# Denoising Multi-Source Weak Supervision for Neural Text Classification

**Wendi Ren[1], Yinghao Li[1], Hanting Su[2], David Kartchner[1], Cassie Mitchell[1]** and **Chao Zhang[1]**
[1] Georgia Institute of Technology, Atlanta, USA
[2] Renmin University, Beijing, China
{wren44, yinghaoli, david.kartchner, chaozhang}@gatech.edu
suhanting@ruc.edu.cn
cassie.mitchell@bme.gatech.edu

## Abstract

We study the problem of learning neural text classifiers without using any labeled data, but only easy-to-provide rules as multiple weak supervision sources. This problem is challenging because rule-induced weak labels are often noisy and incomplete. To address these two challenges, we design a label denoiser, which estimates the source reliability using a conditional soft attention mechanism and then reduces label noise by aggregating rule-annotated weak labels. The denoised pseudo labels then supervise a neural classifier to predicts soft labels for unmatched samples, which address the rule coverage issue. We evaluate our model on five benchmarks for sentiment, topic, and relation classifications. The results show that our model outperforms state-of-the-art weakly-supervised and semi-supervised methods consistently, and achieves comparable performance with fully-supervised methods even without any labeled data. Our code can be found at https://github.com/weakrules/Denoise-multi-weak-sources.

## 1 Introduction

Many NLP tasks can be formulated as text classification problems, such as sentiment analysis (Badjatiya et al., 2017), topic classification (Zhang et al., 2015), relation extraction (Krebs et al., 2018) and question answering like slot filling (Pilehvar and Camacho-Collados, 2018). Recent years have witnessed the rapid development of deep neural networks (DNNs) for this problem, from convolutional neural network (CNN, Kim, 2014; Kalchbrenner et al., 2014), recurrent neural network (RNN, Lai et al., 2015) to extra-large pre-trained language models (Devlin et al., 2019; Dai et al., 2019; Liu et al., 2019). DNNs' power comes from their capabilities of fitting complex functions based on large-scale training data. However, in many scenarios, labeled data are limited, and manually annotating them at a large scale is prohibitively expensive.

Weakly-supervised learning is an attractive approach to address the data sparsity problem. It labels massive data with cheap labeling sources such as heuristic rules or knowledge bases. However, the major challenges of using weak supervision for text classification are two-fold: 1) the created labels are highly noisy and imprecise. The *label noise* issue arises because heuristic rules are often too simple to capture rich contexts and complex semantics for texts; 2) each source only covers a small portion of the data, leaving the labels incomplete. Seed rules have *limited coverage* because they are defined over the most frequent keywords but real-life text corpora often have long-tail distributions, so the instances containing only long-tail keywords cannot be annotated.

Existing works (Ratner et al., 2017; Meng et al., 2018; Zamani et al., 2018; Awasthi et al., 2020) attempt to use weak supervision for deep text classification. Ratner et al. (2017) proposes a data programming method that uses labeling functions to automatically label data and then trains discriminative models with these labels. However, data annotated in this way only cover instances directly matched by the rules, leading to limited model performance on unmatched data. Meng et al. (2018) proposes a deep self-training method that uses weak supervision to learn an initial model and updates the model by its own confident predictions. However, the self-training procedure can overfit the label noise and is prone to error propagation. Zamani et al. (2018) solves query performance prediction (QPP) by boosting multiple weak supervision signals in an unsupervised way. However, they choose the most informative labelers by an ad-hoc user-defined criterion, which may not generalize to all the domains. Awasthi et al. (2020) assumes that human labelers are over-generalized to increase the



*Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3739–3754

coverage, and they learn restrictions on the rules to address learning wrongly generalized labels. However, their method requires the specific formulation process of rules to indicate which rules are generated by which samples, so that it cannot deal with other kinds of labeling sources like knowledge bases or third-party tools.

We study the problem of using multiple weak supervision sources (*e.g.*, domain experts, pattern matching) to address the challenges in weakly-supervised text classification. While each source is weak, multiple sources can provide complementary information for each other. There is thus potential to leverage these multiple sources to infer the correct labels by estimating source reliability in different feature regimes and then aggregating weak labels. Moreover, since each source covers different instances, it is more promising to leverage multiple sources to bootstrap on unlabeled data and address the label coverage issue.

Motivated by the above, we propose a model with two reciprocal components. The first is a *label denoiser* with the conditional soft attention mechanism (Bahdanau et al., 2014) (§ 3.2). Conditioned on input text features and weak labels, it first learns reliability scores for labeling sources, emphasizing the annotators whose opinions are informative for the particular corpus. It then denoises rule-based labels with these scores. The other is a *neural classifier* that learns the distributed feature representations for all samples (§ 3.3). To leverage unmatched samples, it is supervised by both the denoised labels and its confident predictions on unmatched data. These two components are integrated into an end-to-end co-training framework, benefiting each other through cross-supervision losses, including the rule denoiser loss, the neural classifier loss, and the self-training loss(§ 3.4).

We evaluate our model on four classification tasks, including sentiment analysis, topic classification, spam classification, and information extraction. The results on five benchmarks show that: 1) the soft-attention module effectively denoises the noisy training data induced from weak supervision sources, achieving $84\%$ accuracy for denoising; and 2) the co-training design improves prediction accuracy for unmatched samples, achieving at least $9\%$ accuracy increase on them. In terms of the overall performance, our model consistently outperforms SOTA weakly supervised methods (Ratner et al., 2017; Meng et al., 2018; Zamani et al., 2018),

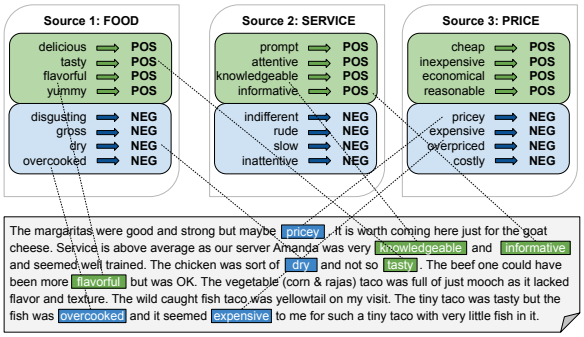


Figure 1: The annotation process for three weak supervision sources. "POS" and "NEG" are the labels for the sentiment analysis task.

semi-supervised method (Tarvainen and Valpola, 2017), and fine-tuning method (Howard and Ruder, 2018) by $5.46\%$ on average.

## 2 Preliminaries

### 2.1 Problem Definition

In *weakly supervised* text classification, we do not have access to clean labeled data. Instead, we assume external knowledge sources providing labeling rules as weak supervision signals.

**Definition 1** (Weak Supervision)**.** A weak supervision source specifies a set of labeling rules $\mathcal{R} = \{r_1, r_2, \ldots, r_k\}$. Each rule $r_i$ declares a mapping $f \rightarrow C$, meaning any documents that satisfy the feature $f$ are labeled as $C$.

We assume there are multiple weak supervision sources providing complementary information for each other. A concrete example is provided below.

**Example 1** (Multi-Source Weak Supervision)**.** Figure 1 shows three weak sources for the sentiment analysis of Yelp reviews. The sources use 'if-else' labeling functions to encode domain knowledge from different aspects. The samples that cannot be matched by any rules remain unlabeled.

