Progressive Self-Training with Discriminator for Aspect Term Extraction

Qianlong Wang[1], Zhiyuan Wen[1], Qin Zhao[1], Min Yang[2], Ruifeng Xu[1,3]*
[1]Harbin Institute of Technology (Shenzhen), China
[2]Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences
[3]Peng Cheng Laboratory, Shenzhen, China
qlwang15@outlook.com, wenzhiyuan2012@gmail.com
zhaoqin@hit.edu.cn
min.yang@siat.ac.cn, xuruifeng@hit.edu.cn

Abstract

Aspect term extraction aims to extract aspect terms from a review sentence that users have expressed opinions on. One of the remaining challenges for aspect term extraction resides in the lack of sufficient annotated data. While self-training is potentially an effective method to address this issue, the pseudo-labels it yields on unlabeled data could induce noise. In this paper, we use two means to alleviate the noise in the pseudo-labels. One is that inspired by the curriculum learning, we refine the conventional self-training to progressive self-training. Specifically, the base model infers pseudo-labels on a progressive subset at each iteration, where samples in the subset become harder and more numerous as the iteration proceeds. The other is that we use a discriminator to filter the noisy pseudo-labels. Experimental results on four SemEval datasets show that our model significantly outperforms the previous baselines and achieves state-of-the-art performance.

1 Introduction

Aspect term extraction (ATE) is a crucial task in aspect-level sentiment analysis, aiming to extract all aspect terms present in the sentence (Pontiki et al., 2014). For example, given a restaurant review *"I looove their eggplant pizza, as well as their pastas!"*, ATE system aims to extract *"eggplant pizza"* and *"pastas"*.

Many researchers typically formulated ATE as a sequence labeling problem or a token-level classification problem. The current state-of-the-art neural models can be classified into two categories. One designs the sophisticated model with a variety of techniques, such as history attention (Li et al., 2018), sequence to sequence (Ma et al., 2019), and constituency lattice (Yang et al., 2020). Although these models achieve satisfactory performance, its sufficient condition is the availability of sufficient training data. However, labeling a large amount of aspect data may not be practical due to its cost.

The other aims at addressing the data insufficiency issue from a different perspective. For example, Li et al. (2020) generated the reviews while preserving the original aspects via formulating the data augmentation as a conditional generation task. However, varying only a small number of consecutive non-aspect words in the reviews would limit the semantic diversity of the sample. Chen and Qian (2020) tackled long-tail distributions problem of aspect terms and context words in the training sets with soft prototypes. Nevertheless, the way that the soft template implicitly uses external data discounts the usefulness of external data.

Different from previous approaches, in this paper, we use self-training (Scudder, 1965) to alleviate the labeled data insufficiency. In self-training, a base model trained on the labeled data is used to infer pseudo-labels on the unlabeled data, and then a new base model is trained to optimize the loss on human labels and pseudo-labels jointly. We can iterate this algorithm a few times by using new base model to relabel the unlabeled data and retraining a new base. Hence, we can supplement the labeled data with some pseudo-labeled data.

However, the approaches relying on self-training typically suffer from *noise* induced by pseudo-labels. In this paper, we use two means to mitigate the negative effects of pseudo-labels. One is to refine the conventional self-training to *progressive self-training*. Here, we use a progressive subset at each iteration instead of the entire unlabeled data set. During the iterative process, the unlabeled samples in the subset become harder and more numerous. Our motivation stems from curriculum learning (Bengio et al., 2009), where we expect to infer pseudo-labels for unlabeled data in the order of easy to hard and few to many. In this process, easy unlabeled data will bring in little noise, and model that have been previously learned will be

* Corresponding author.

better and thus generate less noise at later stages.

The other is to use a discriminator to filter out as much noise as possible from the pseudo-labels. Inspired Mao et al. (2021), we construct a question sequence based on sentence and a pseudo-label aspect term in sentence that is fed to the discriminator to make true-false prediction. To train the discriminator, we fabricate several positive and negative samples from the training set. Here, the negative samples are constructed based on the left and right boundary errors of aspect terms and the errors of non-aspect terms with the same POS tag as aspect terms. A pseudo-labeled sentence is used to train the new base model only if its all aspect terms are true; otherwise, it is filtered out. Moreover, we can also apply self-training method to train the discriminator to enhance its discriminatory power.

Our method follows multiple steps. Step 1: train a base model on the labeled data; Step 2: divide the unlabeled data into progressive subsets for curriculum learning based on the difficultness and quantity; Step 3: synthesize positive and negative samples on the train set to train a discriminator; Step 4: use the base model to infer the pseudo-labels of the samples in current unlabeled subset, and then filter the noisy pseudo-labels with discriminator; Step 5: retrain a new base model (discriminator) using the labeled data and the filtered pseudo-labeled data. Steps 4 and 5 are repeated until the progressive subsets are exhausted (i.e., the curriculum learning is completed).

