Reproducibility report for CSE 517

Yanmeng Kong, Tao Lin, Lijia Ma
University of Washington
Seattle, WA 98105
{yk57, lijiam, soxvlin}@uw.edu

1 Introduction

In this project, we plan to reproduce the paper **"Denoising Multi-Source Weak Supervision for Neural Text Classification"** from EMNLP 2020[1]. This paper aims at doing text classification without using any human-labeled data. To solve this problem, a possible way is to induce weak labels with rules.

Weakly-supervised learning methods label 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.

The authors use a conditional soft attention mechanism to estimate the reliability of labeling sources so as to reduce the label noise. Then the denoised pseudo labels will be used to train a text classifier.

For this project, we plan to reproduce the experiments on five datasets: **youtube**, **imdb**, **yelp**, **agnews**, **spouse** in this paper. With the given code in Github, we will try to get similar results as the authors.

2 Scope of reproducibility

The paper introduces a conditional soft attention mechanism to reduce the noises from rule-induced weak labels. The denoised pseudo labels then supervise a neural classifier to predicts soft labels for unmatched samples. We want to replicate and examine the following claims that central to the paper's argument:

- Claim 1: the proposed label-denoising model achieve higher accuracy than 4 state-of-the-art weakly-supervised and 3 semi-supervised methods on five benchmarks for sentiment, topic, and relation classifications. The 4 baseline weakly-supervised methods are:
  - **Snorkel**: a generative model denoising multiple weak labels.
  - **WeSTClass**: use self-training to classify texts.
  - **ImplyLoss**: use implication loss to denoise rule-based labels.
  - **NeuralQPP**: select useful labels from multiple sources and then classify texts.

  The 3 baseline semi-supervised methods are:
  - **MT**: use Mean-Teacher method to average model weights and add a consistency regularization on the student and teacher model.
  - **ULMFiT**: a strong deep text classifier based on pre-training and fine-tuning.
  - **BERT-MLP**: use pre-trained Transformer as the feature extractor and stacks a multi-layer perceptron on its feature encoder.

  In this reproducibility report, we examine whether the proposed method outperforms **BERT-MLP**, which is the best baseline model in their paper.
- Claim 2: Compared to majority-voted labels, the proposed label denoising method generates labels that can significantly improve the accuracy of supervised models, such as **ULMFiT** and **BERT-MLP**. In this reproducibility report, we examine whether denoised labels can improve the performance of **BERT-MLP**.

3 Methodology

Formally, in the problem of weak supervised learning, we have the following elements:

- a corpus $\boldsymbol{D} = \{\boldsymbol{d}_1, \ldots, \boldsymbol{d}_n\}$ of text documents;
- a set of target classes $\mathcal{C} = \{C_1, \ldots, C_m\}$;
- a set of weak labeling rules $\mathcal{S} = \{\mathcal{R}_1, \ldots, \mathcal{R}_k\}$.

The goal of weak supervised learning is to learn a classifier from $\boldsymbol{D}$ with only multiple weak supervision sources to accurately classify any newly arriving documents. Only a small portion of documents will be covered by certain weak labels, denoted as $\boldsymbol{D}_L$, and those uncovered by weak labeling rules are denoted as $\boldsymbol{D}_U$.

3.1 Model descriptions

The model used in the original paper 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.

3.1.1 Label Denoiser with a Conditional Soft Attention Mechanism

Source Reliability To reduce the noise of weak labels, the proposed label denoiser estimates the reliability of each labeling sources using a conditional soft attention mechanism, and then aggregates weak labels through weighted voting of the labeling sources to achieve "pseudo-clean" labels. The reliability scores are conditioned on both rules and document feature representations.

The soft attention mechanism is conditioned on both weak labels, denoted as $\tilde{y}$, and feature representation (e.g., features learned from BERT), denoted as $\boldsymbol{B}$, to estimate the source reliability. The core of this attention net is a two-layer feed-forward neural network which predicts the attention score for samples that are matched by labeling sources. For each document $\boldsymbol{d}_i$, its attention score $q_{i,j}$ of one labeling source $\mathcal{R}_j$ is:

$$\hat{q}_{ij} = W_2^T \tanh\left(W_1\left(\tilde{y}_{ij} + \boldsymbol{B}_i\right)\right)$$
$$q_{ij} = \frac{\exp\left(\hat{q}_{ij}\right)}{\sum_j \exp\left(\hat{q}_{ij}\right)}$$

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 is calculated over matched annotators as

$$a_{ij} = q_{ij} \cdot \mathbb{I}_C(\tilde{y}_{ij} \geq 0)$$

where $\mathbb{I}_C(\cdot)$ is an indicator function, and a_{ij} is the attention score of labeling source $\mathcal{R}_j$ for document $\boldsymbol{d}_i$ that is subject to the constraint $\sum_{j=1}^{k} a_j = 1$.