**Problem Formulation** Formally, we have: 1) a corpus $\mathbf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ of text documents; 2) a set $\mathcal{C} = \{C_1, \ldots, C_m\}$ of target classes; and 3) a set $\mathcal{S} = \{\mathcal{R}_1, \mathcal{R}_2, \ldots, \mathcal{R}_k\}$ of weak annotators. Our goal is to learn a classifier from $\mathbf{D}$ with only multiple weak supervision sources to accurately classify any newly arriving documents.

### 2.2 Challenges

Although the use of automatic weak annotators largely reduces human labeling efforts, using rule-induced labeled data has two drawbacks: *label noise* and *label incompleteness*.

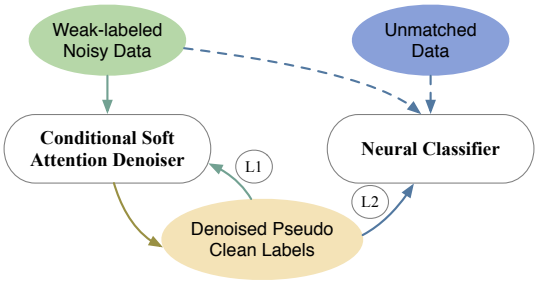


Figure 2: Overview of cross-training between the rule-based classifier and the neural classifier.

Weak labels are noisy since user-provided rules are often simple and do not fully capture complex semantics of the human language. In the Yelp example with eight weak supervision sources, the annotation accuracy is 68.3% on average. Label noise hurts the performance of text classifiers—especially deep classifiers—because such complex models easily overfit the noise. Moreover, the source coverage ranges from 6.8% to 22.2%. Such limited coverage is because user-provided rules are specified over common lexical features, but real-life data are long-tailed, leaving many samples unmatched by any labeling rules.

## 3 Our Method

We begin with an overview of our method and then introduce its two key components as well as the model learning procedure.

### 3.1 The Overall Framework

Our method addresses the above challenges by integrating weak annotated labels from multiple sources and text data to an end-to-end framework with a label denoiser and a deep neural classifier, illustrated in Figure 2.

**Label denoiser & self-denoising** We handle the label noise issue by building a label denoiser that iteratively denoises itself to improve the quality of weak labels. This label denoiser estimates the source reliability using a conditional soft attention mechanism, and then aggregates weak labels via weighted voting of the labeling sources to achieve "pseudo-clean" labels. The reliability scores are conditioned on both rules and document feature representations. They effectively emphasize the opinions of informative sources while down-weighting those of unreliable sources, thus making rule-induced predictions more accurate.

**Neural classifier & self-training** To address the low coverage issue, we build a neural classifier which learns distributed representations for text documents and classifies each of them, whether rule-matched or not. It is supervised by both the denoised weakly labeled data as well as its own high-confident predictions of unmatched data.

### 3.2 The Label Denoiser

When aggregating multiple weak supervision sources, it is key for the model to attend to more reliable sources, where source reliability should be conditioned on input features. This will enable the model to aggregate multi-source weak labels more effectively. Given $k$ labeling resources, we obtain the weak label matrix $\tilde{Y} \in \mathbb{R}^{n \times k}$ through rule matching. Specifically, as shown in the Rule Matching step of 3, by Definition 1, given one rule, if a document is matchable by that rule, it will be assigned with a rule-induced label C; otherwise, the document remains unlabeled, represented as -1. N rules thus generate N weak labels for each document. We then estimate the source reliability and aggregate complementary weak labels to obtain "pseudo-clean" labels.

**Parameterization of source reliability** We introduce a soft attention mechanism conditioned on both weak labels and feature representation, denoted as $\boldsymbol{B}$, to estimate the source reliability. Formally, we denote the denoised "pseudo-clean" labels by $\hat{Y} = [\hat{y_1}, \ldots, \hat{y_n}]^T$ , and the initial ones $\hat{Y_0}$ are obtained by simple majority voting from $\tilde{Y}$.

The core of the label denoiser is an attention net, a two-layer feed-forward neural network which predicts the attention score for matched samples. Formally, we specify a reliability score $a_j$ for each labeling source to represent its annotation quality, and the score is normalized to satisfy $\sum_{j=1}^{k} a_j = 1$. For one document $\boldsymbol{d_i}$, its attention score $q_{i,j}$ of one labeling source $\mathcal{R}_j$ is:

$$\begin{aligned} \hat{q}_{ij} =& W_2^T \tanh(W_1(\tilde{y}_{ij} + \boldsymbol{B}_i)), \\ q_{ij} =& \frac{\exp(\hat{q}_{ij})}{\sum_j \exp(\hat{q}_{ij})}, \end{aligned} \tag{1}$$

where $W_1, W_2$ denote the neural network weights and $\tanh$ is the activation function. Thus, for each document, its conditional labeling source score vector $\boldsymbol{A}_i = [a_{i1}, a_{i2}, \ldots, a_{ik}]^T$ is calculated over matched annotators as $a_{ij} = q_{ij}\chi_C(\tilde{y}_{ij} >= 0)$, where $\chi_C$ is the indicator function. Then, we average the conditional source score $\boldsymbol{A}_i$ over all the $n$ matched samples to get the source reliability vector $\boldsymbol{A}$. The weight of $j_{th}$ $(j = 1, 2, \ldots, k)$ annotator is

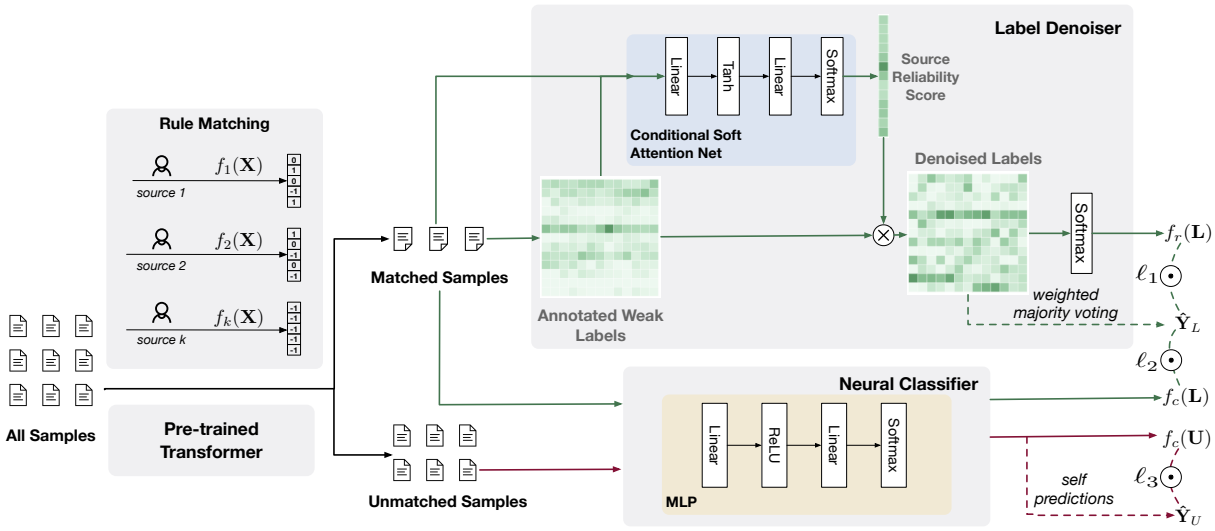


Figure 3: The detailed model architecture. Our model mainly consists of two parts: (1) the label denoiser, including the conditional soft attention reliability estimator and the instance-wise multiplication; (2) the neural classifier, which calculates sentence embedding using the pre-trained Transformer and makes classification.

calculated as $a_j = \frac{1}{n}\sum_{i=1}^{n} a_{ij}$. Finally, We aggregate $k$ reliability scores to get the reliability vector $\boldsymbol{A} = [a_1, a_2, \ldots, a_k]^T$.