Overall, we make the following contributions: (a) To the best of our knowledge, we are the first to use self-training to address the problem of insufficient labeled data in ATE; (b) To mitigate the noise introduced by self-training, we refine the general self-training to progressive self-training and bring in a discriminator to filter the noisy pseudo-labels; (c) Experimental results on four ATE datasets show that our method outperforms the baselines and achieves the state-of-the-art performance. Furthermore, we conduct extensive experiments to verify its effectiveness and generalization.

2 Related Work

Aspect Term Extraction Earlier research endeavors focused on exploiting pre-defined rules (Hu and Liu, 2004; Wu et al., 2009), hand-craft features (Liu et al., 2012), or prior knowledge (Chen et al., 2014) to solve ATE. Recently, researchers employed some deep learning and parse techniques to ATE, such as LSTM (Liu et al., 2015), CNN (Xu et al., 2018), Attention (Li et al., 2018), BERT (Xu et al., 2019), and constituency parsing (Yang et al., 2020). A recent trend is towards the unified framework (Li et al., 2019; Mao et al., 2021). So far, one of the remaining challenges for ATE is the insufficient of annotated data, especially as neural models become more large and more complex. To address this issue, Li et al. (2020) presented a conditional data augmentation approach for ATE. In addition, to solve the data sparsity problem, Chen and Qian (2020) introduced soft prototypes trained by internal or external data. In this paper, we focus on the the insufficient of labeled data scenario, and alleviate it via self-training and unlabeled data.

Self-training The self-training proposed by Scudder (1965) is a semi-supervised approach that leverages unlabeled data to create better models. Self-training first trains a base model on a small amount of labeled data; then utilizes it to pseudo-label unlabeled data, and uses pseudo-labels data to augment the labeled data; finally iteratively retrains the model. Recently, it yields state-of-the-art performance on machine learning tasks like image classification (Zoph et al., 2020), few-shot text classification (Mukherjee and Awadallah, 2020), and neural machine translation (He et al., 2019). The error propagation (Wang et al., 2021) from noisy pseudo-labels is an obvious problem in self-training. In this paper, we alleviate noisy in the pseudo-labels by using progressive subsets (i.e., curriculum learning) and a discriminator.

Curriculum Learning Learning from easier samples first and harder samples later is a common strategy in curriculum learning (Bengio et al., 2009). Our progressive self-training method focuses on easier samples in the early stage, and uses hard samples in the later stage. Our aim is to reduce the noise in the pseudo-labels: the pseudo-labels for easy examples are less prone to errors, and model that have been previously learned could yield more accurate pseudo-labels at later stage.

3 Method

3.1 Problem Formulation

Given a token sequence $x = \{x_1, x_2, ..., x_n\}$ of length n, the ATE task can be characterized as a token-level classification problem. The ATE model takes x as input and outputs a label sequence $y = \{y_1, y_2, ..., y_n\}$, where $y_i \in \{B, I, O\}$ is

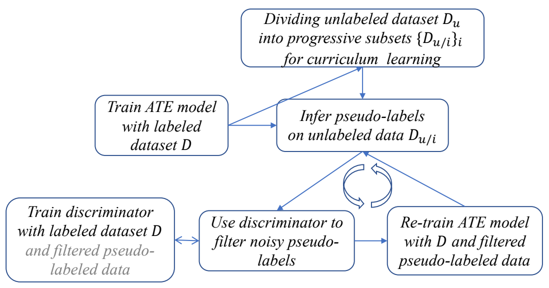

Dividing unlabeled dataset D_u into progressive subsets $\{D_{u/i}\}_i$ for curriculum learning
Train ATE model with labeled dataset D
Infer pseudo-labels on unlabeled data $D_{u/i}$
Train discriminator with labeled dataset D and filtered pseudo-labeled data
Use discriminator to filter noisy pseudo-labels
Re-train ATE model with D and filtered pseudo-labeled data

Figure 1: An illustration of our method. "filtered pseudo-labeled data" indicate that they are unavailable when the iteration is not started. The progressive subsets means that the samples in the subset are becoming harder and more numerous for curriculum learning.

used to indicate if the corresponding token is at the beginning, inside or outside of an aspect term. Given a labeled dataset $D = \{(x^i, y^i)\}_i$ and a unlabeled dataset $D_u = \{\tilde{x}^i\}_i$, our method aims to yield a competitive ATE model.

