Adaptive Ranking-based Sample Selection
for Weakly Supervised Class-imbalanced Text Classification

Linxin Song¹, Jieyu Zhang², Tianxiang Yang¹ and Masayuki Goto¹
¹ Waseda University    ² University of Washington
songlx.imse.gt@ruri.waseda.jp, jieyuz2@cs.washington.edu, you_tensyou@akane.waseda.jp, masagoto@waseda.jp

Abstract
To obtain a large amount of training labels inexpensively, researchers have recently adopted the weak supervision (WS) paradigm, which leverages labeling rules to synthesize training labels rather than using individual annotations to achieve competitive results for natural language processing (NLP) tasks. However, data imbalance is often overlooked in applying the WS paradigm, despite being a common issue in a variety of NLP tasks. To address this challenge, we propose Adaptive Ranking-based Sample Selection (ARS2), a model-agnostic framework to alleviate the data imbalance issue in the WS paradigm. Specifically, it calculates a probabilistic margin score based on the output of the current model to measure and rank the cleanliness of each data point. Then, the ranked data are sampled based on both class-wise and rule-aware ranking. In particular, the two sample strategies correspond to our motivations: (1) to train the model with balanced data batches to reduce the data imbalance issue and (2) to exploit the expertise of each labeling rule for collecting clean samples. Experiments on four text classification datasets with four different imbalance ratios show that ARS2 outperformed the state-of-the-art imbalanced learning and WS methods, leading to a 2%-57.8% improvement on their F1-score. Our implementation can be found in https://github.com/JieyuZ2/wrench/blob/main/wrench/endmodel/ars2.py.

1 Introduction
Deep learning models rely heavily on high-quality yet expensive, labeled data. Owing to this considerable cost, the weak supervision (WS) paradigm has increasingly been used to reduce human efforts (Ratner et al., 2016a; Zhang et al., 2021). This approach synthesizes training labels with labeling rules to significantly improve the efficiency of creating training sets and have achieved competitive results in natural language processing (NLP) (Yu et al., 2020; Ren et al., 2020; Rühling Cachay et al., 2021). However, existing methods leveraging the WS paradigm to perform NLP tasks mostly focus on reducing the noise in training labels brought by labeling rules, while ignoring the common and critical problem of data imbalance. In fact, in a preliminary experiment performed as part of the present work (Fig. 1), we found that the WS paradigm may amplify the imbalance ratio of the dataset because the synthesized training labels tend to have more imbalanced distribution.

To address this issue, we propose ARS2 as a general model-agnostic framework based on the WS paradigm. ARS2 is mainly divided in two steps, including (1) warm-up, in which stage noisy data is used to train the model and obtain a noise detector; (2) continual training with adaptive ranking-based sample selection. In this stage, we use the noise detector trained in the warm-up stage to evaluate the cleanliness of the data, and use the ranking obtained based on this evaluation to sample the data. We followed previous works Ratner et al. (2016a); Ren et al. (2020); Zhang et al. (2022b) in using heuristic programmatic rules to annotate the data. In weak supervised learning, researchers use a label model to aggregate weak labels annotated by

Figure 1: Comparison of class distribution between the ground truth labels and labels produced by weak supervision (WS) on TREC dataset. The uncovered piece represents the data not covered by any labeling rule in WS. It may be observed that WS amplified the class imbalance.
rules to estimate the probabilistic class distribution of each data point. In this work, we use a label model to integrate the weak labels given by the rules as pseudo-labels during the training process to obviate the need for manual labeling.

To select the samples most likely to be clean, we adopt a selection strategy based on small-loss, which is a very common method that has been verified to be effective in many situations (Jiang et al., 2018; Yu et al., 2019; Yao et al., 2020). Specifically, deep neural networks, have strong ability of memorization (Wu et al., 2018; Wei et al., 2021), will first memorize labels of clean data and then those of noisy data with the assumption that the clean data are of the majority in a noisy dataset. Data with small loss can thus be regarded as clean examples with high probability. Inspired by this approach, we propose probabilistic margin score (PMS) as a criterion to judge whether data are clean. Instead of using the confidence given by a model directly, a confidence margin is used for better performance (Ye et al., 2020). We also performed a comparative experiment on the use of margin versus the direct use of confidence, as described in Sec. 3.3.

Sample selection based on weak labels can lead to severe class imbalance. Consequently, models trained using these imbalanced subsets can exhibit both superior performance on majority classes and inferior performance on minority classes (Cui et al., 2019). A reweighted loss function can partially address this problem. However, performance remains nonetheless limited by noisy labels, that is, data with majority-class features may be annotated as minority-class data incorrectly, which misleads the training process. Therefore, we propose a sample selection strategy based on class-wise ranking (CR) to address imbalanced data. Using this strategy, we can select relatively balanced sample batches for training and avoid the strong influence of the majority class.

To further exploit the expertise of labeling rules, we also propose another sample selection strategy called rule-aware ranking (RR). We use aggregated labels as pseudo-labels in the WS paradigm and discards weak labels. However, the annotations generated by rules are likely to contain a considerable amount of valid information. For example, some rules yield a high proportion of correct results. The higher the PMS, the more likely the labeling result of the rules is to be close to the ground truth. Using this strategy, we can select batches with clean data for training and avoid the influence of noise.

