** (Table 27), **20ng** (Table 28), **Trec** (Table 29), **AGNews** (Table 30), **IMDB** (Table 31). Besides, we also present the good case (Table 32, 33) and bad case (Table 34) generated by LLMs for some test examples under one-shot settings, respectively. Given the example "*i will be* .", there is a sentence with the sentiment "positive", but this example is classified as "neutral" (as shown in Table 34). In this case, we can observe that LLMs sometimes can not accurately identify the category, which indicates that the label generated by LLMs may be semantically reasonable but without being fully aligned with the annotations in the dataset.

Dataset	**Trec**			
Method(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**
RoBERTa	93.48$\pm_{0.81}$	92.32$\pm_{1.41}$	94.16$\pm_{0.56}$	72.80$\pm_{5.50}$
Co-Teaching	95.36$\pm_{0.51}$	95.16$\pm_{0.70}$	95.08$\pm_{0.16}$	91.56$\pm_{2.28}$
SCE	94.44$\pm_{1.08}$	94.92$\pm_{0.64}$	95.12$\pm_{0.66}$	82.60$\pm_{4.99}$
ELR	95.36$\pm_{0.45}$	94.72$\pm_{0.74}$	95.14$\pm_{0.15}$	91.64$\pm_{2.05}$
CL	95.88$\pm_{0.32}$	95.04$\pm_{0.69}$	95.32$\pm_{0.47}$	88.36$\pm_{2.11}$
SelfMix	94.44$\pm_{1.11}$	93.88$\pm_{1.84}$	94.72$\pm_{1.03}$	92.44$\pm_{1.58}$
DyGen	95.00$\pm_{0.67}$	93.28$\pm_{0.75}$	94.40$\pm_{0.46}$	92.24$\pm_{0.64}$
Ours	**96.36**$\pm_{0.08}$	**95.72**$\pm_{0.10}$	**97.12**$\pm_{0.10}$	**96.80**$\pm_{0.25}$

Table 23: More results (accuracy %) of our methods on Trec datasets. **Bold** means the best score.

Dataset	**AGNews**					
Method(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
RoBERTa	91.51$\pm_{0.47}$	86.74$\pm_{0.95}$	91.63$\pm_{0.32}$	85.90$\pm_{8.52}$	90.47$\pm_{0.42}$	90.10$\pm_{0.38}$
Co-Teaching	93.12$\pm_{0.14}$	91.81$\pm_{0.22}$	92.79$\pm_{0.30}$	91.60$\pm_{0.37}$	93.05$\pm_{0.14}$	91.63$\pm_{0.18}$
SCE	93.10$\pm_{0.20}$	92.44$\pm_{0.28}$	93.11$\pm_{0.13}$	89.98$\pm_{0.96}$	93.02$\pm_{0.22}$	92.16$\pm_{0.13}$
ELR	93.22$\pm_{0.20}$	92.28$\pm_{0.26}$	92.94$\pm_{0.37}$	92.15$\pm_{0.31}$	93.12$\pm_{0.20}$	91.89$\pm_{0.12}$
CL	93.26$\pm_{0.10}$	92.26$\pm_{0.27}$	92.84$\pm_{0.51}$	91.93$\pm_{0.39}$	91.14$\pm_{0.07}$	91.74$\pm_{0.42}$
SelfMix	93.16$\pm_{1.25}$	92.31$\pm_{0.52}$	92.52$\pm_{0.34}$	92.41$\pm_{0.52}$	92.55$\pm_{0.10}$	92.41$\pm_{0.52}$
DyGen	92.27$\pm_{0.04}$	90.85$\pm_{0.22}$	92.21$\pm_{0.14}$	90.59$\pm_{0.31}$	91.34$\pm_{0.14}$	90.65$\pm_{0.32}$
Ours	**93.80**$\pm_{0.08}$	**93.18**$\pm_{0.15}$	**93.40**$\pm_{0.14}$	**93.02**$\pm_{0.43}$	**93.32**$\pm_{0.07}$	**92.84**$\pm_{0.20}$

Table 24: More results (accuracy %) of our methods on Agnews datasets. **Bold** means the best score.