3.2 Overview

Figure 1 provides an illustration of our method. We first train a base model via the standard cross-entropy loss using the labeled dataset. We then use the base model to estimate the difficultness of the unlabeled samples. Thus, we can divide them into progressive subsets for curriculum learning, where the subsets keep the difficultness increment and the sample amount increment. Meanwhile, we synthesize the training data of the discriminator, and train a discriminator. We then utilize the base model to infer pseudo-labels on the current unlabeled subset. Intuitively, easy unlabeled data is less prone to noise, and previously learned model will be better and thus generate less noise at later stages. In addition, to reduce noise as much as possible, we apply a discriminator to filter noisy pseudo-labels where the filtered pseudo-labeled data can be used to train better discriminator. We then train a new base model by pretraining on the filtered pseudo-labeled data and finetuning on the labeled data. Finally, we iterate this process by using new base model to infer pseudo-labels on the next unlabeled subset.

3.3 Aspect Term Extraction Model

We formulate ATE as a token-level classification task, where for each token x_i in the sentence, our ATE model assigns a label y_i. Our ATE model uses BiLSTM (Hochreiter and Schmidhuber, 1997) or BERT (Devlin et al., 2019) as encoder. The encoder takes a sequence of tokens as input, and produces a sequence of contextual hidden states. To obtain the logits, we attach a linear layer to the end of the encoder. During the training phase, the encoder and the linear layer are trained by minimizing the cross-entropy loss:

$$\ell(x, y) = \frac{1}{n}\sum_{i}^{n} \mathrm{CE}(f_{ATE}(x_i, \theta_{ATE}), y_i) \quad (1)$$

where CE is the cross-entropy loss function, f_{ATE} denotes our ATE model parameterized by θ_{ATE}, and n is the length of the token sequence.

In the inference phase, the ATE model predicts the sequence of labels with the following equation:

$$\widehat{y}_i = \underset{\widehat{y}_i}{\arg\max}\ \mathrm{softmax}(f_{ATE}(x_i, \theta_{ATE})) \quad (2)$$

3.4 Progressive Subsets

The conventional self-training performs inference on all unlabeled data, which undoubtedly leads to much noise. Inspired by the curriculum learning (Bengio et al., 2009), we refine the conventional self-training into the progressive self-training. We assume that in the early stages, easy unlabeled samples are not prone to induce noise, and in the late stages, the learned model has been better and will reduce noise generation on hard unlabeled samples. To divide the unlabeled data into progressive subsets, we define the difficultness of the samples based on the *average logit* of the tokens. We consider that the larger the logit, the more information it contains and the more confident the predictions of the model will be, and hence the easier the unlabeled sample.

$$g_i = f_{ATE}(x_i, \theta_{ATE}) \;\; degree = \frac{1}{n}\sum_{i}^{n} g_{i[\widehat{y}_i]} \quad (3)$$

where $g_i \in \mathbb{R}^3$ is the logit vector of the token x_i, and $g_{i[\widehat{y}_i]}$ is a logit value corresponding to prediction $\widehat{y}_i$. *degree* indicates the difficultness of the sample, and the larger the value the easier the sample is. In addition, we find that the progressive subset size is kept incremental in favor of performance improvement.

3.5 Discriminator

Intuitively, filtering out all the noise in the pseudo-labels accurately and automatically is not quite realistic. We can only filter the noise as much as possible, and to this end, a discriminator is introduced. It makes a true-false determination for each of the inferred aspect terms based on the corresponding contextual. Subsequently, we evaluate whether the sample is suitable for re-training the base model based on the discrimination results of all aspect terms in the sample.

Inspired Mao et al. (2021), we formulate this identification task as a question answering problem, where for each sentence we ask in turn whether the inferred aspect term is true and we expect the response to be affirmative or negative. To derive a suitable input, we pack sentence and custom question as an input sequence. The input sequence is obtained as follows: a [CLS] token is added to the token sequence at the beginning, and two [SEP] tokens are inserted at the end of both the sentence and the custom question, respectively. For instance, we can derive an input sequence based on the above review: *[CLS] I looove their eggplant pizza , as well as their pastas ! [SEP] Is " pastas " an aspect term in the sentence ? [SEP]*

For simplicity, the encoder of the discriminator is identical to that of the ATE model. Here, the final hidden state corresponding to [CLS] token is used as the aggregate sequence representation and fed into the classifier. Suppose the dataset $D_d = \{(x^i, a^i, y^i)\}_i$ where x is a sentence, a is an aspect term in x, and $y \in \{0, 1\}$ is the label of a, we can optimize the discriminator by the following equation:

$$\ell(x, a, y) = \mathrm{BCE}(f_{dis}(x, a, \theta_{dis}), y) \qquad (4)$$

where BCE is the binary cross-entropy loss function and f_{dis} is a discriminator parameterized by θ_{dis}. Subsequently, the trained discriminator is used to do true-false determination for each inferred aspect term $\tilde{a}$ to filter the noisy pseudo-labels.

$$\tilde{y} = \mathrm{INT}(\mathrm{sigmoid} f_{dis}(\tilde{x}, \tilde{a}, \theta_{dis}) >= 0.5) \qquad (5)$$

where INT maps true and false to 0 and 1, respectively.