The primary contributions of this work are summarized as follows. (1) We propose a general, model-agnostic weakly supervised leading framework called ARS2 for imbalanced datasets; (2) we also propose two reliable adaptive sampling strategies to address data imbalance issues. (3) The results of experiments on four benchmark datasets are presented to demonstrate that the ARS2 improved on the performance of existing imbalanced learning and weakly supervised learning methods, by 2%-57.8% in terms of F1-score.

2 Weakly Supervised Class-imbalanced Text Classification

2.1 Problem Formulation
In this work, we study class-imbalanced text classification in a setting with weak supervision. Specifically, we consider an unlabeled dataset \( D \) consisting of \( N \) documents, each of which is denoted by \( x_i \in \mathcal{X} \). For each document \( x_i \), the corresponding label \( y_i \in \mathcal{Y} = \{1, 2, ..., C\} \) is unknown to us, whereas the class prior \( p(y) \) is given and highly imbalanced. Our goal is to learn a parameterized function \( f(\cdot; \theta) : \mathcal{X} \to \Delta^C \) which outputs the class probability \( p(y \mid x) \) and can be used to classify documents during inference.

To address the lack of ground truth training labels, we adopt the two-stage weak supervision paradigm (Ratner et al., 2016b; Zhang et al., 2021). In particular, we rely on \( k \) user-provided heuristic rules \( \{r_i\}_{i \in \{1, ..., k\}} \) to provide weak labels. Each rule \( r_i \) is associated with a particular label \( y_{r_i} \in \mathcal{Y} \), and we denote by \( l_i \) the output of the rule \( r_i \). It either assigns the associated label \( l_i = y_{r_i} \) to a given document or abstains \( l_i = -1 \) on this example. Note that the user-provided rules could be noisy and conflict with one another. For the document \( x \), we concatenate the output weak labels of \( k \) rules \( l_1, ..., l_k \) as \( l_x \). Throughout this work, we apply the weak labels output by heuristic rules to train a text classifier.

2.2 Aggregation of Weak Labels
Label models are used to aggregate weak labels under the weak supervision paradigm, which are in turn used to train the desired end model in the next stage. Existing label models include Majority Voting (MV), Probabilistic Graphical Models

\[ \Delta^C \] is a \( C \)-dimension simplex.
Figure 2: Overview of ARS2. Our framework has two stages: (1) warm-up, which is used to let the model learn how to distinguish noisy data; (2) continual training with adaptive sampling, which is used to sample clean data. We adopt two different adaptive sampling strategies, including class-wise ranking sampling and rule-aware ranking sampling.

(PGM) (Dawid and Skene, 1979; Ratner et al., 2019b; Fu et al., 2020), etc. In this research, we use PGM implement by Ratner et al. (2019b) as our label model $g(\cdot)$, which can be described as

$$g(l_x) = P(y \mid l_x).$$

This assumes that $l_x$ as a random variable for label model. After modeling the relationship between the observed variable $l_x$ and unobserved variable $y$ by Bayes methods, the label model obtains the posterior distribution of $y$ given $l_x$ by inference process like expectation maximization or variational inference. Finally, we set the maximum value of $P(y \mid l_x)$ as the hard pseudo-label $\tilde{y}_x$ of $x$ for later end model training.

2.3 Adaptive Ranking-based Sample Selection

We propose an adaptive ranking sample selection approach to simultaneously solve the problems caused by data imbalance and those resulting from the noise generated by the application of procedural rules. First, the classification model $f(\cdot; \theta)$ is warmed up with pseudo-labels $\tilde{y}_x$, which are used to train the model as a noise indicator that can discriminate noise in the next stage. Then, we continue training the warmed-up model by using the data sampled by adaptive data sampling strategies, including class-wise ranking (CR) and rule-aware ranking (RR) supported by probabilistic margin score (PMS). The training procedures are summarized in Algorithm 1 and the proposed framework is illustrated in Figure 2.

**Warm-up.** Inspired by Zheng et al. (2020), the prediction of a noisy classification model can be a good indicator of whether the label of a training data point is clean. Our method relies on a noise indicator trained at warm-up to determine whether each training data is clean. However, a model with sufficient capacity (e.g., more parameters than training examples) can “memorize” each example, overfitting the training set and yielding poor generalization to validation and test sets (Goodfellow et al., 2016). To prevent a model from overfitting noisy data, we warm-up the model $f(\cdot; \theta)$ with early stopping (Dodge et al., 2020), and solve the optimization problem by

$$\min_{\theta} \frac{1}{N} \sum_{x \in X} L(f(x; \theta), \tilde{y}_x),$$

where $L$ denotes a loss function and $\tilde{y}_x$ is pseudo-label aggregated by a label model. In this research, we do not limit the definition of the loss function; that is, any loss function suitable for a multi-classification task can be used.

**Continual training with sample selection.** Noisy and imbalanced labels impair the predictive

---

**Algorithm 1: ARS2**

**Input:** Weak labeled training data $X$; Pseudo label $\tilde{y}$; Classification model $f(\cdot; \theta)$.

**// Warm-up $f(\cdot; \theta)$ with weak labeled data.**

for $t = 1, 2, \ldots, T$ do

1. Sample a minibatch $B$ from $X$.
2. Update $\theta$ by Eq. (2) before early stop.

**// Continue training with sample selection.**

for $t = 1, 2, \ldots, T$, do

1. Calculate score for all $x \in X$ (Sec. 2.4).
2. Sample $Q(t)$ from $X$ (Sec. 2.5).
3. Sample $S(t)$ from $X$ (Sec. 2.6).
4. Update $\theta$ using $U(t) = Q(t) \cup S(t)$ by Eq. (3).

