SAT: Improving Semi-Supervised Text Classification with Simple Instance-Adaptive Self-Training

Hui Chen  Wei Han  Soujanya Poria
Singapore University of Technology and Design
{hui_chen, wei_han}@mymail.sutd.edu.sg
sporia@sutd.edu.sg

Abstract

Self-training methods have been explored in recent years and have exhibited great performance in improving semi-supervised learning. This work presents a Simple instance-Adaptive self-Training method (SAT) for semi-supervised text classification. SAT first generates two augmented views for each unlabeled data and then trains a meta-learner to automatically identify the relative strength of augmentations based on the similarity between the original view and the augmented views. The weakly-augmented view is fed to the model to produce a pseudo-label and the strongly-augmented view is used to train the model to predict the same pseudo-label. We conducted extensive experiments and analyses on three text classification datasets and found that with varying sizes of labeled training data, SAT consistently shows competitive performance compared to existing semi-supervised learning methods. Our code can be found at https://github.com/declare-lab/SAT.git.

1 Introduction

Pretrained language models have achieved extremely good performance in a wide range of natural language understanding tasks (Devlin et al., 2019). However, such performance often has a strong dependence on large-scale high-quality supervision. Since labeled linguistic data needs large amounts of time, money, and expertise to obtain, improving models’ performance in few-shot scenarios (i.e., there are only a few training examples per class) has become a challenging research topic.

Semi-supervised learning in NLP has received increasing attention in improving performance in few-shot scenarios, where both labeled data and unlabeled data are utilized (Berthelot et al., 2019b; Sohn et al., 2020; Li et al., 2021). Recently, several self-training methods have been explored to obtain task-specific information in unlabeled data. UDA (Xie et al., 2020) applied data augmentations to unlabeled data and proposed an unsupervised consistency loss that minimizes the divergence between different unlabeled augmented views. To give self-training more supervision, MixText (Chen et al., 2020a; Berthelot et al., 2019b) employed Mixup (Zhang et al., 2018; Chen et al., 2022) to learn an intermediate representation of labeled and unlabeled data. Both UDA and MixText utilized consistency regularization and confirmed that such regularization exhibits outstanding performance in semi-supervised learning. To simplify the consistency regularization process, FixMatch (Sohn et al., 2020) classified two unlabeled augmented views into a weak view and a strong view, and then minimized the divergence between the probability distribution of the strong view and the pseudo label of the weak view. However, in NLP, it is hard to distinguish the relative strength of augmented text by observation, and randomly assigning an augmentation strength will limit the performance of FixMatch on text.

To tackle this problem in FixMatch, our paper introduces an instance-adaptive self-training method SAT, where we propose two criteria based on a classifier and a scorer to automatically identify the relative strength of augmentations on text. Our main contributions are:

• First, we apply popular data augmentation techniques to generate different views of unlabeled data and design two novel criteria to calculate the similarity between the original view and the augmented view of unlabeled data in FixMatch, boosting its performance on text.

• We then conduct empirical experiments and analyses on three few-shot text classification datasets. Experimental results confirm the efficacy of our SAT method.
2 Method

2.1 Problem Setting

In this work, we learn a model to map an input $x \in \mathcal{X}$ onto a label $y \in \mathcal{Y}$ in text classification tasks. In semi-supervised learning, we use both labeled examples and unlabeled examples during training. Let $\mathcal{X} = \{(x_b, y_b) : b \in (1, ..., B)\}$ be a batch of $B$ labeled examples, where $x_b$ are the training examples and $y_b$ are labels. Let $\mathcal{U} = \{u_b : b \in (1, ..., \mu B)\}$ be a batch of $\mu B$ unlabeled examples, where $\mu$ is a hyperparameter which determines the relative sizes of $\mathcal{X}$ and $\mathcal{U}$.

2.2 SAT

The entire process of SAT is illustrated in Algorithm 1. Similar to common semi-supervised learning methods, our approach consists of a supervised part and an unsupervised part. Our supervised part minimizes the cross-entropy loss between the labeled data and their targets. Our unsupervised part first generates two unlabeled augmented views, then applies an augmentation choice network to determine the relative augmentation strength, and finally calculates a consistency loss between the original data and their targets. Our unsupervised part is used to automatically identify the augmentation strength in unlabeled data.