**Denoising pseudo labels** With the learned reliability vector $\boldsymbol{A}$, we reweight the sources to get the weighted majority voted labels $\hat{Y}$ by $\tilde{Y}_i \otimes \boldsymbol{A}$. The denoised "pseudo-clean" label $\hat{y}_i$ is:

$$\hat{y}_i = \arg\max_{C_r} \sum_{j=1}^{k} a_j \chi_C(\tilde{y}_{ij} == C_r), \quad \text{where } r = 1, 2, \ldots, m. \tag{2}$$

The updated higher-quality labels $\hat{Y}$ then supervise the rule-covered samples in $\mathbf{D}_L$ to generate better soft predictions and guide the neural classifier later.

**Rule-based classifier prediction** At the epoch $t$ of our co-training framework, we learn the reliability score $\boldsymbol{A}(t)$ and soft predictions $\hat{\boldsymbol{Z}}(t)$ supervised by "pseudo-clean" labels from the previous epoch $\hat{Y}(t-1)$. Then we renew "clean-pseudo" labels as $\hat{Y}(t)$ using the score $\boldsymbol{A}(t)$ by (2).

Specifically, given $m$ target classes and $k$ weak annotators, the prediction probability $\hat{\boldsymbol{z}}_{\boldsymbol{i}}$ for $\boldsymbol{d}_{\boldsymbol{i}}$ is obtained by weighting the noisy labels $\tilde{Y}_i$ according to their corresponding conditional reliability scores $\boldsymbol{A}_i$: $\hat{\boldsymbol{z}}_{\boldsymbol{i}} = \text{softmax}(\tilde{Y}_i \otimes \boldsymbol{A}_i)$, where the masked matrix multiplication $\otimes$ (defined in (3)) is used to mask labeling sources that do not annotate document $i$, and we normalize the resultant masked scores via softmax:

$$y_{ir} = \sum_{j=1}^{k} a_{ij} \chi_C(\tilde{y}_{ij} == C_r)$$
$$\hat{\mathbf{z}}_{ir} = \frac{\exp(y_{ir})}{\sum_{r=1}^{m} \exp(y_{ir})}. \tag{3}$$

We finally aggregate $m$ soft adjusted scores to get the soft prediction vector $\hat{\boldsymbol{z}}_{\boldsymbol{i}} = [z_{i1}, \ldots, z_{im}]^T$.

### 3.3 The Neural Classifier

The neural classifier is designed to handle all the samples, including matched ones and unmatched ones. The unmatched corpus where the documents cannot be annotated by any source is denoted as $\mathbf{D}_U$. In our model, we use the pre-trained BERT (Devlin et al., 2019) as our feature extractor, and then feed the text embeddings $\mathbf{B}$ into a feed-forward neural network to obtain the final predictions. For $\boldsymbol{d}_i \in \mathbf{D}_L \cup \mathbf{D}_U$, the prediction $\tilde{\boldsymbol{z}}_{\boldsymbol{i}}$ is:

$$\tilde{\boldsymbol{z}}_{\boldsymbol{i}} = f_\theta(\boldsymbol{B}_i; \theta_w), \tag{4}$$

where $f_\theta$ denotes the two-layer feed-forward neural network, and $\theta_w$ denotes its parameters.

### 3.4 The Training Objective

**The rule denoiser loss** $\ell_1$ is the loss of the rule-based classifier over $\mathbf{D}_L$. We use the "pseudo-clean" labels $\hat{Y}$ to self-train the label denoiser and define the loss $\ell_1$ as the negative log likelihood of $\hat{y}_i$,

$$\ell_1 = -\sum_{i \in \mathbf{D}_L} \hat{y}_i \log \hat{\boldsymbol{z}}_{\boldsymbol{i}}. \tag{5}$$

**Algorithm 1** Training process of our model

**Require:** $\mathbf{D}_L$, $\mathbf{D}_U$, $\mathcal{C}$, $\boldsymbol{B}$, $\tilde{Y}$, $g_w(x)$ and $f_\theta(x)$: feed-forward rule-based and nerual classifier with trainable parameters $W$ and $\theta$; $s$: number of training iteraions;

$\hat{Y} \leftarrow \hat{Y}_0$, initialize by simple majority voting
**for** $t \leftarrow 1$ to $s$ **do**
  $\boldsymbol{A}, \hat{\boldsymbol{z}}_{i\in\mathbf{D}_L} \leftarrow g_w(\tilde{Y}_i, \boldsymbol{B}_i, \hat{y}_i)$ ▷ learn reliability score and evaluate attention network output supervised by "pseudo-clean" labels from (1) and (3)
  $\hat{y}_i \leftarrow$(2) ▷ renewed pseudo labels
  $\tilde{\boldsymbol{z}}_{i\in\mathbf{D}_L\cup\mathbf{D}_U} \leftarrow f_\theta(\boldsymbol{B}_i, \hat{y}_i)$ ▷ evaluate neural classifier output
  update $\theta, W$ using ADAM by (8)
**end for**
**return** $W, \theta$

**The neural classifier loss** $\ell_2$ is the loss of the neural classifier over $\mathbf{D}_L$. Similarly, we regard the negative log-likelihood from the neural network outputs $\tilde{\boldsymbol{Z}}$ to the pseudo-clean labels $\hat{Y}$ as training loss, formally

$$\ell_2 = -\sum_{i\in\mathbf{D}_L} \hat{y}_i \log \tilde{\boldsymbol{z}}_i. \tag{6}$$

**The unsupervised self-training loss** $\ell_3$ is the loss of the neural classifier over $\mathbf{D}_U$. To further enhance the label quality of $\mathbf{D}_U$ we apply the temporal ensembling strategy (Laine and Aila, 2016), which aggregates the predictions of multiple previous network evaluations into an ensemble prediction to alleviate noise propagation. For a document $\boldsymbol{d}_i \in \mathbf{D}_U$, the neural classifier outputs $\tilde{\boldsymbol{z}}_i$ are accumulated into ensemble outputs $\boldsymbol{Z}_i$ by updating $\boldsymbol{Z}_i \leftarrow \alpha\boldsymbol{Z}_i + (1-\alpha)\tilde{\boldsymbol{z}}_i$, where $\alpha$ is a term that controls how far the ensemble looks back into training history. We also need to construct target vectors by bias correction, namely $\boldsymbol{p}_i \leftarrow \boldsymbol{Z}_i/(1-\alpha^t)$, where $t$ is the current epoch. Then, we minimize the Euclidean distance between $\boldsymbol{p}_i$ and $\tilde{\boldsymbol{z}}_i$, where

$$\ell_3 = \sum_{i\in\mathbf{D}_U} \|\tilde{\boldsymbol{z}}_i - \boldsymbol{p}_i\|^2 \tag{7}$$

**Overall Objective** The final training objective is to minimize the overall loss $\ell$:

$$\ell = c_1\ell_1 + c_2\ell_2 + c_3\ell_3, \tag{8}$$

where $0 \le c_1 \le 1$, $0 \le c_2 \le 1$, and $0 \le c_3 \le 1$ are hyper-parameters for balancing the three losses and satisfy $c_1 + c_2 + c_3 = 1$.

| Dataset | Task | $C$ | #Train | #Dev | #Test | Cover | Acc. |
|---|---|---|---|---|---|---|---|
| youtube | Spam | 2 | 1k | 0.1k | 0.1k | 74.4 | 85.3 |
| imdb | Sentiment | 2 | 20k | 2.5k | 2.5k | 87.5 | 74.5 |
| yelp | Sentiment | 2 | 30.4k | 3.8k | 3.8k | 82.8 | 71.5 |
| agnews | Topic | 4 | 96k | 12k | 12k | 56.4 | 81.4 |
| spouse | Relation | 2 | 1k | 0.1k | 0.1k | 85.9 | 46.5 |

Table 1: Data Statistics. $C$ is the number of classes. Cover is fraction of rule-induced samples. Acc. refers to precision of labeling sources (number of correct samples / matched samples). Cover and Acc. are in %.