**Output:** Output final model $f(\cdot; \theta)$. 

---

*PGM* (Dawid and Skene, 1979; Ratner et al., 2019b; Fu et al., 2020), etc. In this research, we use PGM implement by Ratner et al. (2019b) as our label model $g(\cdot)$, which can be described as

$$g(l_x) = P(y \mid l_x).$$

This assumes that $l_x$ as a random variable for label model. After modeling the relationship between the observed variable $l_x$ and unobserved variable $y$ by Bayes methods, the label model obtains the posterior distribution of $y$ given $l_x$ by inference process like expectation maximization or variational inference. Finally, we set the maximum value of $P(y \mid l_x)$ as the hard pseudo-label $\tilde{y}_x$ of $x$ for later end model training.
performance of learning models, especially deep neural networks, which tend to exhibit a strong memorization capability (Wu et al., 2018; Wei et al., 2021). To reduce the influence of imbalance and noise in the classification model, we adopt the intuitive approach of simply continuing the training process with clean and balanced data. In the continual training phase, we filter for clean data using a warmed-up model and sample data in a balanced way to calibrate the model. This procedure provides a more robust performance on a noisy, imbalanced dataset. To achieve this, we propose a measurement of label quality called probabilistic margin score (PMS) (Sec. 2.4), using two sample selection strategies, including class-wise ranking (CR) (Sec. 2.5) and rule-aware ranking (RR) (Sec. 2.6). We adopt batch sampling by using CR as $S$ and RR as $Q$ after several steps, using batch $U = Q \cup S$ to continue training the model by solving the following optimization problem.

$$
\min_{\theta} \frac{1}{|U|} \sum_{x \in U} \mathcal{L}(f(x; \theta), \tilde{y}_x).
$$

(3)

**Dynamic-Rate Sampling** During the training procedure, the model learns from data with easy/frequent patterns and those with harder/irregular patterns in separate training stages (Arpit et al., 2017). According to Chen et al. (2020), a small amount of clean data should be selected first, and then a larger amount of clean data can be selected after the model has been trained well. Therefore, instead of fixing the number of selected data, as the continual training stage proceeds, we linearly increase the size of the training batch $U$ by

$$
k = \frac{B}{C} \times (1 + \frac{t(\gamma - 1)}{T_s}),
$$

(4)

where $B$ indicates batch size. The sampling ratio $\gamma$ ranged from 1 to 10 in our experiments.

### 2.4 Probabilistic Margin Score

To measure the impact of the training label on label quality, inspired by Pleiss et al. (2020), we use the margin of output predictions to identify data with low label quality for later adaptive ranking-based sample selection. The predictions of the network can be regarded as a measurement of the quality of labels, which means that a correct prediction indicates high label quality, and vice versa (Jiang et al., 2018; Yu et al., 2019; Yao et al., 2020). Moreover, margins are advantageous because they are simple and efficient to compute during training, and they naturally factorize across samples, which makes it possible to estimate the contribution of each data point to label quality.

Base on this idea, we propose the probabilistic margin score (PMS) to reflect the quality of a given label. PMS is formulated as

$$
s(x) = f_y(x; \theta) - \max_{y \in Y \setminus \{y\}} f_y(x; \theta),
$$

(5)

where $f_y(\cdot; \theta)$ denotes the prediction of $y$ by classification model $f(\cdot; \theta)$ and $s(x) \in [-1, 1]$. A negative margin corresponds to an incorrect prediction, and vice versa. The margin captures how much lager the (potentially incorrect) assigned confidence is than all other confidences, which indicates the label quality of $x$.

### 2.5 Class-wise Ranking

If the top-ranked data are directly selected based on PMS to form the training batch, the majority-class data are highly like to dominate the selected batch. This class imbalance in the data results in suboptimal performance. To overcome this problem, we propose class-wise ranking (CR). Specifically, instead of naively selecting the top-k data ranked by PMS from the entire training set, we select top-ranked data from each class to maintain the class balance of the resultant batch. However, CR might introduce more noise if we attempted to construct a perfectly class-balanced data batch. Thus, to reduce the noise while ensuring class balance as much as possible, we set a threshold on PMS to filter the noisy data.

We assume that the set of data belonging to class $y_i$ is $X_{y_i}$, and extract the top-k ranked data from each $X_{y_i}$ as $S_{y_i}^{(t)}$ at step $t$ according to the ranking result from $s(X)$. We also set a threshold $\xi$ for $s(X)$ to filter noisy data.

$$
S_{y_i}^{(t)} = \text{top-k}_{x \in X_{y_i}, s(x) > \xi} s(x).
$$

(6)

Then, we concatenate $S_{y_i}^{(t)}$ drawn from each class as a new batch $S^{(t)}$ as given below.

$$
S^{(t)} = \bigcup_{y_i \in Y} S_{y_i}^{(t)}.
$$

(7)

### 2.6 Rule-aware Ranking

As discussed above, the training labels are synthesized from multiple labeling rules. In the proposed
framework, these labeling rules are used not only to produce training labels as in a typical WS pipeline, but also to guide the sample selection. Specifically, each labeling rule typically assigns a certain label, e.g., sports, to only a part of the training set based on its expertise. Within this part of the dataset, some of the data must actually belong to the class sports; otherwise, the relevant labeling rule would be useless. Thus, we propose rule-aware ranking (RR) to separately rank the data covered by each labeling rule and then extract the top-ranked data from each individual ranking. We plan to elaborate on this selection process in a sequel to the present work.