However, we can only obtain positive samples in D_d from the ATE dataset, but not negative samples. We observe that the wrong aspect terms tend to be boundary errors and non-aspect term errors. Inspired by this observation, we synthesize negative samples based on left and right boundary errors and the errors of non-aspect terms with the same POS tag[1] as aspect terms. Table 1 gives examples of wrong aspect terms.

3.6 Training

We first train a base model on labeled data and use the average logit from the base model to partition the unlabeled data into progressive subsets; second, train a discriminator on the synthetic dataset; third, infer pseudo-labels on the current unlabeled subset and filter noisy aspect terms with discriminator; then train a new base model and discriminator using both labeled data and filtered pseudo-labeled data; and finally, apply this new base model to the next unlabeled subset. To understand our method clearly, Algorithm 1 procedure is presented.

Correct Aspect Term	Wrong Aspect Term	Error Type
black keyboard	keyboard	E1
	black	E2
	notebook	E3

Table 1: Examples of different error types. E1, E2 and E3 denote *left-boundary errors*, *right-boundary errors* and *non-aspect term errors*, respectively. In the sentence *"the newer black keyboard took a little bit ......, but it is still a great notebook!"*, *notebook* and *black keyboard* have the identical POS tag.

Algorithm 1 progressive self-training with discriminator

```
INPUT: labeled data $D = \{(x^i, y^i)\}_i$; unlabeled data $D_u = \{\tilde{x}^i\}_i$; empty set $D_{fu} = \{\}$
OUTPUT: base model $f_{ATE}$
 1: train a base model $f_{ATE}$ using $D$ via Eq. 1
 2: divide $D_u$ into progressive subsets $\{D_{u/i}\}_1^T$ via Eq. 3
    where $||D_{u/i+1}|| > ||D_{u/i}||$    ▷ $||\cdot||$ is difficultness and quantity.
 3: construct $D_d = \{(x^i, a^i, y^i)\}_i$ by synthesizing positive
    and negative samples from $D$
 4: train a discriminator $f_{dis}$ using $D_d$ via Eq. 4
 5: for each subset $D_{u/i}$ in $D_u$ do
 6:     use $f_{ATE}$ to infer pseudo-labels on $D_{u/i}$ via Eq. 2,
    thus $D_{u/i} = \{(\tilde{x}^i, \widehat{y}^i)\}_i$
 7:     use $f_{dis}$ to filter the noise in the pseudo-labels via
    Eq. 5, and obtain $D'_{u/i} \in D_{u/i}$
 8:     $D_{fu}$ += $D'_{u/i}$
 9:     $D_d$ += $\{(\tilde{x}^i, \tilde{a}^i, \tilde{y}^i)\}_i$ by synthesizing positive and
    negative samples from $D'_{u/i}$
10:     train a new base model $f_{ATE}$ by pre-training on $D_{fu}$
    and fine-tuning on $D$ via Eq. 1
11:     retrain $f_{dis}$ using $D_d$ via Eq. 4
12: end for
13: return $f_{ATE}$
```

4 Experiments

4.1 Datasets

We conduct experiments on four datasets from SemEval 2014 Task 4 (Pontiki et al., 2014), SemEval 2015 Task 12 (Pontiki et al., 2015), and SemEval 2016 Task 5 (Pontiki et al., 2016). Statistics of the datasets are presented in Table 2. In addition, as Xu et al. (2018) did, we randomly hold out 150 examples from the train set as the validation set for tuning hyper-parameters. We employ the F1 metric to evaluate the performance of the models.

[1]We use NLTK to derive the POS tag of each token.

	Lap14		Res14		Res15		Res16	
	train	test	train	test	train	test	train	test
#Sent	3045	800	3041	800	1315	685	2000	676
#Aspect	2342	650	3686	1134	1209	547	1757	622

Table 2: Statistics of datasets. #Sent and #Aspect denote the number of sentence and aspect, respectively.

We select the first 2,754 and 6,754 samples from Amazon Cell Phones and Accessories dataset[2] (He and McAuley, 2016) and Yelp Review dataset[3] (Zhang et al., 2015), respectively. The former is treated as unlabeled data in the laptop domain, while the latter is considered as unlabeled data in the restaurant domain. After these samples are preprocessed[4], we can obtain $10k$ unlabeled data.