### 3.5 Model Learning and Inference

Algorithm 1 sketches the training procedure. Two classifiers provide supervision signals for both themselves and their peers, iteratively improving their classification abilities. In the test phase, the corpus is sent into our model with the corresponding annotated noisy labels. The final target $C_i$ for a document $i$ is predicted by ensembling the soft predictions. If two predictions from the label denoiser and the neural classifier conflict with each other, we choose the one with higher confidence, where the confidence scores are softmax outputs.

## 4 Experiments

### 4.1 Experimental Setup

**Datasets and tasks** We evaluate our model on five widely-used text classification datasets, covering four different text classification tasks: **youtube** (Alberto et al., 2015) (Spam Detection), **imdb** (Maas et al., 2011), **yelp** (Zhang et al., 2015) (Sentiment Analysis), **agnews** (Zhang et al., 2015) (Topic Classification), and **spouse** (Ratner et al., 2017) (Relation Classification). Table 1 shows the statistics of these datasets and the quality of weak labels (the details of each annotation rule are given in the appendix A.4). Creating such rules required very light efforts, but is able to cover a considerable amount of data samples (*e.g.*, 54k in agnews).

**Baselines** We compare our model with the following advanced methods: 1) **Snorkel** (Ratner et al., 2017) is a general weakly-supervised learning method that learns from multiple sources and denoise weak labels by a generative model; 2) **WeSTClass** (Meng et al., 2018) is a weakly-supervised text classification model based on self-training; 3) **ImplyLoss** (Awasthi et al., 2020) propose the rule-exemplar supervision and implication loss to denoise rules and rule-induced labels jointly; 4) **NeuralQPP** (Zamani et al., 2018) is a boosting prediction framework which selects useful

| Method | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| Snorkel | 78.6 | 73.2 | 69.1 | 62.9 | 56.9 |
| WeSTClass | 65.1 | 74.7 | 76.9 | 82.8 | 56.6 |
| Implyloss | 93.6 | 51.1 | 76.3 | 68.5 | 68.3 |
| NeuralQPP | 85.2 | 53.6 | 57.3 | 69.5 | 74.0 |
| MT | 86.7 | 72.9 | 71.2 | 70.6 | 70.7 |
| ULMFiT | 56.1 | 70.5 | 67.3 | 66.8 | 72.4 |
| BERT-MLP | 77.0 | 72.5 | 81.5 | 75.8 | 70.7 |
| Ours | **94.9** | **82.9** | **87.5** | **85.7** | **81.3** |

Table 2: Classification accuracy in the test set for all methods on five datasets.

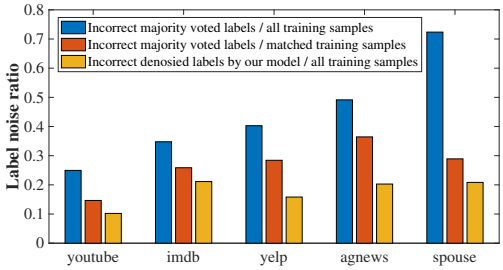


Figure 4: The label noise ratio of the initial majority voted labels and our denoised labels in the training set.

labelers from multiple weak supervision signals; 5) **MT** (Tarvainen and Valpola, 2017) is a semi-supervised model that uses Mean-Teacher method to average model weights and add a consistency regularization on the student and teacher model; and 6) **ULMFiT** (Howard and Ruder, 2018) is a strong deep text classifier based on pre-training and fine-tuning. 7) **BERT-MLP** takes the pre-trained Transformer as the feature extractor and stacks a multi-layer perceptron on its feature encoder.

## 4.2 Experimental Results

### 4.2.1 Comparison with Baselines

We first compare our method with the baselines on five datasets. For fair comparison, all the methods use a pre-trained BERT-based model for feature extraction, and use the same neural architecture as the text classification model. All the baselines use the same set of weak labels $\tilde{Y}$ for model training, except for WeSTClass which only requires seed keywords as weak supervision (we extract these keywords from the predicates of our rules).

Table 2 shows the performance of all the methods on five datasets. As shown, our model consistently outperforms all the baselines across all the datasets. Such results show the strength and robustness of our model. Our model is also very time-efficient (4.5 minutes on average) with trainable parameters only from two simple MLP neural networks (0.199M trainable parameters).

Similar to our methods, Snorkel, NeuralQPP, and Implyloss also denoise the weak labels from multiple sources by the following ideas: 1) Snorkel uses a generative modeling approach; 2) Implyloss adds one regularization to estimate the rule over-generalizing issue, but it requires the clean data to indicate which document corresponds to which rule. Without such information in our setting, this advanced baseline cannot perform well; 3) NeuralQPP selects the most informative weak labelers by boosting method. The performance gaps verify the effectiveness of the our conditional soft attention design and co-training framework.

WeSTClass is similar to our method in that it also uses self-training to bootstrap on unlabeled samples to improve its performance. The major advantage of our model over WeSTClass is that it uses two different predictors (rule-based and neural classifier) to regularize each other. Such a design not only better reduces label noise but also makes the learned text classifier more robust.

Finally, ULMFiT and BERT-MLP are strong baselines based on language model fine-tuning. MT is a well-known semi-supervised model which achieved inspiring results for image classification. However, in the weakly supervised setting, they do not perform well due to label noise. The results show that ULMFiT and MT suffer from such label noise, whereas our model is noise-tolerant and more suitable in weakly supervised settings. Overall BERT-MLP performs the best and we further compare it with ours in more perspectives.

### 4.2.2 Effectiveness of label denoising

To study the effectiveness of label denoising, we first compare the label noise ratio in training set given by the majority-voted pseudo labels ($\tilde{Y}$ defined in § 3.2) and our denoised pseudo labels. Figure 4 shows that after applying our denoising model, the label noise is reduced by 4.49% (youtube), 4.74% (imdb), 12.6% (yelp), 3.87% (agnews) and 8.06% (spouse) within the matched samples. If we count all the samples, the noise reduction is much more significant with 23.92% on average. Such inspiring results show the effectiveness of our model in denoising weak labels.

**Train a classifier with denoised labels** We further study how the denoised labels benefit the training of supervised models. To this end, we feed the labels generated by majority voting and denoised ones generated by our model into two state-of-the-

| Method | Labels | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|---|
| BERT+ MLP | major | 77.0 | 72.5 | 81.5 | 75.8 | 70.7 |
| | ours | **89.8** | **80.2** | **85.8** | **84.3** | **78.0** |
| UlmFit | major | 56.1 | 70.5 | 67.3 | 66.8 | 72.4 |
| | ours | **90.8** | **81.6** | **85.9** | **84.7** | **81.3** |

Table 3: Classification accuracy of two supervised methods with labels generated by majority voting and denoised ones generated by our model.