Dataset	**IMDB**					
Method(↓) / **Noise**(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
RoBERTa	91.90$\pm_{0.34}$	82.69$\pm_{1.13}$	91.02$\pm_{0.42}$	80.78$\pm_{2,26}$	86.37$\pm_{0.35}$	73.35$\pm_{0.65}$
Co-Teaching	92.66$\pm_{0.35}$	89.32$\pm_{0.50}$	92.62$\pm_{0.24}$	90.03$\pm_{0.25}$	90.11$\pm_{0.42}$	80.65$\pm_{0.81}$
SCE	91.20$\pm_{0.48}$	87.40$\pm_{2.43}$	86.20$\pm_{1.90}$	88.00$\pm_{1.18}$	86.21$\pm_{2.18}$	76.22$\pm_{2.29}$
ELR	92.86$\pm_{0.35}$	87.85$\pm_{0.76}$	92.95$\pm_{0.86}$	88.83$\pm_{0.59}$	89.29$\pm_{0.80}$	77.37$\pm_{1.17}$
CL	93.15$\pm_{0.27}$	87.97$\pm_{1.64}$	93.49$\pm_{0.27}$	89.24$\pm_{1.51}$	90.28$\pm_{0.41}$	78.84$\pm_{0.79}$
SelfMix	93.78$\pm_{0.25}$	91.94$\pm_{0.44}$	92.08$\pm_{1.93}$	92.34$\pm_{0.23}$	85.61$\pm_{4.57}$	80.94$\pm_{2.33}$
DyGen	90.46$\pm_{0.14}$	84.53$\pm_{0.85}$	90.40$\pm_{0.12}$	84.31$\pm_{0.82}$	89.53$\pm_{0.23}$	80.28$\pm_{1.94}$
Ours	**94.56**$\pm_{0.14}$	**93.72**$\pm_{0.24}$	**94.60**$\pm_{0.26}$	**93.76**$\pm_{0.33}$	**94.28**$\pm_{0.23}$	**90.25**$\pm_{0.32}$

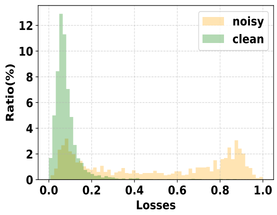
Table 25: More results (accuracy %) of our methods on IMDB datasets. **Bold** means the best score.

Dataset	**SST-2**					
Method(↓) / Noise(→)	20%**S**	40%**S**	20%**A**	40%**A**	20%**I**	40%**I**
Roberta	$87.70 \pm_{0.83}$	$74.96 \pm_{3.58}$	$87.79 \pm_{1.67}$	$68.75 \pm_{4.62}$	$82.41 \pm_{1.62}$	$69.46 \pm_{1.73}$
Co-Teaching	$92.41 \pm_{0.41}$	$53.55 \pm_{7.15}$	$91.57 \pm_{1.01}$	$59.58 \pm_{11.93}$	$82.67 \pm_{1.26}$	$52.26 \pm_{2.64}$
SCE	$89.45 \pm_{1.34}$	$61.72 \pm_{9.78}$	$89.03 \pm_{3.26}$	$55.95 \pm_{10.14}$	$82.28 \pm_{1.01}$	$66.55 \pm_{1.97}$
ELR	$90.98 \pm_{0.60}$	$73.06 \pm_{12.43}$	$91.75 \pm_{0.33}$	$73.45 \pm_{11.84}$	$82.16 \pm_{1.06}$	$68.42 \pm_{0.94}$
CL	$91.75 \pm_{1.40}$	$86.69 \pm_{2.31}$	$93.45 \pm_{0.40}$	$85.37 \pm_{3.43}$	$83.43 \pm_{1.25}$	$69.25 \pm_{1.47}$
SelfMix	$91.02 \pm_{2.06}$	$88.12 \pm_{1.58}$	$90.89 \pm_{0.89}$	$86.23 \pm_{1.66}$	$85.00 \pm_{3.03}$	$69.89 \pm_{0.66}$
DyGen	$89.87 \pm_{0.92}$	$77.73 \pm_{2.30}$	$90.25 \pm_{1.03}$	$71.10 \pm_{3.79}$	$89.82 \pm_{0.59}$	$78.74 \pm_{2.65}$
Ours	$\mathbf{94.21} \pm_{0.11}$	$\mathbf{92.83} \pm_{0.17}$	$\mathbf{94.32} \pm_{0.15}$	$\mathbf{92.71} \pm_{0.46}$	$\mathbf{94.45} \pm_{0.19}$	$\mathbf{91.78} \pm_{0.44}$

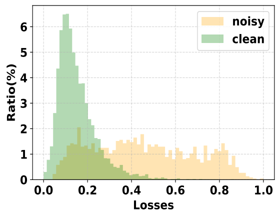
Table 26: More results (accuracy %) of our methods on SST-2 datasets. **Bold** means the best score.