4.2 Implementation Details

We choose two representative encoders (BiLSTM and BERT) as the backbone to implement our method[5]. For BiLSTM encoder, the word embeddings are initialized with GloVe-840B-300d (Pennington et al., 2014). The hidden size is set to 300, and we use Adam (Kingma and Ba, 2014) with the learning rate of 1e-4 to optimize parameters. For BERT encoder, we use the BERT$_{\text{base}}$ with 12 attention heads, 12 hidden layers and the hidden size of 768, resulting into 110M pretrained parameters. During the fine-tuning process, we employ AdamW (Loshchilov and Hutter, 2018) to optimize parameters. The learning rates are 3e-5 and 3e-4 for the pre-trained parameters and the added parameters, respectively. In addition, we set batch size to 48 and dropout rate to 0.1. For the progressive set $\{D_{u/i}\}_{i=1}^{T}$, we set T to 4; and for each subset size, we set $|D_{u/i}| = i * 1k$. We run all experiments in a single Tesla V100S GPU.

4.3 Baselines

To evaluate the effectiveness of our method, we compare it with four groups of baselines. The first group of baselines are the SemEval winners. **IHS-RD** (Chernyshevich, 2014), **DLIREC** (Toh and Wang, 2014), **EliXa** (San Vicente et al., 2015) and **NLANGP** (Toh and Su, 2016) are the winners for Lap14, Res14, Res15, and Res16 datasets, respectively. The second group of baselines generally employs neural networks with complex structures to solve ATE, such as **MIN** (Li and Lam, 2017), **HAST** (Li et al., 2018), **Seq2Seq4ATE** (Ma et al., 2019), **DECNN** (Xu et al., 2018), and **CLATE** (Yang et al., 2020). The third group of baselines aims to tackle the problem of insufficient annotated data, such as conditional data augmentation (**CDA**) (Li et al., 2020) and soft prototype trained on external data (**SoftProtoE**) (Chen and Qian, 2020). The last group of baselines is our customized model for clear comparison. **BiLSTM(BERT, BERT-PT)-TC** uses the BiLSTM (pre-trained BERT, post-trained BERT-PT (Xu et al., 2019)) with a linear layer for token classification. **BERT-RC** (Mao et al., 2021) treats ATE as a reading comprehension task.

	Lap14	Res14	Res15	Res16
IHS-RD	74.55	79.62	-	-
DLIREC	73.78	84.01	-	-
EliXa	-	-	70.04	-
NLANGP	-	-	67.12	72.34
MIN	77.58	-	-	73.44
HAST	79.52	85.61	71.46	73.61
Seq2Seq4ATE♠	79.02	84.08	69.89	72.82
DECNN♠	81.39	86.04	71.18	74.39
CDA♦	81.58	-	-	75.19
SoftProtoE♠	83.19	*87.39*	73.27	76.98
BiLSTM	72.16	81.23	63.58	66.10
+CDA♦	74.28	-	-	71.44
+SoftProtoE♠	74.75	84.27	66.06	69.65
+our method	74.86	84.86	65.13	70.32
CLATE$_{\text{BERT}}$♡	80.45	-	-	74.32
BERT-TC	80.32	84.45	69.24	74.04
+CDA♦	81.14	-	-	75.89
+our method	84.17†	87.63†	72.81	77.09†
BERT-RC	81.84	85.27	70.16	75.47
+our method	85.01†	88.15†	72.98†	77.51†
CLATE$_{\text{BERT-PT}}$♣	*85.61*	-	-	*81.14*
BERT-PT	84.23	86.32	73.85	78.32
+SoftProtoE♡	85.01	87.10	*73.99*	78.85
+CDA♦	85.33	-	-	80.29
+our method	**86.91**†	**88.75**†	**75.82**†	**82.56**†

Table 3: F1-score (%) obtained on the test set for all methods. Results for the first six methods are taken from Li et al. (2018); ♠: results from Chen and Qian (2020); ♦: results from Li et al. (2020); ♣: results from Yang et al. (2020); ♡: results from our reproduction; Other results are the average scores of three runs with random initialization. + denotes the method combined with the benchmark model; † indicates that the score is significantly better than that of the customized baseline at significance level $p < 0.01$. The scores of best baselines are *italicized*, and the best scores are in **bold**.

4.4 Main Results

The main experimental results on four datasets are reported in Table 3. We can draw the following conclusions from the table. First, our method sub-

[2]https://jmcauley.ucsd.edu/data/amazon

[3]https://www.yelp.com/dataset

[4]It mainly consists of dividing clauses based on symbols (e.g., periods, question marks, and exclamation points), and word completions (e.g., replacing *cant* with *can't*).