Denoising Pseudo labels After getting the source reliability score $\boldsymbol{A}$, we reweight the sources to get **weighted majority voted labels** $\hat{Y}$ by $\tilde{Y}_i \otimes \boldsymbol{A}$. Namely, for each document $\boldsymbol{d}_i$, we have

$$\hat{y}_i = \arg\max_{C_r} \sum_{j=1}^{k} a_j \mathbb{I}_C(\tilde{y}_{ij} = C_r)$$

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

Soft Predicted Probabilities from Rule-based Classifier At the epoch t, we learn the reliability score $\boldsymbol{A}^{(t)}$ and soft predictions $\hat{\boldsymbol{Z}}^{(t)}$ supervised by "pseudo-clean" labels from the previous epoch $\hat{\boldsymbol{Y}}^{(t-1)}$. Then we renew "clean-pseudo" labels as $\hat{\boldsymbol{Y}}^{(t)}$ using the score $\boldsymbol{A}^{(t)}$.

Given m target classes and k weak annotators, the prediction probability $\hat{\boldsymbol{z}}$ for $\boldsymbol{d}$ is obtained by weighting the noisy labels and corresponding reliability scores, that is, $\hat{\boldsymbol{z}}_i^{(t)} = \text{softmax}(\tilde{\boldsymbol{Y}}_i^{(t-1)} \otimes \boldsymbol{A}_i^{(t)})$.

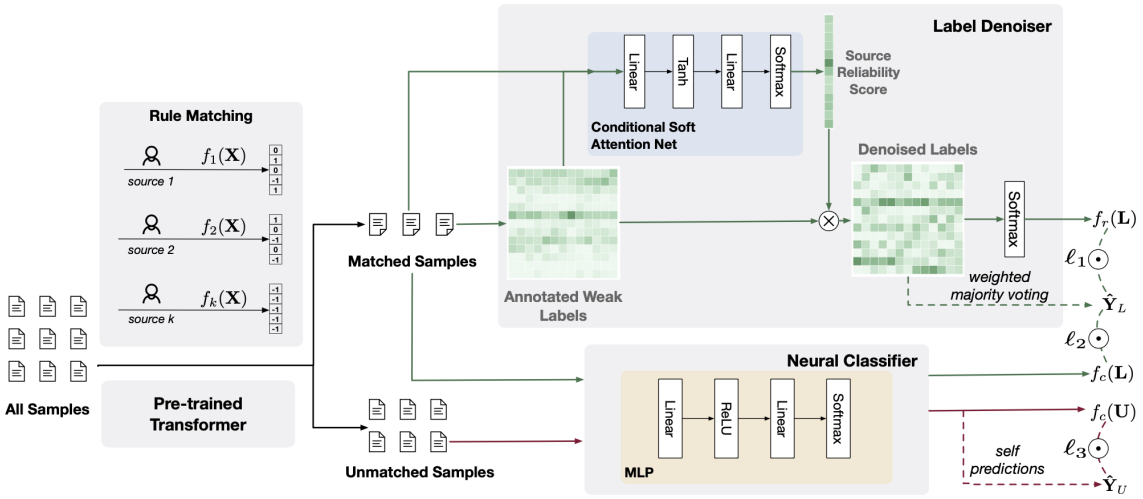

Figure 1: Model Architecture

3.1.2 Neural Classifier

The neural classifier is designed to handle all the samples, including matched ones and unmatched ones. The authors use pre-trained BERT to extract features, and then feed the text embeddings $\boldsymbol{B}$ into a feed-forward neural network to obtain the final predictions. Hence, For $\boldsymbol{d}_i \in \boldsymbol{D}_L \cup \boldsymbol{D}_U$, the soft prediction $\tilde{\boldsymbol{z}}_i$ from neural classifier is:

$$\tilde{\boldsymbol{z}}_i = f_\theta(\boldsymbol{B}_i; \theta_w)$$

where $f_\theta(\cdot)$ denotes the two-layer feed-forward neural network, and θ_w denotes its parameters.