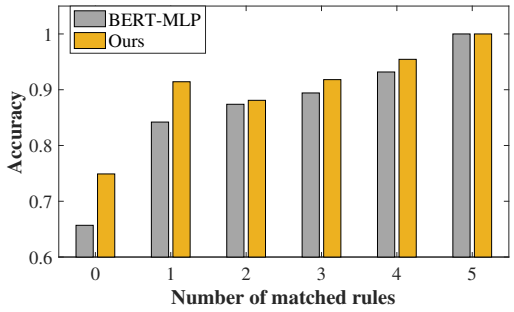


Figure 5: Accuracy on low-resource samples (matched by a small number of rules) in Youtube dataset.

art supervised models: ULMFiT and BERT-MLP (described in § 4.1). Table 3 shows that denoised labels significantly improve the performance of supervised models on all the datasets.

#### 4.2.3 Effectiveness of handling rule coverage

We proceed to study how effective our model is when dealing with the low-coverage issue of weak supervision. To this end, we evaluate the performance of our model for the samples covered by different numbers of rules. As shown in Figure 5, the strongest baseline (BERT-MLP) trained with majority-voted labels performs poorly on samples that are matched by few rules or even no rules. In contrast, after applying our model, the performance on those less matched samples improves significantly. This is due to the neural classifier in our model, which predicts soft labels for unmatched samples and utilizes the information from the multiple sources through co-training.

#### 4.2.4 Incorporating Clean Labels

We also study how our model can further benefit from a small amount of labeled data. While our model uses weak labels by default, it can easily incorporate clean labeled data by changing the weak labels to clean ones and fix them during training. We study the performance of our model in this setting, and compare with the fully-supervised BERT-MLP model trained with the same amount of clean labeled data.

| Labeled | Method | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|---|
| 0.5% | Bert-MLP | 80.6 | 76.9 | 86.2 | 82.6 | 68.2 |
| | Ours | **92.4** | **81.9** | **87.5** | **86.4** | **81.3** |
| 2% | Bert-MLP | 83.2 | 78.8 | 87.4 | 84.7 | 72.3 |
| | Ours | **92.9** | **83.1** | **87.6** | **85.7** | **81.3** |
| 5% | Bert-MLP | 87.7 | 83.6 | 89.0 | 86.4 | 74.8 |
| | Ours | **93.8** | **86.1** | **90.4** | **88.2** | **82.1** |
| 20% | Bert-MLP | 90.8 | 86.0 | 90.3 | **89.2** | 75.6 |
| | Ours | **94.0** | **86.1** | **90.5** | **89.2** | **84.5** |
| 50% | Bert-MLP | 91.8 | **86.2** | **90.5** | 89.2 | 78.0 |
| | Ours | **95.4** | **86.2** | **90.5** | **89.3** | **85.9** |
| 100% | Bert-MLP | 94.4 | 87.2 | 91.1 | 90.7 | 79.6 |

Table 4: The classification accuracy of BERT-MLP and our model with ground truth labeled data

As shown in Table 4, the results of combining our denoised labels with a small amount of clean labels are inspiring: it further improves the performance of our model and consistently outperforms the fully supervised BERT-MLP model. When the labeled ratio is small, the performance improvement over the fully-supervised model is particularly large: improving the accuracy by 6.28% with 0.5% clean labels and 3.84% with 5% clean labels on average. When the ratio of clean labels is large, the performance improvements becomes marginal.

The performance improvement over the fully-supervised model is relatively smaller on yelp and agnews datasets. The reason is likely that the text genres of yelp and agnews are similar to the text corpora used in BERT pre-training, making the supervised model fast achieve its peak performance with a small amount of labeled data.

#### 4.2.5 Ablation Study

We perform ablation studies to evaluate the effectiveness of the three components in our model: the label denoiser, the neural classifier, and the self-training over unmatched samples. By removing one of them, we obtain four settings: 1) Rule-only, represents w/o neural classifier and self-training; 2) Neural-only, represents w/o label denoiser and self-training; 3) Neural-self: represents w/o label denoiser; 4) Rule-Neural: represents w/o self training. 3) and 4) are supervised by the initial simple majority voted labels. Table 5 shows the results. We find that all the three components are key to our model, because: 1) the rule-based label denoiser iteratively obtains higher-quality pseduo labels from the weak supervision sources; 2) the neural classifier extracts extra supervision signals from unlabeled data through self-training.

| Method | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| Ours | **94.9** | **82.9** | **87.5** | **85.7** | **81.3** |
| Rule-only | 90.3 | 73.1 | 70.2 | 63.6 | 77.2 |
| Neural-only | 77.0 | 72.5 | 81.5 | 75.8 | 70.7 |
| Neural-self | 89.3 | 81.4 | 82.9 | 81.3 | 79.7 |
| Rule-Neural | 87.2 | 80.1 | 80.8 | 84.8 | 69.9 |

Table 5: Ablation Study Results.

#### 4.2.6 Case Study

We provide a example of Yelp dataset to show the denoising process of our model.

A reviewer of says "My husband tried this place. He was pleased with his experience and he wanted to take me there for dinner. We started with calamari which was so greasy we could hardly eat it...The bright light is the service. Friendly and attentive! The staff made an awful dining experience somewhat tolerable." The ground-truth sentiment should be NEGATIVE.

This review is labeled by three rules as follows: 1) keyword-mood, `pleased` $\rightarrow$ POSITIVE; 2) keyword-service, `friendly` $\rightarrow$ POSITIVE; 3) keyword-general, `awful` $\rightarrow$ NEGATIVE. The majority-voted label is thus POSITIVE, but it is wrong. After applying our method, the learned conditional reliability scores for the three rules are 0.1074, 0.1074, 0.2482, which emphasizes rule 3) so the denoised weighted majority voted is thus NEGATIVE, and it becomes correct.

#### 4.2.7 Parameter Study

The primary parameters of our model include: 1) the dimension of hidden layers $d_{\text{h}}$ in the label denoiser and the feature-based classifier; 2) learning rate $lr$; 3) the weight $c_1$, $c_2$, and $c_3$ of regularization term for $\ell_1$, $\ell_2$, and $\ell_3$ in (8); 4) We fix momentum term $\alpha = 0.6$ followed the implementation of Laine and Aila (2016). By default, we set $d_{\text{h}} = 128$, $lr = 0.02$, and $c_1 = 0.2, c_2 = 0.7, c_3 = 0.1$ as our model achieves overall good performance with these parameters. The search space of $d_{\text{h}}$ is $2^{6-9}$, $lr$ is $0.01 - 0.1$, $c_1$ and $c_3$ are $0.1 - 0.9$ (note that $c_2 = 1 - c_1 - c_3$). The hyperparameter configuration for the best performance reported in Table 2 is shown in the appendix A.3.

We test the effect of one hyperparameter by fixing others to their default values. In Figure 6 (a) and (b), we find the performance is stable except that the loss weight is too large. For (c) and (d), except for the spouse dataset when $lr$ is too small and $d_{\text{h}}$ is too large (instability due to the dataset size is small), our model is robust to the hyperparameters when they are in a reasonable range. We also report overall performance for all the search trails in Table 10 of appendix A.3.