SST-2
Task description: You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative.
Demonstration: Text: wonder of wonders – a teen movie with a humanistic message . Candidate sentiments: positive, negative The sentiment is: positive
Inputs: Text: jonathan parker 's bartleby should have been the be-all-end-all of the modern-office anomie films . Candidate sentiments: positive, negative Let's think step-by-step.

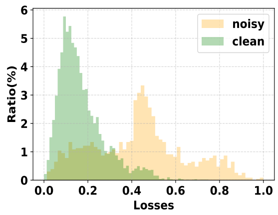
Table 27: The prompt instruction for ChatGPT on **SST-2**.



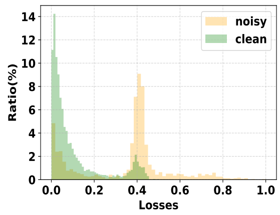
(a) Epoch1 (87.86%)



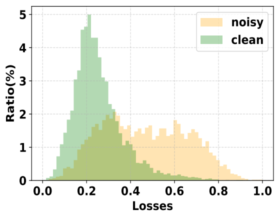
(b) Epoch2 (86.92%)



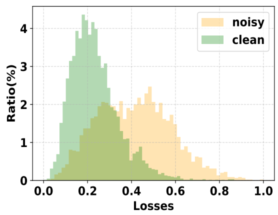
(c) Epoch3 (87.69%)



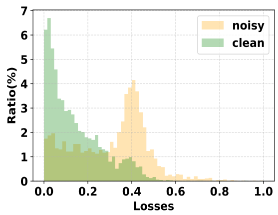
(d) Epoch4 (83.96%)



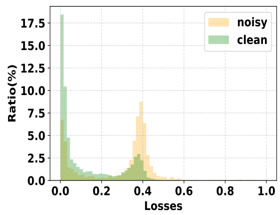
(e) Epoch1 (79.73%)



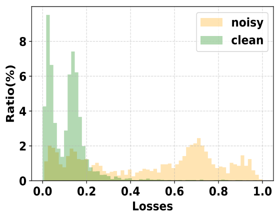
(f) Epoch2 (78.08%)



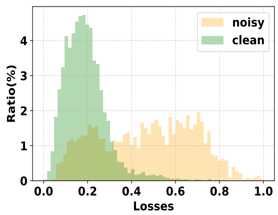
(g) Epoch3 (76.48%)



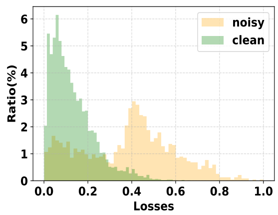
(h) Epoch4 (76.15%)



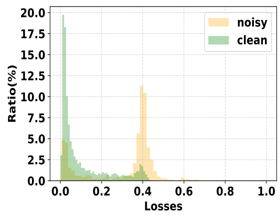
(i) Epoch1 (90.38%)



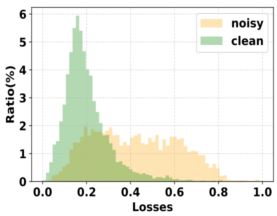
(j) Epoch2 (89.01%)



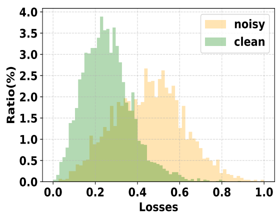
(k) Epoch3 (88.19%)



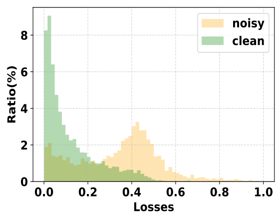
(l) Epoch4 (84.95%)



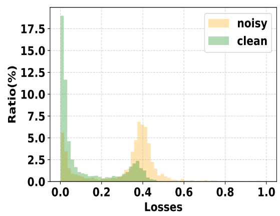
(m) Epoch1 (82.36%)