We denote the set of data covered by each labeling rule \( r_i \) as \( \mathcal{X}_{r_i} \). We then rank \( \mathcal{X}_{r_i} \) based on PMS \( s(x) \) and extract the top-k data from each \( \mathcal{X}_{r_i} \) as \( Q^{(t)}_{r_i} \) at step t.

\[
Q^{(t)}_{r_i} = \text{top-k} \ s(x). \quad (8)
\]

Specifically, to avoid the error-propagation, we use the weak label \( l_x \) instead of \( \tilde{y}_x \) as the new label of \( x \in Q^{(t)}_{r_i} \) when \( l_x \) is unipolar. We claim that data with high confidence will also have a high probability of weak labels being equal to ground truth labels, especially when multiple rules assign the same weak label to the data. The main reason is that those weak labels are not influenced by the training process and are weakly supervised by human level. Then, we take the union of \( Q^{(t)}_{r_i} \) drawn from each subset as a new batch \( Q^{(t)} \).

\[
Q^{(t)} = \bigcup_{r_i \in r} Q^{(t)}_{r_i}. \quad (9)
\]

3 Experiment

3.1 Experiment Setup

Tasks and Datasets. To evaluate the proposed methods, we used four open benchmark datasets, including AGNews (news topic classification), Yelp (sentiment classification) (Zhang et al., 2015), TREC (question classification Li and Roth (2002)) and ChemProt (relation classification Krallinger et al. (2017)). Specifically, each dataset was weakly annotated by several rules provided by Ren et al. (2020); Awasthi et al. (2020); Yu et al. (2020). The relevant statistic for each dataset are shown in Table 1.

Imbalance Learning Setups. Following Cui et al. (2019), we created an imbalanced version of AGNews and Yelp by reducing the training and validation examples for each class with an exponential function according to the ground truth data labels. We set four different imbalance ratios \( \rho = \max_y \ P(x) / \min_y \ P(x) \) for AGNews and Yelp in 1, 10, 20 and 50, respectively. We used the original versions of TREC and Chemprot, as they are known to be imbalanced. Relevant statistics on the imbalanced datasets are shown in A.1.

Weak Supervision Setups. In weak supervision, assume that the data is not artificially annotated, the labels are annotated by a label model instead of using the ground truth labels. The label model can analyze the results of rule-based annotation and output the most likely label for a given data sample. Throughout the experiments, we used Snorkel (Ratner et al., 2019b) as the label model to aggregate the outputs of labeling rules.

Implementation Details. We choose the Multi-Layer Perceptron (MLP) with 2 hidden layers and RoBERTa (Liu et al., 2019) as the backbone language model for our method in all baselines. In the case of using RoBERTa as a backbone model, rather than training the RoBERTa as a noise indicator in warm-up stage, we used a warmed-up MLP as a noisy indicator to sample training batches for RoBERTa because a large model may easily overfit noisy data in a few steps. We used the classification macro-average F1 score on the test set as the evaluation metric for all datasets. We implemented our method using PyTorch (Paszke et al., 2019) with the WRENCH code-base (Zhang et al., 2021)\(^2\).

3.2 Baselines

Imbalance Learning Methods: (1) Logit Adjustment (LA) (Menon et al., 2020): This method combines post-hoc weight normalization and loss modification to balance head and tail classes by adding a class-wise offset to the loss. (2) Effective Number (EN) (Cui et al., 2019): This method uses the proportion of sampled data as the class-wise weight of loss function, which helps the model to learn a useful decision boundary. (3) Dice Loss

---

\(^2\)Our implementations will be released upon the acceptance of this work.

Table 1: Dataset Statistics.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Task</th>
<th># Class</th>
<th># Rule</th>
<th># Train</th>
<th># Valid</th>
<th># Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>AG News</td>
<td>News Topic Class.</td>
<td>4</td>
<td>9</td>
<td>96k</td>
<td>12k</td>
<td>12k</td>
</tr>
<tr>
<td>Yelp</td>
<td>Sentiment Class.</td>
<td>2</td>
<td>8</td>
<td>30.4k</td>
<td>3.8k</td>
<td>3.8k</td>
</tr>
<tr>
<td>TREC</td>
<td>Question Class.</td>
<td>6</td>
<td>68</td>
<td>4.9k</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>ChemProt</td>
<td>Relation Class.</td>
<td>10</td>
<td>26</td>
<td>12.8k</td>
<td>1.6k</td>
<td>1.6k</td>
</tr>
</tbody>
</table>
Table 2: F1-macro result on 2-layer MLP. Comparison among imbalance learning methods, weak supervision methods, and ARS2 (as well as its variants). The ranking of using confidence is identical to that ranked by negative loss. Note that ARS2 outperform all baselines in all imbalanced datasets.

(5) Effective Number + Logit Adjustment: Because the effective number is a class-wise re-weighting method, we have also combined this method with logit adjustment as a baseline.

Weak Supervision Methods: (1) COSINE (Yu et al., 2020): The COSINE method uses weakly labeled data to fine-tune pre-trained language models by contrastive self-training. (2) Denoiser (Ren et al., 2020): This method estimates the source reliability using a conditional soft attention mechanism and then reducing label noise by aggregating weak labels annotated by a set of rules.