[5]Our code is available at: https://github.com/qlwang25/progressive_self_training

stantially enhances our custom baselines. For example, although BERT-TC achieves competitive performance among baselines, our method further achieves 3.85%, 3.18%, 3.57%, and 3.05% absolute gains on four datasets. Second, compared to BiLSTM, the performance of the baselines based on the pre-trained models is more significantly improved when combined with our method. We attribute this phenomenon that the pre-trained models could better alleviate the noise in the pseudo-labels. This also proves the result of Du et al. (2020) that the combination of pre-training and self-training can further improve performance. Third, BERT-PT exceeds most existing ATE models by a great margin, confirming the power of domain-specific post-training. Surprisingly, BERT-PT can be further improved significantly (2.68%, 2.43%, 1.97%, 4.23%) and reach a new state-of-the-art when combined with our method. Finally, our method is obviously more effective than SoftProtoE and CDA in alleviating the insufficient labeled data, and it is also notable that we use less unlabeled data (2,754 vs. 100,000).

	Lap14	Res14	Res15	Res16
BERT-TC	80.32	84.45	69.24	74.04
+ST	81.65	85.72	70.01	75.65
+ST&Dis	83.13	*87.11*	*71.93*	76.08
+PST	*83.21*	86.73	71.91	*76.13*
+our model	**84.17**	**87.63**	**72.81**	**77.09**

Table 4: Ablation studies (F1 scores) on the components of our method. **ST**: conventional self-training; **ST&Dis**: conventional self-training method with discriminator (remove line 2 and for loop several times until convergence); **PST**: progressive self-training method without discriminator (remove line 3, 4, 7, 9, 11, and $D'_{u/i} = D_{u/i}$). Our method is equivalent to the combination of PST and Dis.

4.5 Ablation Studies

Compared to conventional self-training, our method differs in two aspects: based on the average logit of tokens, the unlabeled samples are divided into progressive subsets for curriculum learning, and a discriminator is used to filter as much noise as possible from the pseudo-labels. To verify the validity of these two points, we create three variants for conducting ablation studies. As shown in Table 4, all variants exceed the baseline, suggesting that the use of unlabeled data is helpful, even when strong language model is encountered. In addition, a modest gain (1.48%, 1.39%, 1.92%, 0.43%) over the peer is observed when self-training combined with discriminator, which shows that the discriminator improves the quality of pseudo-labeled data and thus the model performance by reducing noise. Among the three variants, progressive self-training is the best overall, suggesting that the quality of pseudo-labels can be effectively improved through the curriculum learning idea. Combining progressive self-training with discriminator can further improve performance, showing that both can complement each other in promoting the quality of pseudo-labeled data.

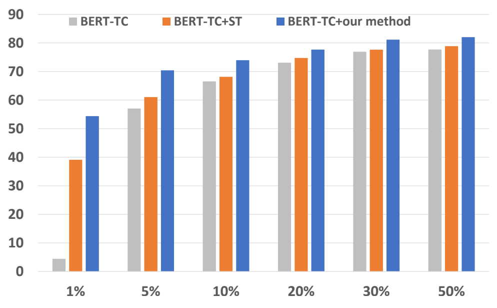
	Lap14		Res14	
	Acc.	F1	Acc.	F1
E1	72.52	70.64	74.42	72.16
E2	75.19	73.12	76.19	75.85
E1&E2	*88.21*	*87.18*	*88.52*	*87.44*
E1&E3	85.54	84.21	85.28	84.16
E2&E3	85.77	83.82	85.15	84.01
E1&E2&E3	**90.26**	**90.12**	**91.27**	**91.10**

Table 5: Ablation studies (accuracy and F1 scores) on the error rules. **E1**, **E2**, and **E3** denote *left-boundary errors*, *right-boundary errors*, and *non-aspect term errors with the same POS tag*, respectively. The best scores are in **bold** and the second-best scores are in *italics*. **Note that** we apply three error rules to synthesize negative samples in the **test set**.

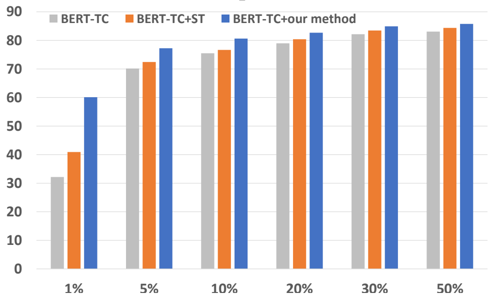
To train the discriminator, we synthetic negative samples from labeled data under three error rules (Table 1). To verify the effectiveness of each rule, we conduct relevant ablation studies. As shown in Table 5, the discriminator achieves substantial gains on the combination of E1 and E2. This indicates that negative samples of the boundary error type play an essential role in training the discriminator. Additionally, the addition of E3 can improves the performance a bit more, showing that the error type of non-aspect terms is useful and reasonable.