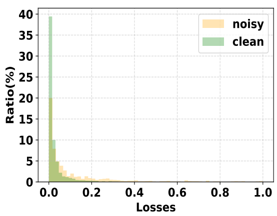
(n) Epoch2 (81.48%)



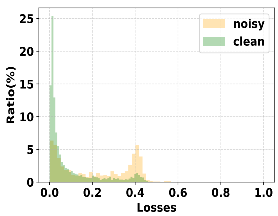
(o) Epoch3 (81.32%)



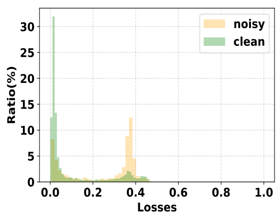
(p) Epoch4 (79.73%)



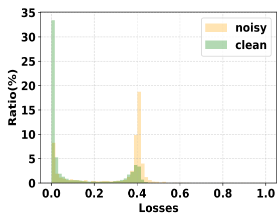
(q) Epoch1 (81.04%)



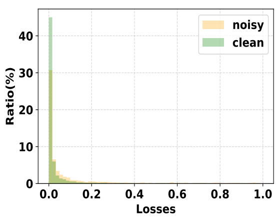
(r) Epoch2 (78.57%)



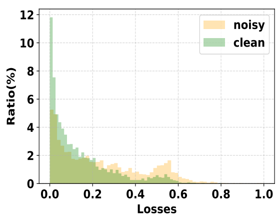
(s) Epoch3 (85.11%)



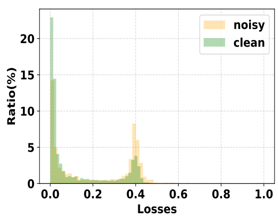
(t) Epoch4 (82.09%)



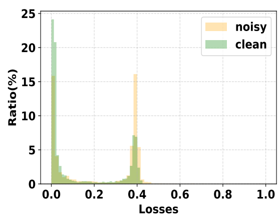
(u) Epoch1 (70.11%)



(v) Epoch2 (70.00%)

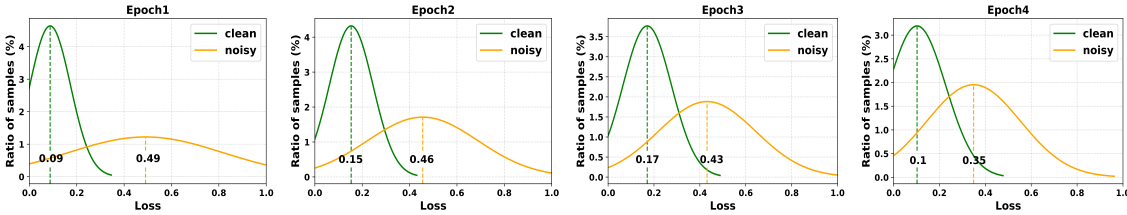


(w) Epoch3 (69.07%)

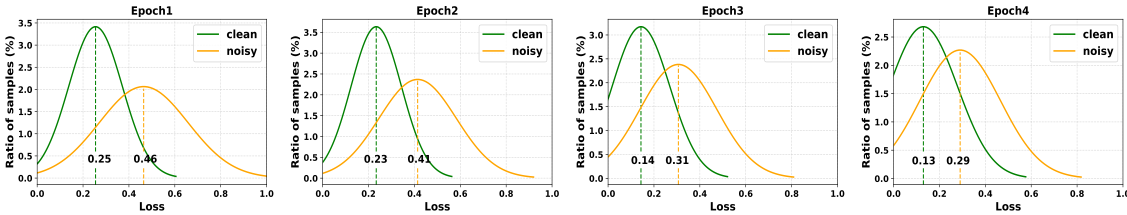


(x) Epoch4 (68.02%)