3.3 Main Result

Our main results on a 2-layer MLP model are reported in Table 2. Our method outperformed all the baselines on both balanced and imbalanced datasets. The results of the experiment also indicated the following.

The performance of all baseline methods generally showed a downward trend with increasing imbalance ratio, and the decline was more evident in the binary classification problem of Yelp. In contrast, on the multi-classification problem dataset AgNews, although the head class and the tail class exhibited a relatively large gap because the number of each class declined step-by-step, the ratio between the head class and the second head class did not reach the value of the imbalance ratio. Therefore, the model’s performance decline on multi-classification was slightly smaller than in binary classification.

On the balanced dataset, the performance of ARS2 was very similar to that of using RR only, whereas the performance of ARS2 (w/o CR) was slightly better than that of ARS2 in AGNews. This occurred because CR did not operate as intended on the balanced dataset but instead affected the diversity of data, resulting in a small gap between ARS2 and RR on the balanced set. The role of CR became gradually more prominent with increasing imbalance ratio because CR was able to balance the original unbalanced sample set such that the model was not affected by the data from the majority class, improving performance.

The baseline performance gradually weakened with the increase in the imbalance ratio. These baselines were all loss-modification-related methods. In the noisy dataset, if a data point originally was not affected by the data from the majority class, whereas the performance of ARS2 (w/o CR) was slightly better than that of ARS2 in AGNews. This occurred because CR did not operate as intended on the balanced dataset but instead affected the diversity of data, resulting in a small gap between ARS2 and RR on the balanced set. The role of CR became gradually more prominent with increasing imbalance ratio because CR was able to balance the original unbalanced sample set such that the model was not affected by the data from the majority class, improving performance.

3.4 Ablation Study

We also examined the importance of CR, RR, and different measurements of label quality. The 8th, 9th and 11th rows of Table 2 summarize the results. It may be observed that the performance of ARS2 without CR on the balanced set was almost the same as that of ARS2, which shows that RR was the main contributor to the balanced set. However, the contributions of both start to converge with increasing imbalance ratio. This shows that the
data filtered by CR can allow the model to learn more balance features from the imbalance dataset, whereas RR can make the process smoother and reduce the model’s misjudgment of high-ranking data.

We also compared the performance of two ranking scores. The 11th and 12th rows of Table 2 show that ARS2 with PMS performed better than with confidence, except for the Yelp dataset with an imbalance ratio of 1. However, the difference in performance in this situation was less than 1%. This shows that PMS can reflect the cleanliness of data better, and thus may be considered a more useful ranking score.

3.5 Hyperparameter Study

We studied two hyperparameters, namely $\gamma$ and $\xi$ used in controlling the quality of adaptive sampling. We set the linear ratio from 1 to 10, where higher values of the linear ratio indicate larger amounts of data in the sampled batch. We also set the threshold from -0.5 to 0.5, which indicated the quality of the data of our sampled batch. Figures 5(a) shows that ARS2 is insensitive to $\gamma$ as the performance is less than 0.8%. In the case of Yelp, ARS2 cannot perform well with a small threshold (less than 0) because the loose threshold may lead a large amount of noisy data to affect the training process. While in the case of AGNews, the high quality of rules helps RR to re-corrected the noisy label, increase the clean data points accessed by model during training.

3.6 Quality Analysis of Sampled Data

Figure 3 and Figure 4 illustrate the cleanliness of the selected data from AGNews and Yelp at each step. It may be observed that ARS2 performed similarly to ARS2 (w/o RR) on the balanced set, which means that combining CR and RR can ad-
dress the problem of decreasing cleanliness caused by RR. The results also show that there was noise in the judgment of rules, and using RR alone could not eliminate the weak labels with noise; rather, it further increased the noise of the data. ARS2 generally outperforms a single method with increasing values of the imbalance ratio because the model learns the features of clean data and uses these features to filter out more clean data, gradually improving the cleanliness of the dataset and improving training performance.

3.7 Performance on Fine-tuning Pre-trained Model

From Table 3 it may be observed that our optimized RoBERTa training process was able to achieve state-of-the-art results. On the Yelp dataset with $\rho = 50$, all the baselines could not be fitted well, but the data filtered by our method enable RoBERTa to fit well even under such extreme conditions. This occurred because the fine-tuning process of RoBERTa only requires a small amount of data. Hence, we used warmed-up MLP to dynamically choose a small amount of clean and balanced data for training according to the batch size of RoBERTa to achieve good results. When training RoBERTa, we also linearly increased the training set to expose the model to a greater diversity of data.