4.6 Discussion

Performance on Different Amounts of Labeled Data To investigate the performance of our method when lack of labeled data, we intentionally control the amount of reviews in labeled data and run evaluations with the new training set. As shown in Figure 2, we observe that our method can significantly improve the scores compared to using only a small amount of labeled data (4.41% vs. 54.39% on Lap14 dataset, 57.03% vs. 70.4% on Res14 dataset). Moreover, our method substantially outperforms the conventional self-training method. In particular, our method shows a strong superiority when the proportion of original labeled data is less than 10% (39.1% vs. 54.39% on Lap14 dataset, 40.92% vs. 60.11% on Res14 dataset).



(a) on Lap14 dataset



(b) on Res14 dataset

Figure 2: Comparison of our method with conventional self-training (ST) on different amounts of labeled data. The x-axis represents the proportion of the original labeled data, and the y-axis is the F1 scores.

Effect of Progressive Subsets on Performance Inspired by curriculum learning (Bengio et al., 2009), in this paper, we refine the self-training to the progressive self-training. We expect that in the early stages, easy unlabeled data induce less noise, while in the later stages, the model has become better after learning and can generate less noise on hard unlabeled data. To this end, we divide the unlabeled data into progressive subsets according to the order of increasing difficultness and quantity. To examine our motivation, we conduct relevant comparative experiments. As shown in Table 6, we observe that using harder and more unlabeled data in the early stages can have a discount on performance. The underlying reason may be the introduction of much noise, which makes the model difficult to learn. Moreover, this verifies the reasonable and effectiveness of the progressive subset from the side.

Effect of Retraining Way on Performance In our algorithm, we first pre-train the model on pseudo-labeled data and then finetune it on labeled data (line 10). Here, we compare with an alternative way which trains the model with labeled

		Lap14	Res14
BERT-base		80.32	84.45
difficultness	easy→hard	**84.15**	**87.64**
	random sampling	83.21	86.72
	hard→easy	82.19	86.02
quantity	less→more	**84.18**	**87.63**
	average	83.15	87.13
	more→less	82.01	86.56

Table 6: Comparison (F1 scores) of progressive subsets with different settings. →: use the order of unlabeled data. The difficultness of the unlabeled data is evaluated by the average logit of tokens; The quantity denotes the size of the subset, and $D_{u/i}$=$i * 1k$, $D_{u/i}$=2.5k, and $D_{u/i}$=$(5-i) * 1k$ corresponds to its three settings respectively.

data and pseudo-labeled data jointly. From Table 7, we can find that the combination of pretraining and finetuning slightly exceeds the joint training (84.19% vs. 83.41%). We observe that pre-training only on the pseudo-labeled data leads to lower F1 scores than training only on the labeled data (75.37% vs. 80.32%), suggesting that the distribution of the unlabeled data differs from that of the labeled data. In this case, pre-training first and then fine-tuning can relieve the effect of different data distributions.

		Lap14	Res14
BERT-TC		80.32	84.45
ST	pretraining	73.49	82.91
	joint training	81.44	86.20
	pretraining + finetuning	**81.89**	**86.47**
our method	pretraining	75.37	83.59
	joint training	83.41	86.92
	pretraining + finetuning	**84.19**	**87.61**

Table 7: Comparison (F1 scores) of different retraining ways. ST: conventional self-training method.

Effect of Pre-trained Models of Different Power on Performance As can be seen from Table 3, the combination of self-training and pre-trained models may create more sparks. For further exploration and validation, we conduct comparative experiments using pre-trained models of different power as the backbone. We observe a significant increase in improvement from BERT$_{\text{mini}}$ to BERT$_{\text{base}}$ in Table 8, but the improvement seems to saturate when going from BERT$_{\text{base}}$ to BERT$_{\text{large}}$. Therefore, we can conclude that the self-training method can create more gains when combined with a more capable pre-trained model, but the gains do not always increase as the power of the pre-trained model increases.