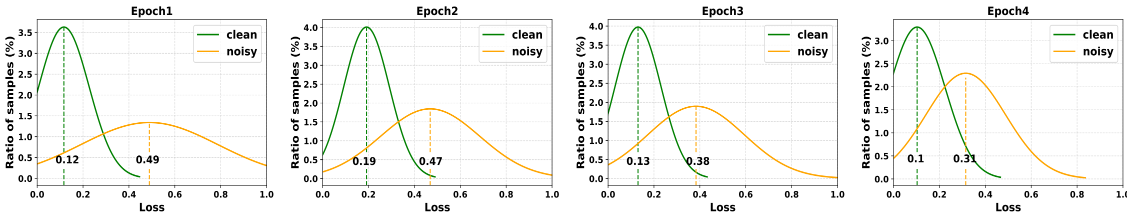
Figure 8: Loss histogram of SST-2 with 20% symmetric label noise (a-d), 40% symmetric label noise (e-h), 20% asymmetric label noise (i-l), 40% asymmetric label noise (m-p), 20% idn label noise (q-t), 40% idn label noise (u-x). Clean data and noisy data are marked by green and yellow respectively. The accuracy is listed in the parentheses. We observe that the loss value can not separate the clean data from the noisy ones. Hence, the methods that fixed-loss-value-based (such as SelfMix) is not applicable in our scenario.



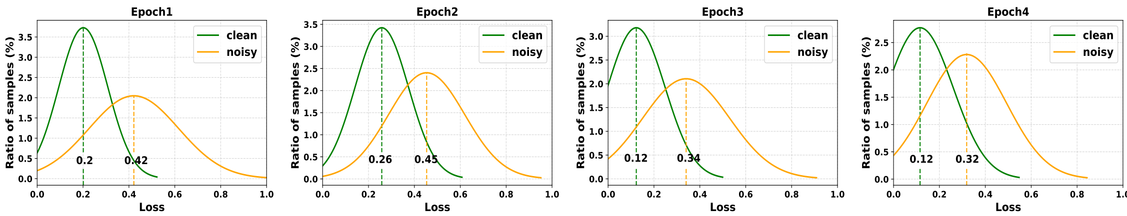
(a) Epoch1 (87.86%) (b) Epoch2 (86.92%) (c) Epoch3 (87.69%) (d) Epoch4 (83.96%)



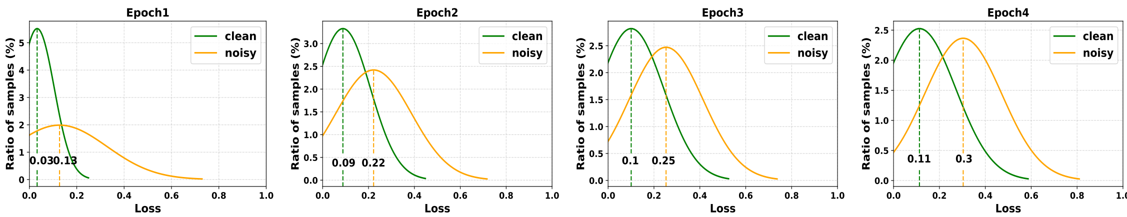
(e) Epoch1 (79.73%) (f) Epoch2 (78.08%) (g) Epoch3 (76.48%) (h) Epoch4 (76.15%)



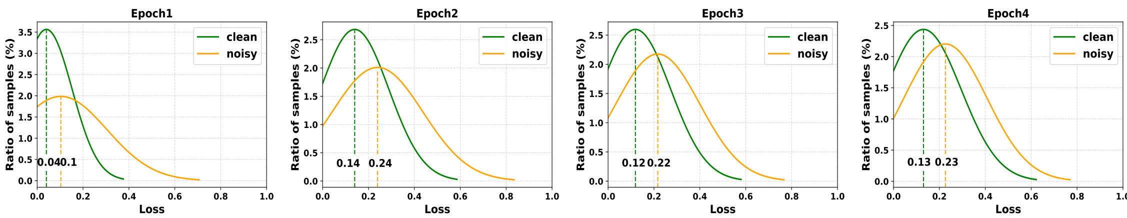
(i) Epoch1 (90.38%) (j) Epoch2 (89.01%) (k) Epoch3 (88.19%) (l) Epoch4 (84.95%)



(m) Epoch1 (82.36%) (n) Epoch2 (81.48%) (o) Epoch3 (81.32%) (p) Epoch4 (79.73%)



(q) Epoch1 (81.04%) (r) Epoch2 (78.57%) (s) Epoch3 (85.11%) (t) Epoch4 (82.09%)



(u) Epoch1 (70.11%) (v) Epoch2 (70.00%) (w) Epoch3 (69.07%) (x) Epoch4 (68.02%)

Figure 9: The loss distributions of SST-2 with 20% symmetric label noise (a-d), 40% symmetric label noise (e-h), 20% asymmetric label noise (i-l), 40% asymmetric label noise (m-p), 20% idn label noise (q-t), 40% idn label noise (u-x). The solid line represents the loss distributions, and the dashed line points out the mean value of loss distributions. The mean value of samples with clean labels (green) and noisy labels (orange) gradually decreases during the training process.