3.2 Training Objective

The overall objective is to minimize the following ensemble loss function:

$$\ell = c_1\ell_1 + c_2\ell_2 + c_3\ell_3$$

where $c_1, c_2, c_3 \in [0, 1]$ are hyper-parameters for balancing the three losses and satisfy $c_1 + c_2 + c_3 = 1$. ℓ_1, ℓ_2, ℓ_3 are loss function for each component of the proposed model:

Rule Denoiser Loss ℓ_1 is the loss of the rule-based classifier over D_L. The proposed method uses the "pseudo- clean" labels $\hat{Y}$ to self-train the label denoiser iteratively.

$$\ell_1 = -\sum_{i \in \boldsymbol{D}_L} \hat{y}_i \log \hat{\boldsymbol{z}}_i$$

Neural Classifier Loss ℓ_2 is the loss of the neural classifier over matched sample D_L. The proposed method compare soft predictions from neural classifer to the pseudo-clean labels, and then derive the training loss:

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

Unsupervised Self-training Loss ℓ_3 is the loss of the neural classifier over unmatched sample D_U. To further enhance the label quality of $\boldsymbol{D}_U$, the authors aggregate the predictions of multiple previous network evaluations into an ensemble prediction to alleviate noise propagation. That is, a document $\boldsymbol{d}_i \in \boldsymbol{D}_U$, the neural classifier prediction $\tilde{\boldsymbol{z}}_i$ are accumulated into ensemble output $\boldsymbol{Z}_i$ by updating $\boldsymbol{Z}_i^{(t)} \leftarrow \alpha \boldsymbol{Z}_i^{(t-1)} + (1-\alpha)\tilde{\boldsymbol{z}}_i^{(t)}$, and then adjusted by bias correction, namely $\boldsymbol{p}_i = \boldsymbol{Z}_i/(1-\alpha^{(t)})$, where t is the current epoch. Then, we minimize the Euclidean distance between $\boldsymbol{p}_i$ and $\tilde{\boldsymbol{z}}_i$

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

3.3 Hyperparameters

We follow the original paper and tune hyperparameters based on the following searching space:

Table 1: Searching Space for Hyperparameters

Parameters	Search Range
Dimension of Hidden Layers	32, 64, 128, 256, 512
Learning Rate	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

3.4 Implementation

Well will use the existing code and add our additional code for experiments: `https://github.com/AnnyKong/Denoise-multi-weak-sources-rep`. Python 3 will be used. And here is the repo for the existing code: `https://github.com/weakrules/Denoise-multi-weak-sources`.

3.5 Experimental setup

3.5.1 Datasets and tasks

To reproduce the experiment, we will use 5 benchmark datasets for text classification: **youtube** [2] (Spam Detection), **imdb** [3], **yelp** [4] (Sentiment Analysis), agnews [4] (Topic Classification), and spouse [5] (Relation Classification). Table 1 shows the statistics of these datasets and the coverage and accuracy of weak labels.

Table 2: Data Statistics

Dataset	**Task**	**#Classes**	**# Train**	**#Dev**	**#Test**	**Coverage**	**Accuracy**
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

Besides, we have also created a new dataset by ourselves. This dataset **fbnews** contains 450 posts (50 labeled, 350 unlabeled, and 50 test)from news organizations on Facebook. Table 2 shows a subset of this new dataset:

Table 3: fbnews examples

0	Because Louis Barnes didn't fire the shot that killed a nurse, he may no longer face a manslaughter charge, a judge decided.
1	President Donald Trump has signed an order that will temporarily bar entry to the U.S. of foreign nationals, other than immediate family of U.S. citizens and permanent residents, who have traveled in China within the last 14 days.
2	Memorial days, like the Holocaust Remembrance Day and Martin Luther King Jr. Day, help us remember the horrors of the past so we don't relive them.
3	The three-day strike at every location of health-care provider Swedish is over, but some nurses and other workers are still rallying outside the hospitals.
4	L.F. Trottier & Sons, the longtime dealer of John Deere farm and garden equipment in the core of the Upper Valley, told employees on Friday that it is in discussions to sell the family-owned business to a large Texas-based distributor of John Deere equipment.
5	The new record total exceeded the previous mark (set in 2018) by 7.4 percent.

3.5.2 Baseline Model

In this reproducibility report, we compare the proposed method with the best baseline model in the original paper, **BERT-MLP**. **BERT-MLP** takes the pre-trained Transformer as the feature extractor and stacks a multi-layer perceptron on its feature encoder.

3.6 Computational requirements

Before we actually did any experiment, since the datasets used in this paper are not so large, we planned to use around 4 GPUs to finish all the experiments.

However, after we read the paper more carefully, we found that this model is well-designed and does not require GPU usage. Therefore, we ran all the experiments on our own laptops. For most datasets, the training process will take less than 20 minutes.

4 Results

4.1 Result 1: Does the proposed model outperforms baseline model?

In Experiment 1, we aim to examine the first hypothesis, that is, the proposed label-denoising model achieve higher accuracy than **BERT-MLP** on five benchmarks for sentiment, topic, and relation classifications.