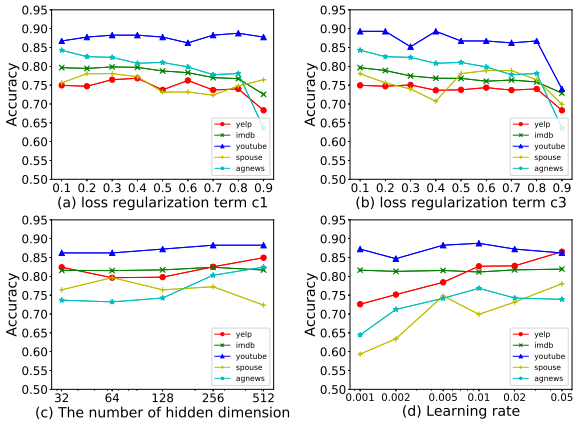


Figure 6: The prediction accuracy over different parameter settings.

## 5 Related Work

**Learning from Noisy Supervision.** Our work is closely related to existing work on learning from noisy supervision. To deal with label noise, several studies (Brodley and Friedl, 1999; Smith and Martinez, 2011; Yang et al., 2018) adopt a data cleaning approach that detects and removes mislabeled instances. This is achieved by outlier detection (Brodley and Friedl, 1999), a-priori heuristics (Smith and Martinez, 2011), self-training (Liang et al., 2020), or reinforcement learning (Yang et al., 2018; Zhang et al., 2020). One drawback of this data cleaning approach is that it can discard many samples and incur information loss.

Different from data cleaning, some works adopt a data correction approach. The most prominent idea in this line is to estimate the noise transition matrix among labels (Sukhbaatar and Fergus, 2014; Sukhbaatar et al., 2014; Goldberger and Ben-Reuven, 2016; Wang et al., 2019; Northcutt et al., 2019) and then use the transition matrices to re-label the instances or adapt the loss functions. Specifically, Wang et al. (2019) and Northcutt et al. (2019) generate label noise by flipping clean labels based on such noise transition matrices. They are thus not applicable to our weak supervision setting where no clean labels are given. Meanwhile, reweighting strategies have been explored to adjust the input training data. These techniques weigh training samples according to the predictions confidence (Dehghani et al., 2017), one-sided noise assumption (Zhang et al., 2019), a clean set (Ren

et al., 2018) or the similarity of their descent directions (Yang et al., 2018). Recently, a few studies (Veit et al., 2017; Hu et al., 2019) have also explored designing denoising modules for neural networks. However, our method differs from them in that: (1) our method learns *conditional reliability scores* for multiple sources; and (2) these methods still require clean data for denoising, while ours does not.

**Learning from Multi-Source Supervision** The crowdsourcing area also faces the problem of learning from multiple sources (*i.e.*, crowd workers). Different strategies have been proposed to integrate the annotations for the same instance, such as estimating the confidence intervals for workers (Joglekar et al., 2015) or leveraging approval voting (Shah et al., 2015). Compared with crowdsourcing, our problem is different in that the multiple sources provide only feature-level noisy supervision instead of instance-level supervision.

More related to our work are data programming methods (Ratner et al., 2016, 2017, 2019) that learn from multiple weak supervision sources. One seminal work in this line is Snorkel (Ratner et al., 2017), which treats true labels as latent variables in a generative model and weak labels as noisy observations. The generative model is learned to estimate the latent variables, and the denoised training data are used to learn classifiers. Our approach differs from data programming methods where we use a soft attention mechanism to estimate source reliability, which is integrated into neural text classifiers to improve the performance on unmatched samples.

**Self-training** Self-training is a classic technique for learning from limited supervision (Yarowsky, 1995). The key idea is to use a model's confident predictions to update the model itself iteratively. However, one major drawback of self-training is that it is sensitive to noise, *i.e.*, the model can be mis-guided by its own wrong predictions and suffer from error propagation (Guo et al., 2017).

Although self-training is a common technique in semi-supervised learning, only a few works like WeSTClass (Meng et al., 2018) have applied it to weakly-supervised learning. Our self-training differs from WeSTClass in two aspects: 1) it performs weighted aggregation of the predictions from multiple sources, which generates higher-quality pseudo labels and makes the model less sensitive to the error in one single source; 2) it uses temporal ensembling, which aggregates historical pseudo labels and alleviates noise propagation.

## 6 Conclusion

We have proposed a deep neural text classifier learned not from excessive labeled data, but only unlabeled data plus weak supervisions. Our model learns from multiple weak supervision sources using two components that co-train each other: (1) a label denoiser that estimates source reliability to reduce label noise on the matched samples, (2) a neural classifier that learns distributed representations and predicts over all the samples. The two components are integrated into a co-training framework to benefit from each other. In our experiments, we find our model not only outperforms state-of-the-art weakly supervised models, but also benefits supervised models with its denoised labeled data. Our model makes it possible to train accurate deep text classifiers using easy-to-provide rules, thus appealing in low-resource text classification scenarios. As future work, we are interested in denoising the weak supervision further with automatic rule discovery, as well as extending the co-training framework to other tasks beyond text classification.

## Acknowledgments

This work was supported in part by the National Science Foundation award III-2008334, Amazon Faculty Award, and Google Faculty Award.

## References

Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. 2015. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 138–143. IEEE.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from rules generalizing labeled exemplars. In *International Conference on Learning Representations*.

Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. 2017. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 759–760, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. Cite arxiv:1409.0473 Comment: Accepted at ICLR 2015 as oral presentation.

C. E. Brodley and M. A. Friedl. 1999. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Mostafa Dehghani, Aliaksei Severyn, Sascha Rothe, and Jaap Kamps. 2017. Avoiding your teacher's mistakes: Training neural networks with controlled weak supervision. *CoRR*, abs/1711.00313.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Goldberger and Ehud Ben-Reuven. 2016. Training deep neural-networks using a noise adaptation layer.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the International Conference on Machine Learning*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Mengying Hu, Hu Han, Shiguang Shan, and Xilin Chen. 2019. Weakly supervised image classification through noise regularization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11517–11525.

Manas Joglekar, Hector Garcia-Molina, and Aditya G. Parameswaran. 2015. Comprehensive and reliable crowd assessment algorithms. In *Proceedings of the IEEE International Conference on Data Engineering*, pages 195–206.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Alicia Krebs, Alessandro Lenci, and Denis Paperno. 2018. SemEval-2018 task 10: Capturing discriminative attributes. In *Proceedings of The 12th International Workshop on Semantic Evaluation*, pages 732–740, New Orleans, Louisiana. Association for Computational Linguistics.

Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2267–2273. AAAI Press.

Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, page 1054–1064, New York, NY, USA. Association for Computing Machinery.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. 2018. Weakly-supervised neural text classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, pages 983–992. ACM.

Curtis G. Northcutt, Lu Jiang, and Isaac L. Chuang. 2019. Confident learning: Estimating uncertainty in dataset labels. *CoRR*, abs/1911.00068.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2019. Training complex models with multi-task weak supervision. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 4763–4771.

Alexander J Ratner, Christopher De Sa, Sen Wu 0002, Daniel Selsam, and Christopher Ré. 2016. Data programming - creating large training sets, quickly. In *Proceedings of the Annual Conference on Neural Information Processing Systems*.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4331–4340. PMLR.

Nihar B. Shah, Dengyong Zhou, and Yuval Peres. 2015. Approval voting and incentives in crowdsourcing. In *Proceedings of the International Conference on Machine Learning*, volume 37, pages 10–19.

M. R. Smith and T. Martinez. 2011. Improving classification accuracy by identifying and removing instances that should be misclassified. In *International Joint Conference on Neural Networks*.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2014. Training convolutional networks with noisy labels. *arXiv:1406.2080 [cs]*.