20ng
Task description:
You are a text classifier and your task is to classify a given news according to candidate categories. The true category must be one of the candidate categories.
Demonstration:
Text: re you will all go to hell in article tan psuvm psu edu andrew newell tan psuvm psu edu writes in article jsn psuvm psu edu jsn psuvm psu edu writes you blashephemers you will all go to hell for not believing in god be prepared for your eternal damnation readers of the group how convenient that he doesn t have a real name let s start up the letters to the sysadmin shall we his real name is jeremy scott noonan vmoper psuvm psu edu should have at least some authority or at least know who to email postmast psuvm bitnet respectively p rfowles or p wverity the sys admins at the same node are probably a better idea than the operator benedikt ?
Candidate categories: alt.atheism, comp.graphics, talk.religion.misc, soc.religion.christian, talk.politics.guns, sci.space, comp.os.ms-windows.misc, talk.politics.misc, comp.windows.x, sci.crypt, rec.autos, sci.electronics, comp.sys.mac.hardware, rec.motorcycles, talk.politics.mideast, rec.sport.hockey, misc.forsale, rec.sport.baseball, sci.med, comp.sys.ibm.pc.hardware
The category is: alt.atheism
Inputs:
Text: re wholly babble was re free moral agency in article p psilink com robert knowles p psilink com writes deletion of course there is also the book of the subgenius and that whole collection of writings as well does someone know a ftp site with it benedikt ?
Candidate categories: alt.atheism, comp.graphics, talk.religion.misc, comp.os.ms-windows.misc, soc.religion.christian, sci.crypt, talk.politics.guns, sci.space, sci.electronics, rec.motorcycles, talk.politics.misc, rec.sport.hockey, comp.windows.x, comp.sys.mac.hardware, rec.autos, comp.sys.ibm.pc.hardware, sci.med, talk.politics.mideast, rec.sport.baseball, misc.forsale
Let's think step-by-step. The category is:

Table 28: The prompt instruction for ChatGPT on **20ng**.

Trec
Task description: You are a text classifier and your task is to classify a given text according to candidate categories. The true category must be one of the candidate categories. **Demonstration:** Text: what is gymnophobia? Candidate categories: description, entity, human, abbreviation, location, numeric The category is: description **Inputs:** Text: what is the name of the art of growing miniature trees? Candidate categories: entity, description, human, abbreviation, numeric, location Let's think step-by-step. The category is :

Table 29: The prompt instruction for ChatGPT on **Trec**.

AGNews
Task description: You are a text classifier and your task is to classify a given text according to candidate topics. Your answer must be exactly one of ['World', 'Sports', 'Business', 'Science/Technology']. **Demonstration:** Text: AP - As of Wednesday, Nov. 17, 2004, at least 1,214 members of the U.S. military have died since the beginning of the Iraq war in March 2003, according to an Associated Press count. At least 944 died as a result of hostile action, the Defense Department said as of Wednesday. The figures include three military civilians. Candidate topics: World, Sports, Science/Technology, Business Topic: World **Inputs:** Text: AP - A U.S. Army tank company commander accused of murdering a man in Iraq went before a military court Wednesday for a fresh round of hearings to determine whether he should be court-martialed. Candidate topics: World, Sports, Science/Technology, Business Let's think step-by-step. The topic is :

Table 30: The prompt instruction for ChatGPT on **AGNews**.

IMDB

Task description:

You are a Sentiment classifier and your task is to classify a given text according to candidate labels. Your answer can be either positive or negative.

Demonstration:

Text: I read some previous comments stating that this movie loses steam towards the end of the movie and also that it has a similar ending to Cape Fear. I completely disagree. I'm going to give a simple review for the normal moviegoers out there. I thought the casting was perfect. I thought this was one of Ed Harris's best performances. What an evil psychopath!!! I have a lot of respect for his acting after viewing his performance i?

Candidate label: positive, negative

Label: Positive

Inputs:

Text: Woah! Is one thing I can say about this movie. Personally I'm one of those people who loves cats so that would have been a big down side to the movie, but I loved it how cats from every were got their revenge at the end.I liked the movie, but I have to admit it was because I found Brian Krause who played Charles Brady very very irresistible, I guess lust got the better of me while watching this movie.There was one thing that I?