Algorithm 1: SAT: Simple Instance-Adaptive Self-Training

| Input: $\mathcal{D}^{\text{train}} = \{\mathcal{X}, \mathcal{U}\}$ where $\mathcal{X} = \{(x_b, y_b) : b \in (1, ..., B)\}$ and $\mathcal{U} = \{u_b : b \in (1, ..., \mu B)\}$; augmentation methods $\alpha_1, \alpha_2$; main network $f(; \theta)$ with parameters $\theta$ and its probability distribution $p$; augmentation choice network $G(; \theta_G)$ with parameters $\theta_G$; criteria $C, \Gamma$; cross-entropy loss $H$; unlabeled loss weight $\lambda_u$; confidence threshold $\tau$; learning rates $\beta, \eta$
| Output: Updated network weights $\theta$
| 1. $l_s = \frac{1}{B} \sum_{b=1}^{B} H(y_b, p(y|x_b))$
| 2. for $(x_b, y_b) \in \mathcal{X}$ do
| 3. $i_1^b, i_2^b = (C(\alpha_1(x_b), x_b, y_b), C(\alpha_2(x_b), x_b, y_b))$
| 4. $i_{w_1}^b, i_{w_2}^b = \text{Descending}(i_1^b, i_2^b)$
| 5. end
| 6. $l_{\text{aug,choice}} = \frac{1}{B} \sum_{b=1}^{B} \Gamma(x_b, \alpha_1(x_b), \alpha_2(x_b), i_{w_1}^b)$
| 7. $\theta_G = \theta_G - \beta \nabla l_{\text{aug,choice}}$
| 8. for each $u_b \in \mathcal{U}$ do
| 9. $i_{w_1}^b, i_{w_2}^b = G(u_b, \alpha_1(u_b), \alpha_2(u_b); \theta_G)$
| 10. end
| 11. $l_u = \frac{1}{\mu B} \sum_{b=1}^{\mu B} \{\max(p(y|\alpha_{w_1}^b(u_b))), p(y|\alpha_{w_2}^b(u_b))\} > \tau\} H(\text{argmax}(p(y|\alpha_{w_1}^b(u_b))), p(y|\alpha_{w_2}^b(u_b)))$
| 12. $l_{\text{total}} = l_s + \lambda_u l_u$
| 13. $\theta = \theta - \eta \nabla l_{\text{total}}$

3 Experimental Setup

We conducted empirical experiments to compare our approach with a couple of existing semi-supervised learning methods on a variety of text classification benchmark datasets.
We considered three diverse few-shot text classification scenarios in our experiments: AG News which categorizes more than 1 million news articles into 4 categories — World, Sports, Business, and Sci/Tech (Zhang et al., 2015), and Yahoo! which classifies question-answer pairs into 10 clusters where all question-answer pairs in a cluster ask about the same thing (Zhang et al., 2015), and IMDB which predicts sentiment of movie reviews to be positive or negative (Maas et al., 2011).

We used the original test set as our test set and randomly sampled from the training set to construct the training unlabeled set and development set. To balance the class distribution in the experiments, we randomly sampled $N_c$ samples per class to be used for training. In AG News and IMDB, $N_c = 10$. In the Yahoo! dataset, we set $N_c = 20$ to ensure consistent results. For all experiments, we used accuracy (%) and macro F1 score (%) as the evaluation metrics.

### 3.2 Baselines

To test the effectiveness of our approach, we compared it with several popular self-training methods: UDA (Xie et al., 2020), MixText (Chen et al., 2020a), FixMatch (Sohn et al., 2020). To ensure fair comparisons, we used the same augmentation techniques, i.e., back-translation (Sennrich et al., 2016b) and synonym replacement (Wei and Zou, 2019), in all baselines. For back-translation augmentation, we used German as the middle language. For synonym replacement augmentation, the substitution percentage is 30%. Also, we used the same BERT-based-uncased model, unlabeled data size, and batch size in all methods.

### 4 Results

#### 4.1 Main Results

This section compares our SAT method with BERT, UDA, MixText, and FixMatch on three text classification datasets. Our main results are shown in Table 1. Results are averaged over five different runs.

We observed that BERT achieves a mean score across our three datasets of 63.17%, and UDA improves performance noticeably by +3.57%. MixText and FixMatch show better performance in improving the BERT baseline, where the mean score increases are +4.41% and +5.01%, respectively. Our classifier-based SAT method, which applies a cross-entropy loss to measure the similarity between the original data and its augmented variants, achieves a mean score of 70.83%, outperforming FixMatch by +2.65%. The scorer-based SAT, which employs cosine distance to measure the similarity further improves about 1% over the classifier-based SAT.

From these results, we obtained the following findings. Firstly, strategically selecting the strongly and weakly augmented samples in self-training can effectively boost performance. Compared to FixMatch, we improved the performance by adding a lightweight meta-learner to automatically identify augmentation strengths, without sacrificing much training time. Secondly, when tackling datasets with few training examples, using cosine distance to measure the similarity between two examples and using contrastive loss to train the meta-learner shows better and more robust performance.