Sainbayar Sukhbaatar and Rob Fergus. 2014. Learning from noisy labels with deep neural networks. In *Proceedings of the International Conference on Learning Representations*.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pages 1195–1204. Curran Associates, Inc.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6575–6583.

Hao Wang, Bing Liu, Chaozhuo Li, Yan Yang, and Tianrui Li. 2019. Learning with noisy labels for sentence-level sentiment classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 6285–6291. Association for Computational Linguistics.

Yaosheng Yang, Wenliang Chen, Zhenghua Li, Zhengqiu He, and Min Zhang. 2018. Distantly supervised ner with partial annotation learning and reinforcement learning. In *Proceedings of the International Conference on Computational Linguistics*, pages 2159–2169.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 189–196.

Hamed Zamani, W Bruce Croft, and J Shane Culpepper. 2018. Neural query performance prediction using weak supervision from multiple signals. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 105–114.

Rongzhi Zhang, Yue Yu, and Chao Zhang. 2020. Seqmix: Augmenting active sequence labeling via sequence mixup. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.

Zhen-Yu Zhang, Peng Zhao, Yuan Jiang, and Zhi-Hua Zhou. 2019. Learning from incomplete and inaccurate supervision. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1017–1025.

# A Supplemental Material

## A.1 Dataset Preparation

We randomly split the full datasets into three parts – a training set, a validation set and a test set, with ratios of 80%, 10% and 10%, respectively. The splitting is fixed for all the methods for fair comparisons. We use the training set to train the model, the validation set to for optimal early stopping and hyperparameters fine-tuning, and finally evaluate different methods on the test set.

Recall our definition of the matched corpus $\mathbf{D}_L$. In practice, we only regard instances covered by more than $p$ sources as "matched" instances, where $p \in [0, 1, 2, \ldots k-1]$. Specifically, $p$ is set to $2, 1, 1, 0, 0$ for YouTube, Yelp, IMDB, AGNews, and Spouse datasets.

We obtain the pre-trained BERT embeddings from the 'bert-base-uncased' model. Our pre-processed data with the BERT embeddings and weak labels are available to download at https://drive.google.com/drive/u/1/folders/1MJe1BJYNPudfmpFxCeHwYqXMx53Kv4h_.
The dataset description can be found in our Github repo https://github.com/weakrules/Denoise-multi-weak-sources/blob/master/README.md.

## A.2 Model Training

**Computing infrastructure** Our code can be run on either CPU or GPU environment with Python 3.6 and Pytorch.

**Running time** Our model consists of two simple MLP networks with **0.199M** trainable parameters, thus the model is very time efficient with the avearge running time **4.5 minutes**. The running time differ based on the dataset size. We test our code on the System Ubuntu 18.04.4 LTS with *CPU: Intel(R) Xeon(R) Silver 4214 CPU @ 2.20GHz* and *GPU: NVIDIA GeForce RTX 2080*. All the models are trained for a maximum of 500 epochs.

| Dataset | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| Running time (min) | 1.9 | 3.65 | 3.92 | 11.92 | 1.5 |

Table 6: Running time for one experiment on CPU for five datasets in minutes

**Validation performance** For the main results in Table 2, the corresponding validation accuracy for our model is shown in Table 7.

| Dataset | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| Validation accuracy | 87.8 | 81.8 | 88.2 | 85.6 | 79.7 |
| Test accuracy | 94.9 | 82.9 | 87.5 | 85.7 | 81.3 |

Table 7: validation accuracy on for five datasets of the main results in Table 2.

## A.3 Hyperparameter Search

Since our datasets are well balanced, we use *accuracy* as the criterion for optimal early stopping and hyperparameters fine-tuning. Our hyperparameter values are uniform sampled within a reasonable range with particular numbers in Table 8.

| Parameters | Search Range |
|---|---|
| $d_{\text{h}}$ | 32, 64, 128, 256, 512 |
| $lr$ | 0.001, 0.002, 0.005, 0.01, 0.02, 0.05 |
| $c_1$ | 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 |
| $c_3$ | 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9 |

Table 8: The hyper parameters search bounds.

Table 9 shows the hyper parameters used to get the best results for Table 2.

| Parameters | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| $d_{\text{h}}$ | 128 | 64 | 128 | 256 | 256 |
| $lr$ | 0.02 | 0.02 | 0.02 | 0.05 | 0.02 |
| $c_1$ | 0.2 | 0.2 | 0.2 | 0.1 | 0.2 |
| $c_3$ | 0.1 | 0.2 | 0.2 | 0.1 | 0.1 |

Table 9: The hyper parameters setting for the best accuracy results of Table 2.

For the above four parameters with their range, we perform 1350 search trails. The test and validation results accuracy with mean and standard deviation for hyperparameters search experiments are in Table 10.

## A.4 Labeling Sources

We have four types of annotation rules which are Keyword Labeling Sources, Pattern-matching (Regular Expressions) Labeling Sources, Heuristic Labeling Sources, and Third-party Tools. For the first and second one, we give the uniform definitions for all the datasets.

- *Keyword Labeling Sources*

  Given $x$ as a document $\boldsymbol{d_i}$ in a corpus of text documents $\mathbf{D}$, a keywords list $L$, and a class label $C$ in the set of target classes $\mathcal{C}$, we define keywords matching annotation process `HAS` as

| | youtube | imdb | yelp | agnews | spouse |
|---|---|---|---|---|---|
| Val Mean | 81.5 | 77.1 | 79.1 | 80.0 | 83.5 |
| Val Stdev | 0.019 | 0.036 | 0.034 | 0.073 | 0.093 |
| Test Mean | 87.1 | 78.0 | 81.2 | 79.8 | 79.5 |
| Test Stdev | 0.021 | 0.031 | 0.042 | 0.070 | 0.118 |

Table 10: The validation and test results for the hyper-parameters search trails with the mean and standard deviation.

**Definition 2** (Keywords rules)**.** `HAS(x, L)` $\Rightarrow$ `C` if $x$ matches one of the words in the list $L$.

- *Pattern-matching Labeling Sources*

  Given $x$, a regular expression $R$, and a class label $C$, we define the pattern-matching annotation process `MATCH` as

  **Definition 3** (Pattern-matching rules)**.** `MATCH(x, R)` $\Rightarrow$ `C` if $x$ matches the regular expression $R$.

For the remaining third and fourth types, each dataset has specific definitions. We then state all the labeling rules for each dataset from Table 12 to Table 16.

### A.4.1 Statistics of Labeling Sources

We show the accuracy and coverage of each rule in the Fig 7, where the shape represents the coverage and the color depth represents the accuracy of the rule-induced labeled data. The average accuracy of these rules is 67.5%, and the average coverage is 23.3%.

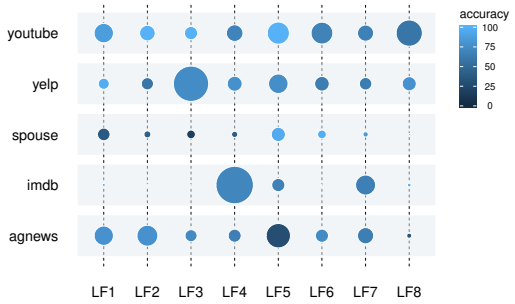


Figure 7: The coverage and accuracy of our used labeling functions on five datasets. Larger circle denotes higher coverage and lighter color denotes higher accuracy.