Table 4: The Best Performance of the proposed model and **BERT-MLP**

Model	Performance	youtube	imdb	yelp	agnews	spouse
proposed model	Validation Accuracy	81.6	78.7	77.3	77.3	76.4
	Test Accuracy	83.2	80.0	83.3	72.1	74.8
BERT-MLP	Accuracy	81.6	80.0	73.8	77.3	66.6

We find that the performance of the proposed model mostly outperforms the **BERT-MLP** as shown in the table 4. And as we validate the results, with the same hyper parameters used in the original paper, we find the best validation accuracy and test accuracy to be very different from the results in the original paper.

Additionally, we performed various experiments with different hyper-parameters and find that some parameters may affect the results randomly but are not reflected in the original paper. For example, with a different seed/k/x0 being set (i.e. seed was default to 0 but can be set to 1) may improve both the validation and test accuracy. More experiment results can be referred in our GitHub Repository.

4.2 Result 2: Does denoised labels improve the performance of baseline model?

In Experiment 2, we aim to examine the second hypothesis, that is, the denoised labels can improve the performance of state-of-art semi-supervised model. Particularly, we apply denoised labels to **BERT-MLP**, and compare its performance with the one trained on majority-voted labels. The denoised labels are obtained from the proposed model with the best parameter combination in the orginal paper. For each benchmark dataset and each label type, we run **BERT-MLP** 50 epochs. The results are shown in Table 5.

Table 5: The Performance of **BERT-MLP** with Majority-voted and Denoised Labels

Label Type	youtube	imdb	yelp	agnews	spouse
Majority voted labels	88.2	80.6	75.8	85.7	64.5
Denoised labels	85.2	72.9	72.3	79.6	47.9

We find that using denoised labels from the proposed model rather decrease the performance of **BERT-MLP**. At the current stage, we think it might be attributed to the calculation of denoised pseudo labels:

$$\hat{y}_i = \arg\max_{C_r} \sum_{j=1}^{k} a_j \mathbb{I}_C(\tilde{y}_{ij} = C_r)$$

In dataset except for **youtube**, there are a lot of texts that are not covered by any weak rules, so their majority voted labels are assigned with an "abstain label" (-1). However, in their source code, when they calculate $\hat{Y}$, there will be no predefined categories matched these abstain labels, so they will be converted to reference labels (0) by default. This automatic transformation for texts without any weak labels might be incorrect and harm the performance of text classification.

4.3 Additional results not present in the original paper

With the new **fbnews** dataset we built, we tested the performance of this model. The test accuracy of **fbnews** is **0.60**. Since this new dataset is rather small - with only 50 items in the test set, we hope the performance could be better when we have a larger dataset.

5 Discussion

In this reproducibility project, we examine whether the label denoising approach proposed by [1] outperforms the state-of-the-arts semi-supervised model, **BERT-MLP**, and whether the denoised labels derived from their proposed model can effectively improve the performance of **BERT-MLP**. We cannot reproduce their experiment results. First, we find that the proposed model does not outperforms **BERT-MLP** in all 5 benchmark tasks. We also find that it has a large variation in test set accuracy when we choose different random seeds. Second, instead of improving **BERT-MLP**, the denoised labels decrease the performance of **BERT-MLP**. In the current stage, we think this is because their method cannot deal with those texts that are not covered by any weak rules. The denoised labels derived from their proposed model will convert these "abstain labels" (-1) to reference labels (0), which might be an incorrect classification.

5.1 What was easy

Since there are well-documented code, we find it not very hard to run the experiments in the paper. Also, the authors provided search ranges of hyper-parameters and the values they used in their appendix, we find it very helpful to start with that information. That saved us a lot of time.

5.2 What was difficult

At the very beginning, we thought the only hyper-parameters need to be considered are the $c1, c2$ etc. But then we realized the random states have a significant influence on the results. Therefore, it would be very hard(not correct as well) to get exactly the same results as the authors.

Besides, when we tried to build a new dataset, we find it far from trivial. Since the authors used an unusual format(.pt) to save their results, we have to follow the same preprocessing method to build our own dataset. We think if the input of the end-to-end framework can be .csv files, it would be more convenient for other people to use.

5.3 Recommendations for reproducibility

Basically, we have the following recommendations for people who want to publish their code and want to improve reproducibility:

1. Clear comments can be really helpful for others.
2. Avoid using ambiguous names for variables.
3. Use common format to store your dataset. People can save a lot of time if they want test your model on some other datasets.
4. If you want to publish your code as a pip package, make sure you include a simple tutorial for users.

References

[1] Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. Denoising multi-source weak supervision for neural text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3739–3754, Online, November 2020. Association for Computational Linguistics.

[2] Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. Tubespam: Comment spam filtering on youtube. In *2015 IEEE 14th international conference on machine learning and applications (ICMLA)*, pages 138–143. IEEE, 2015.

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

[4] Xiang Zhang, Junbo Zhao, and Yann Lecun. Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, 2015:649–657, 2015.

[5] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.