<table>
<thead>
<tr>
<th>Method (↓)</th>
<th>AGNews (Imbalance Ratio (↓))</th>
<th>Yelp (Imbalance Ratio (↓))</th>
<th>TREC</th>
<th>Chemprot</th>
</tr>
</thead>
<tbody>
<tr>
<td>CE+LA</td>
<td>0.858 ± 0.002</td>
<td>0.847 ± 0.004</td>
<td>0.833 ± 0.009</td>
<td>0.823 ± 0.030</td>
</tr>
<tr>
<td>CE-EN</td>
<td>0.859 ± 0.002</td>
<td>0.841 ± 0.005</td>
<td>0.852 ± 0.006</td>
<td>0.823 ± 0.020</td>
</tr>
<tr>
<td>CE+EN+LAM</td>
<td>0.861 ± 0.001</td>
<td>0.851 ± 0.004</td>
<td>0.858 ± 0.007</td>
<td>0.809 ± 0.051</td>
</tr>
<tr>
<td>Dice</td>
<td>0.874 ± 0.001</td>
<td>0.875 ± 0.002</td>
<td>0.865 ± 0.005</td>
<td>0.800 ± 0.064</td>
</tr>
<tr>
<td>LDAM</td>
<td>0.859 ± 0.002</td>
<td>0.823 ± 0.007</td>
<td>0.815 ± 0.004</td>
<td>0.856 ± 0.009</td>
</tr>
<tr>
<td>CONE</td>
<td>0.874 ± 0.003</td>
<td>0.852 ± 0.013</td>
<td>0.816 ± 0.015</td>
<td>0.825 ± 0.012</td>
</tr>
<tr>
<td>DenseCE</td>
<td>0.868 ± 0.003</td>
<td>0.802 ± 0.002</td>
<td>0.801 ± 0.024</td>
<td>0.246 ± 0.096</td>
</tr>
<tr>
<td>ASR2</td>
<td>0.893 ± 0.007</td>
<td>0.890 ± 0.011</td>
<td>0.808 ± 0.026</td>
<td>0.851 ± 0.024</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Dice 0.874</th>
<th>CONE 0.874</th>
<th>DenseCE 0.868</th>
<th>ASR2 0.893</th>
<th>CE+EN 0.859</th>
<th>CE-EN 0.859</th>
<th>CE+EN+LAM 0.861</th>
<th>CE+LA 0.858</th>
<th>LDAM 0.859</th>
<th>- 0.874</th>
<th>- 0.874</th>
<th>- 0.874</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.874</td>
<td>0.874</td>
<td>0.868</td>
<td>0.893</td>
<td>0.859</td>
<td>0.859</td>
<td>0.861</td>
<td>0.858</td>
<td>0.859</td>
<td>0.874</td>
<td>0.874</td>
<td>0.874</td>
</tr>
<tr>
<td></td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
<td>0.007</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
<td>0.005</td>
<td>0.002</td>
<td>0.003</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>±0.003</td>
<td>±0.003</td>
<td>±0.003</td>
<td>±0.007</td>
<td>±0.002</td>
<td>±0.002</td>
<td>±0.001</td>
<td>±0.005</td>
<td>±0.002</td>
<td>±0.003</td>
<td>±0.003</td>
<td>±0.003</td>
</tr>
</tbody>
</table>

Table 3: F1-macro result on RoBERTa. Comparison among imbalance learning methods, weak supervision methods, and ASR2 (as well as its variants). Note that ARS2 outperform all baselines in all imbalanced datasets.

Class Imbalance Learning. Four primary methods has been proposed to solve data imbalance problem, including post-hoc correction, loss weighting, data modification and margin modification. **Post-hoc correction** modifies the logit computed by the model by using the prior of data to bias the training process towards fewer classes with fewer data (Fawcett and Provost, 1996; Provost, 2000; Maloof, 2003; King and Zeng, 2001; Coltell et al., 2016; Kim and Kim, 2020; Kang et al., 2019). **Loss weighting** weights the loss by the prior distribution of the training set, so that the data of the minority class exhibits a larger loss, and thus the model learn more bias towards the minority class to balance the minority and majority classes (Xie and Manski, 1989; Morik et al., 1999; Menon et al., 2013; Cui et al., 2019; Fan et al., 2017). **Data modification** balances a training set by increasing the number of data samples of minority classes or decreasing the data for majority classes so that the model is trained without favoring the majority classes or ignoring the minority classes (Kubat et al., 1997; Wallace et al., 2011; Chawla et al., 2002). **Margin modification** balances the minority and majority classes by increasing the margin of
minority class data and majority class data, which makes it easier for a model to learn a discriminative, robust decision boundary between the minority and majority class data (Masnadi-Shirazi and Vasconcellos, 2010; Iranmehr et al., 2019; Zhang et al., 2017; Cao et al., 2019b; Tan et al., 2020). In the case of training on a noisy training set, adding weights or adding data may give incorrect results, causing the model to learn to be biased towards noise.

5 Conclusion

In this article, we have proposed ARS2 as a method based on the WS paradigm to reduce both noise and the impact of natural imbalances in data. We have proposed PMS to evaluate the level of noise in training data. To reduce the impact of noisy data on training, we have proposed two ranking strategies based on PMS, including CR and RR. Finally, adaptive sampling is performed on the data based on this ranking to clean the data. We have also presented experimental results on eight different datasets, which demonstrate that ARS2 outperformed traditional WS and loss modification methods.

6 Limitation

First, because the presented work is focused on adaptive clean data sampling, we use the MLP as a teacher model for a large language model like RoBERTa. In the future research, we can consider using co-teaching methods (Han et al., 2018), which provide a more efficient teacher-student structure to train large models, to further improve the efficiency of teacher model. Also, due to the lack of computational resources, we only used a 2-layer MLP and RoBERTa as our backbone model. A larger language model like RoBERTa-large (Liu et al., 2019) could be considered in future work. Finally, the proposed approach could be extended to other tasks such as sequence labeling or natural language inference.

References


Shingo Takamatsu, Issei Sato, and H. Nakagawa. 2012. Reducing wrong labels in distant supervision for relation extraction. In ACL.