We also show one example of Yelp dataset with the detail statistics for each labeling source, and the rule descriptions are in Table 14.

| Labeling source | Coverage | Emp. Accu |
|---|---|---|
| textblob | 6.80 | 97.06 |
| keyword_recommand | 8.40 | 59.52 |
| keyword_general | 75.20 | 74.20 |
| keyword_mood | 12.80 | 78.12 |
| keyword_service | 33.30 | 75.68 |
| keyword_price | 23.30 | 63.93 |
| keyword_environment | 8.80 | 63.64 |
| keyword_food | 11.40 | 78.95 |

Table 11: The labeling rules statictics for Yelp dataset. Both Coverage and Emp. Accu (number of corrected samples / rule-matched samples) are in %.

### A.4.2 Rules Description

We show some examples of labeling rules here, and the full description of rules and their corresponding weak labels are in our Github repo https://github.com/weakrules/Denoise-multi-weak-sources/tree/master/rules-noisy-labels.

**Youtube** We use the same labeling functions as (Ratner et al., 2017), and we show the rules with an example in Table 12.

**IMDB** The rules are straightforward so we show the rules without the sentence examples in Table 13.

**Yelp** The rules are straightforward so we show the rules without the sentence examples in Table 14. We provide labeling rules in eight views.

**AGnews** The rules are straightforward so we show the rules without the sentence examples in Table 15.

**Spouse** We use the same rule as (Ratner et al., 2017) and we show the definition as well as examples in Table 16.

| Rule | Example |
|---|---|
| `HAS(x,[my]) ⇒ SPAM` | Plizz withing my channel |
| `HAS(x, [subscribe]) ⇒ SPAM` | Subscribe to me and I'll subscribe back!! |
| `HAS(x, [http]) ⇒ SPAM` | please like : http://www.bubblews.com/news/9277547-peace-and-brotherhood |
| `HAS(x, [please, plz]) ⇒ SPAM` | Please help me go here http://www.gofundme.com/littlebrother |
| `HAS(x, [song]) ⇒ HAM` | This song is great there are 2,127,315,950 views wow |
| `MATCH(x, check.*out ) ⇒ SPAM` | Please check out my vidios |
| We define `LENGTH(x)` as the number of words in $x$.<br>`LENGTH(x) < 5 ⇒ HAM` | 2 BILLION!! |
| We define $x.ents$ as the tokens of $x$, and $x.ent.label$ as its label.<br>$LENGTH(x) < 20 \ AND \ any([ent.label == PERSON \ for \ ent \ in \ x.ents] \Rightarrow HAM$ | Katy Perry is garbage. Rihanna is the best singer in the world. |
| We define `POLARITY(x)` as the sentiment subjectivity score obtained from the TextBlob tool, a pretrained sentiment analyzer.<br>$POLARITY(x) > 0.9 \Rightarrow HAM$ | Discover a beautiful song of A young Moroccan http://www.linkbucks.com/AcN2g |

Table 12: Youtube labeling sources examples

| Rule |
|---|
| `[masterpiece, outstanding, perfect, great, good, nice, best, excellent, worthy, awesome, enjoy, positive, pleasant, wonderful, amazing, superb, fantastic, marvellous, fabulous] ⇒ POS` |
| `[bad, worst, horrible, awful, terrible, crap, shit, garbage, rubbish, waste] ⇒ NEG` |
| `[beautiful, handsome, talented]⇒ POS` |
| `[fast forward, n t finish] ⇒ NEG` |
| `[well written, absorbing, attractive, innovative, instructive, interesting, touching, moving]⇒ POS` |
| `[to sleep, fell asleep, boring, dull, plain]⇒ NEG` |
| `[ than this, than the film, than the movie]⇒ NEG` |
| `MATCH(x, *PRE*EXP* ) ⇒ POS`<br>`PRE = [will, ll , would , d , can t wait to ]`<br>`EXP = [next time, again, rewatch, anymore, rewind]` |
| `MATCH(x, *PRE*EXP* ) ⇒ POS`<br>`PRE = [highly, do, would, definitely, certainly, strongly, i, we]`<br>`EXP = [recommend, nominate]` |
| `MATCH(x, *PRE*EXP* ) ⇒ POS`<br>`PRE = [high, timeless, priceless, has, great, real, instructive]`<br>`EXP = [value, quality, meaning, significance]` |

Table 13: IMDB labeling sources examples

| View | Rule |
|---|---|
| General | [outstanding, perfect, great, good, nice, best, excellent, worthy, awesome, enjoy, positive, pleasant,wonderful, amazing] ⇒ POS |
| General | [bad, worst, horrible, awful, terrible, nasty, shit, distasteful,dreadful, negative]⇒ NEG |
| Mood | [happy, pleased, delighted,contented, glad, thankful, satisfied] ⇒ POS |
| Mood | [sad, annoy, disappointed,frustrated, upset, irritated, harassed, angry, pissed]⇒ NEG |
| Service | [friendly, patient, considerate, enthusiastic, attentive, thoughtful, kind, caring, helpful, polite, efficient, prompt] ⇒ POS |
| Service | [slow, offended, rude, indifferent, arrogant]⇒ NEG |
| Price | [cheap, reasonable, inexpensive, economical] ⇒ POS |
| Price | [overpriced, expensive, costly, high-priced]⇒ NEG |
| Environment | [clean, neat, quiet, comfortable, convenien, tidy, orderly, cosy, homely] ⇒ POS |
| Environment | [noisy, mess, chaos, dirty, foul]⇒ NEG |
| Food | [tasty, yummy, delicious,appetizing, good-tasting, delectable, savoury, luscious, palatable] ⇒ POS |
| Food | [disgusting, gross, insipid]⇒ NEG |
| | [recommend] ⇒ POS |
| Third-party Tools | $POLARITY(x) > 0.5 \Rightarrow$ POS<br><br>$POLARITY(x) > 0.5 \Rightarrow$ NEG |

Table 14: Yelp labeling sources examples

| Rule |
|---|
| [ war , prime minister, president, commander, minister, annan, military, militant, kill, operator] ⇒ POLITICS |
| [baseball, basketball, soccer, football, boxing, swimming, world cup, nba,olympics,final, fifa] ⇒ SPORTS |
| [delta, cola, toyota, costco, gucci, citibank, airlines] ⇒ BUSINESS |
| [technology, engineering, science, research, cpu, windows, unix, system, computing, compute] ⇒ TECHNOLOGY |

Table 15: AGnews labeling sources examples

| Rule | Example |
|---|---|
| `[father, mother, sister, brother, son, daughter, grandfather, grandmother, uncle, aunt, cousin] ⇒ NEG` | His 'exaggerated' sob stories allegedly include claiming he had cancer, and that his son had made a suicide attempt. |
| `[boyfriend, girlfriend, boss, employee, secretary, co-worker] ⇒ NEG` | Dawn Airey's departure as European boss of Yahoo after just two years will bring a smile to the face of Armando Iannucci. |
| `MATCH(x, *PERSON1*LIST*PERSON2* ⇒ POS`<br>`LIST = [spouse, wife, husband, ex-wife, ex-husband]` | On their wedding day, last week sundayGhanaian actress Rose Mensah, popularly known as Kyeiwaa, has divorced her husband Daniel Osei, less than four days after the glamorous event. |
| We define `LASTNAME(x)` as the last name of x. `LASTNAME(person1) == LASTNAME(person2) ⇒ POS` | Karen Bruk and Steven Bruk, Mrs. Bruk's spouse, exercise shared investment power over the Shares of the Company held by Karen Bruk and KMB. |

Table 16: Spouse labeling sources examples