#### 4.2 Ablation: Size of Labeled Data

This ablation investigates how our semi-supervised learning method performs for different sizes of labeled data. Fig. 1 compares the average scores of accuracy and macro F1 of FixMatch and our meth-

---

Table 1: Accuracy (%) and Macro F1 (%) on three diverse text classification tasks for BERT, UDA, MixText, FixMatch, and our SAT method. $c$: number of classes; $\Delta$: improvement compared with BERT.
ods. First, we observed that our methods can consistently outperform FixMatch with varying data sizes. This indicates that strategically selecting the strongly and weakly augmented samples contributes to the final performance in self-training. Second, when $N_c$ increases from 3 to 10, the scores of the three methods increase accordingly. When $N_c$ becomes 20, the performance of FixMatch and the classifier-based SAT drops, which is consistent with prior findings on the diminished effect of data augmentation for larger datasets (Xie et al., 2020; Andreas, 2020). However, the scorer-based SAT does not show an obvious performance decrease, showing that in few-shot datasets, the scorer-based method is more robust than the classifier-based method.

4.3 Ablation: Various Augmentation Techniques

To evaluate the effect of various augmentation techniques in our method, we performed experiments using different text augmentation technique combinations in our method. These augmentation techniques are widely-used: (1) Synonym Replacement (SR) substitutes words with WordNet synonyms (Wei and Zou, 2019); (2) Pervasive Dropout (PD) applies a word-level dropout with a probability of 0.1 on text (Sennrich et al., 2016a); (3) Random Insertion (RI) randomly inserts words in a sentence (Wei and Zou, 2019); (4) Back-translation (BT) translates text into another language and then back into the original language (Sennrich et al., 2016b).

Fig. 2 compares average scores of accuracy and macro F1 from different augmentation technique combinations in our method on the AG News dataset. The combination of back-translation and synonym replacement improves performance the best, perhaps because they maintain a good balance between injecting proper perturbation noise and preserving the original meaning of the text.

Figure 1: Average scores of accuracy and macro F1 from FixMatch and our method for different sizes of labeled data on the AG News dataset.

5 Related Work

Our work combines data augmentation and consistency regularization to improve semi-supervised learning and is inspired by prior work in these areas. Recently, several papers have proposed reinforcement learning policies (Cubuk et al., 2019; Ho et al., 2019) and curriculum learning strategies (Wei et al., 2021; Zhang et al., 2021) to automatically augment data. Also, a couple of consistency regularization methods are introduced to simplify the semi-supervised learning process (Berthelot et al., 2019a; Sohn et al., 2020) as well as to boost performance in domain adaptation scenarios (Berthelot et al., 2021) to improve semi-supervised learning. As far as we know, our work is the first to apply a meta-learner to automatically determine the augmentation strength in consistency regularization in semi-supervised text classification.

6 Conclusion

In closing, this paper has proposed an instance-adaptive self-training method SAT to boost performance in semi-supervised text classification. Inspired by FixMatch, SAT combines data augmentation and consistency regularization and designs a novel meta-learner to automatically determine the relative strength of augmentations. Empirical
experiments and ablation studies confirm SAT is simple yet effective in improving semi-supervised learning.

Limitations

Our proposed method has two limitations. First, in our experiments, we found the semi-supervised process is easy to be influenced by unlabeled data size. During training, we adjusted the size of unlabeled data in each batch by adjusting the $\mu$ value, as mentioned in Section 2.1. It will be a future direction that we design some strategies to automatically learn the $\mu$ value. Second, this work only considered the situation where there are only two augmentations in consistency regularization. As our SAT method can automatically rank the augmentation strengths, our future work is to extend SAT to regularize more than two augmentations.

Ethical Considerations

Augmentation techniques discussed in Section 4.3 should be used with care since they might generate data that do not align with the original meaning.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. This research is supported by the Ministry of Education, Singapore, under its AcRF Tier-2 grant (Project no. T2MOE2008, and Grantor reference no. MOET2EP20220-0017). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not reflect the views of the Ministry of Education, Singapore.

References


Andrew 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, pages 142–150.


### A Dataset Statistics

The dataset statistics and split information are presented in Table 2.

### B Implementation Details

We performed a grid search for hyperparameters: $\eta_{main} \in \{5e^{-5}, 1e^{-3}\}$, $\eta_{bert} \in \{1e^{-5}, 5e^{-5}\}$, $\beta$ is fixed at $1e^{-4}$, $\mu \in \{3, 4, 5, 8, 10, 20\}$, $\tau \in \{0.90, 0.95, 0.99\}$, and batch size is fixed at 32. We tuned our model on a single NVIDIA RTX 8000 GPU. We ran each experiment for 50 epochs with a patience of 15 or 10 for early stopping.

<table>
<thead>
<tr>
<th>Datasets</th>
<th># Unlabeled</th>
<th># Dev</th>
<th># Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>AG News</td>
<td>5000</td>
<td>2000</td>
<td>1900</td>
</tr>
<tr>
<td>Yahoo!</td>
<td>5000</td>
<td>2000</td>
<td>6000</td>
</tr>
<tr>
<td>IMDB</td>
<td>5000</td>
<td>1000</td>
<td>12500</td>
</tr>
</tbody>
</table>

Table 2: Dataset statistics and data splits. The number of unlabeled data, dev data and test data in the table means the number of data per class.