Quanming Yao, Honsi Yang, Bo Han, Gang Niu, and James Tin-Yau Kwok. 2020. Searching to exploit memorization effect in learning with noisy labels. In International Conference on Machine Learning, pages 10789–10798. PMLR.


Jieyu Zhang, Bohan Wang, Xiangchen Song, Yujing Wang, Yaming Yang, Jing Bai, and Alexander Ratner. 2022b. Creating training sets via weak indirect supervision. In ICLR.


A Datasets Details

A.1 Data Source

We use the data from WRENCH benchmark (Zhang et al., 2021).

AGNews: Dataset is available at https://drive.google.com/drive/u/1/folders/1VFJeVCvckD5-qAd5Sdl4n44JoryiEun.

Yelp: The raw dataset is available at https://huggingface.co/datasets/yelp_review_full. The preprocessed dataset is available at https://drive.google.com/drive/u/1/folders/1VFJeVCvckD5-qAd5Sdl4n44JoryiEun.

TREC, Chemprot: The preprocessed dataset is available at https://drive.google.com/drive/u/1/folders/1VFJeVCvckD5-qAd5Sdl4n44JoryiEun.

For these two datasets, we design eight different imbalance ratios to evaluate all methods. The details are shown in Table 4 and the label distribution for each dataset are shown in Fig 6.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Imbalance Ratio ρ (%)</th>
<th>Training Noise</th>
<th>#Train</th>
<th>#Train (Covered)</th>
<th>#Valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGNews</td>
<td>1 18.4% 96.0k 66.3k 12.0k</td>
<td>12.8% 42.7k 30.5k 5.4k</td>
<td>20 12.9% 37.3k 26.9k 4.7k</td>
<td>50 12.5% 32.8k 23.7k 4.1k</td>
<td></td>
</tr>
<tr>
<td>Yelp</td>
<td>1 29.8% 30.4k 25.2k 3.8k</td>
<td>22.3% 16.8k 13.2k 2.0k</td>
<td>20 18.1% 16.0k 12.5k 1.9k</td>
<td>50 10.2% 15.6k 12.1k 1.9k</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Weakly Annotated Imbalance Dataset Statistics.

B Details on Implementation and Experiment Setups

B.1 Computing Infrastructure

System: Windows Subsystem Linux 2; Python 3.8; PyTorch 1.9.
CPU: Intel(R) Core(TM) i9-12900K CPU.
GPU: NVIDIA RTX 3090.

B.2 Number of Parameters

ARS2 and all baselines use 2-layer MLP and Roberta-base (Liu et al., 2019) with a task-specific classification head on the top as the backbone. The 2-layer MLP has 128 neural unit in each layer, the total parameters are 128 × 128 × number of classes. The Roberta-base model contains 125M trainable parameters, and we fine-tune the last 4 layers with token size 512. We do not introduce any other parameters in our experiments.

B.3 Experiment Setups

All of our methods and baselines are run with 5 different random seeds and the result is based on the average performance on them. This indeed creates 8 (the number of datasets with four imbalance ratios) × 5 (the number of random seeds) × 12 (the number of methods) × 2 (the number of end models, MLP and RoBERTa) = 960 experiments for fine-tuning, which is almost the limit of our computational resources, not to mention grid search for hyperparameter for each method. We have shown both the mean and the standard deviation of the performance criteria in our experiment sections.

B.4 Implementations Baselines

For these three methods listed below, since they are mainly used in CV tasks; thus the code is hard to directly used for our experiments. We re-implement these methods based on their implementations in WRENCH codebase.

EN: https://github.com/richardaecn/class-balanced-loss.
LA: https://github.com/google-research/google-research/tree/master/logit_adjustment.
LDAM: https://github.com/kaidic/LDAM-DRW.

For these two weak supervision baselines listed below, we use the implementation provided by WRENCH.


Our implementation of ARS2 will be published upon acceptance.

B.5 Hyperparameters for General Experiments

We use AdamW (Loshchilov and Hutter, 2019) as the optimizer, and the learning rate of 2-layer MLP is chosen from 1 × 10−5 to 1 × 10−1, and 1 × 10−5, 3 × 10−5, 1 × 10−6, 3 × 10−6 for RoBERTa. Dropout rate of 2-layer MLP is chosen from 0, 0.2. We set weight decay rate as 0 and batch size as 128 for 2-layer MLP, and weight decay rate as 1 × 10−4 and batch size as 16 for
RoBERTa. We warm-up and continual training until early stop, and evaluate 2-layer MLP in every 100 steps and RoBERTa every 5 steps. Finally, we use the model with the best performance on the development set for testing. The general hyperparameters we use are shown in Table 5.

### B.6 Hyperparameters for ARS2

The hyperparameter of ARS2 includes $\gamma, \xi$ thus it does not require heavy hyperparameter tuning. In our experiments, we search $\gamma$ from 1 to 10, and $\xi$ from $-0.5$ to 0.5.

### B.7 Hyperparameters for Loss Modification Baselines

For loss modification methods, we mainly tune their key hyperparameters. For EN (Cui et al., 2019), we tune the number for effective number $\beta$ from $[0.9, 0.99, 0.999, 0.9999]$, $\gamma$ from $[0.5, 1, 2]$ and report the best performance. For LA (Menon et al., 2020), they use $\tau$ to scale the prior ratio. But in our experiment, we directly set $\tau$ as 1 to achieve LA’s performance as efficiently as possible. For Dice (Li et al., 2019), it use $\alpha$ to scale the $(1-p)$ factor avoiding it become too large, we search $\alpha$ from 0.1 to 1. $\gamma_{Dice}$ is designed to help numerator and denominator become smooth, we search $\gamma_{Dice}$ from $1 \times 10^{-4}$ to 1. Denominator square is designed to control the size of denominator, we search this hyperparameter in [True, False]. For LDAM (Cao et al., 2019a), we search max margin from 0.1 to 0.9 and $s$ from 1 to 30.