Candidate label: positive, negative

Let's think step-by-step. The label is :

Table 31: The prompt instruction for ChatGPT on **IMDB**.

SST-2

Good Case:

You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative.

Text: return to never land is clean , standard disney animated fare , with enough creative energy and wit to entertain all ages .

Candidate sentiments: positive, negative

The sentiment is: positive

Text: will undoubtedly play well in european markets , where mr. besson is a brand name , and in asia , where ms. shu is an institution , but american audiences will probably find it familiar and insufficiently cathartic .

Candidate sentiments: positive, negative

Let's think step-by-step.

Output:

The sentiment is: negative
The sentiment is: negative
The sentiment is: negative
The sentiment is: negative
The sentiment is: negative

You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative.

Text: the rare imax movie that you 'll wish was longer than an hour .

Candidate sentiments:positive, negative

The sentiment is: positive

Text: an edgy thriller that delivers a surprising punch .

Candidate sentiments: positive, negative

Let's think step-by-step.

Output:

The sentiment is: positive
The sentiment is: positive
The sentiment is: positive
The sentiment is: positive
The sentiment is: positive

Table 32: The good case of SST-2. LLMs (chatGPT) run the generation 5 times with a temperature of 0.5 to produce different reasoning paths and predictions. Then we use majority voting to get the right prediction results.

SST-2
Good Case: You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative. Text: it 's hard to quibble with a flick boasting this many genuine cackles , but notorious c.h.o. still feels like a promising work-in-progress . Candidate sentiments: positive, negative The sentiment is: positive Text: overall , cletis tout is a winning comedy that excites the imagination and tickles the funny bone . Candidate sentiments: positive, negative Let's think step-by-step. Output: I'm sorry, but as an AI developed by OpenAI, I don't have the capability to accept tips or process transactions. However, I'm here to help you with any questions or tasks within my capabilities. If you need assistance with sentiment classification or any other topic, feel free to ask! It appears there might be a misunderstanding. As an AI developed by OpenAI, I don't have the functionality to accept tips or payments. My purpose is to provide information and answer questions to the best of my ability based on the data I've been trained on. Regarding the sentiment classification task you've provided, the sentiment for the second text is: positive The sentiment is: positive The sentiment is: positive I'm sorry for any confusion, but as an AI developed by OpenAI, I don't have the capability to accept tips or process transactions. My purpose is to provide information and answer questions to the best of my ability based on the data I was trained on. If you have any questions or need assistance with a topic, feel free to ask! Regarding the sentiment analysis task you've provided: Text: "overall, cletis tout is a winning comedy that excites the imagination and tickles the funny bone." The sentiment is: positive

Table 33: The good case of SST-2. LLMs (chatGPT) run the generation 5 times with a temperature of 0.5 to produce different reasoning paths and predictions. Then we use majority voting to get the right prediction results.

SST-2

Bad Case:

You are a sentiment classifier and your task is to classify a given text according to candidate sentiment. Your answer can be either positive or negative.

Text: Text:a well-executed spy-thriller .

Candidate sentiments: positive, negative

The sentiment is: positive

Text: i will be .

Candidate sentiments: positive, negative

Let's think step-by-step.

Output:

The sentiment is: neutral. The text "i will be" is incomplete and does not provide enough context or information to determine a positive or negative sentiment. It's a neutral statement without additional context.
The sentiment is: neutral. The text "i will be" is not sufficient to determine a clear positive or negative sentiment. It is a neutral statement without additional context or emotional language.
The sentiment is: neutral. The text "i will be" does not contain enough information to determine a positive or negative sentiment. It is a neutral statement without any context or emotional content.
The sentiment is: neutral. The text "i will be" does not contain enough information to determine a positive or negative sentiment. It is a neutral statement without any clear emotional content.
The sentiment is: neutral. The text "i will be" is neutral as it does not express a clear positive or negative sentiment. It is a fragment of a sentence and lacks context or emotion that would allow for a sentiment classification.

Table 34: The bad case of SST-2. LLMs (chatGPT) run the generation 5 times with a temperature of 0.5 to produce different reasoning paths and predictions. Then we use majority voting to get the wrong results.