### B.8 Hyperparameters for Weak Supervision Baselines

For COSINE (Yu et al., 2020), we search learning rate from $[1 \times 10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-6}, 3 \times 10^{-6}]$, teacher model update frequency from $[50, 100, 200]$, regularize power scale from $[0.01, 0.05, 0.1]$ and margin threshold $\gamma_C$ from $[0.1, 0.3, 0.5, 0.7, 0.9]$. For Denoise (Ren et al., 2020), we search learning rate from $[1 \times 10^{-6}]

Table 5: Hyperparameter configurations for all baselines.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Hyper-parameter</th>
<th>Agnews (Imbalance Ratio (↓))</th>
<th>Yelp (Imbalance Ratio (↓))</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Batch Size</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>$\gamma$</td>
<td>3 10 1 1 1 1 1 1</td>
<td>0.2 -0.3 0 -0.2 0 -0.1 -0.1 -0.3</td>
</tr>
<tr>
<td></td>
<td>$\xi$</td>
<td>-0.2 0.0 0.0 0.2 0.2 0.0 0.0 0.0</td>
<td></td>
</tr>
<tr>
<td>CE+LA</td>
<td>Dropout Ratio</td>
<td>0.2 0.2 0.0 0.2 0.2 0.0 0.0 0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning Rate</td>
<td>0.001 0.001 0.003 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td>CE+EN</td>
<td>Dropout Ratio</td>
<td>0.2 0.0 0.0 0.0 0.2 0.0 0.0 0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning Rate</td>
<td>0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\beta_{EN}$</td>
<td>0.9999 0.9999 0.9 0.99 0.9999 0.9999 0.9999 0.9999</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\gamma_{EN}$</td>
<td>2.0 0.5 1.0 2.0 1.0 0.5 1.0 0.5</td>
<td></td>
</tr>
<tr>
<td>CE+EN+LA</td>
<td>Dropout Ratio</td>
<td>0.0 0.2 0.2 0.2 0.2 0.0 0.0 0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning Rate</td>
<td>0.0001 0.001 0.001 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\beta_{EN}$</td>
<td>0.9999 0.9999 0.9999 0.9999 0.9999 0.9999 0.9999 0.9999</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\gamma_{EN}$</td>
<td>2.0 0.5 0.5 0.5 2.0 1.0 0.5 0.5</td>
<td></td>
</tr>
<tr>
<td>Dice</td>
<td>Dropout Ratio</td>
<td>0.0 0.2 0.0 0.2 0.2 0.0 0.0 0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning Rate</td>
<td>0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\gamma_{Dice}$</td>
<td>0.005 1.0 0.268 0.0005 1.0 0.268 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Denominator Square</td>
<td>False True True False True False False False</td>
<td></td>
</tr>
<tr>
<td>LDAM</td>
<td>Dropout Ratio</td>
<td>0.2 0.0 0.2 0.0 0.2 0.2 0.2 0.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Learning Rate</td>
<td>0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>max margin</td>
<td>0.7 0.8 0.9 0.2 0.8 0.6 0.6 0.3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$s$</td>
<td>4.0 1.0 1.0 1.0 16.0 4.0 1.0 1.0</td>
<td></td>
</tr>
<tr>
<td>COSINE</td>
<td>Learning Rate</td>
<td>1e-5 1e-5 3e-5 1e-5 3e-5 1e-5 3e-5 3e-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$T_i$</td>
<td>100 50 50 200 200 100 50 200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\lambda$</td>
<td>0.1 0.1 0.1 0.1 0.1 0.1 0.05 0.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\gamma_C$</td>
<td>0.3 0.9 0.9 0.7 0.3 0.9 0.9 0.3</td>
<td></td>
</tr>
<tr>
<td>Denoise</td>
<td>Learning Rate</td>
<td>1e-5 3e-6 1e-6 3e-6 3e-5 1e-5 3e-5 3e-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Hidden Size</td>
<td>512 512 256 512 128 64 256 256</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\alpha$</td>
<td>0.001 0.001 0.001 0.001 0.001 0.001 0.001 0.001</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$c_1$</td>
<td>0.1 0.1 0.3 0.3 0.7 0.3 0.5 0.1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$c_2$</td>
<td>0.7 0.7 0.3 0.3 0.1 0.5 0.1 0.5</td>
<td></td>
</tr>
<tr>
<td>Dataset</td>
<td>Original (Covered)</td>
<td>Imbalance Ratio 10 (Covered)</td>
<td>Imbalance Ratio 20 (Covered)</td>
</tr>
<tr>
<td>-----------</td>
<td>--------------------</td>
<td>------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>AGNews</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yelp</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chemprot</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TREC</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6: Label statistic of each dataset.

$10^{-5}, 3 \times 10^{-5}, 1 \times 10^{-6}, 3 \times 10^{-6}$, denoiser hidden size from $[64, 128, 256, 512]$ and $c_1, c_2$ from $[0.1, 0.3, 0.5, 0.7, 0.9]$. 

1655