Effect of Unlabeled Data Size on Performance We conduct experiments to understand the impact

	Lap14	Res14
BERT$_{mini}$	72.82	76.61
+ST	73.35$_{(0.53\uparrow)}$	77.95$_{(1.34\uparrow)}$
+our method	74.21$_{(1.39\uparrow)}$	78.27$_{(1.66\uparrow)}$
BERT$_{medium}$	80.11	84.13
+ST	81.75$_{(1.64\uparrow)}$	85.56$_{(1.43\uparrow)}$
+our method	82.07$_{(1.96\uparrow)}$	86.19$_{(2.06\uparrow)}$
BERT$_{base}$	80.32	84.45
+ST	81.89$_{(1.67\uparrow)}$	86.47$_{(2.02\uparrow)}$
+our method	84.11$_{(3.79\uparrow)}$	87.61$_{(3.16\uparrow)}$
BERT$_{large}$	82.24	84.91
+ST	83.77$_{(1.53\uparrow)}$	87.32$_{(2.41\uparrow)}$
+our method	84.97$_{(2.73\uparrow)}$	88.12$_{(3.21\uparrow)}$

Table 8: Results (F1 scores) of the combination of self-training method and different pre-training models. ST: conventional self-training. The numbers in parentheses are the improvement after using self-training.

of using different amounts of unlabeled data. We start with no unlabeled data, and then gradually increase the amount of unlabeled data. As shown in Figure 3, the performance increases significantly until the amount of unlabeled data is $10k$, and then increases slowly. Thus, we can conclude that using a large amount of unlabeled data can facilitate the performance improvement, but the improvement slows down gradually.

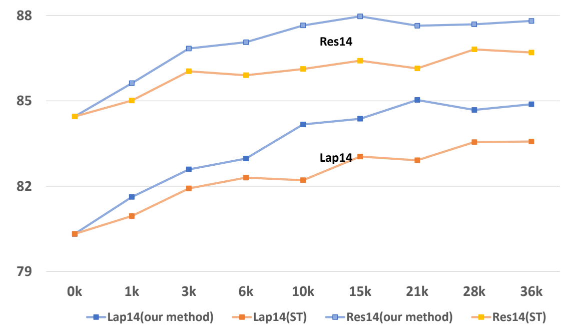


Figure 3: Comparison of different unlabeled data sizes. ST: conventional self-training method; The x-axis is the unlabeled data size and the y-axis is the F1 score.

Case Study We present the predictions of the models on three random examples in Table 9. We can see that our method indeed corrects the predictions of the baseline. In addition, we discover that over-correction (e.g., *staff person→staff*) and under-correction (e.g., *pie company→pie*) problems occur with the conventional self-training method, which we attribute to the introduction of too much noise. The cases on pseudo-labeled data are available from Table 14 in Appendix.

Error Analysis We examine the log files and classify the error predictions into three categories (*under-prediction*, *over-prediction*, and *boundary errors*). We show examples of each category in Table 10 for better understanding. After reviewing

Label	BERT-TC	+ST	+our model
RAM memory	RAM memory 8MB 4MB	RAM memory	RAM memory
win8	win8	–	win8
staff person fork	staff person fork pie company	staff fork pie	staff person fork

Table 9: Case study. ST: the conventional self-training; Red word: incorrect prediction outcome. Sentence 1: *the RAM memory is good but should have splurged for 8MB instead of 4MB.*; Sentence 2: *but I do not like win8.*; Sentence 3: *I had to flag down a third staff person for a fork so now it's goodbye little rude pie company.*

these files, we find that two methods yield some similar errors, suggesting that hard samples are indeed difficult to predict. In addition, we observe a higher percentage of over-prediction than that of the other two types, which may be the underlying reason for the higher recall than precision (91.06% vs. 84.43%).

	Label	BERT-TC+ST	BERT-TC+our model
E1	works	–	–
E2	external mics	external mics iM	external mics iMac
E3	portions of the food	portions food	portions food

Table 10: Error cases. ST: the conventional self-training method; Red word: incorrect prediction outcome; E1: *under-prediction*; E2: *over-prediction*; E3: *boundary errors*. Sentence 1: *super light, super sexy and everything just works.*; Sentence 2: *I never tried any external mics with that iMac.*; Sentence 3: *the portions of the food that came out were mediocre.*

Furthermore, we have the following discussion in Appendix:

- Effect of the Number of Progressive Subsets on Performance
- Effect of Different Incremental Magnitude on Performance
- Fine-grained Named Entity Recognition Experiments
- Comparison of the Parameter Amount and the Computational Complexity

5 Conclusion

In this paper, we focus on the problem of insufficient labeled data in ATE, and try to solve it via self-training. To mitigate the noise in pseudo-labels, we make two efforts. (i) motivated by curriculum learning, we refine the conventional self-training to progressive self-training, expecting to reduce the generation of noisy pseudo-labels; (ii) we introduce a discriminator to filter the noisy pseudo-labels. Experimental results show that our method beats the baselines and achieves SoTA performance. Moreover, we verify its effectiveness and generalization through extensive experiments.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China (61632011, 61876053, 62006062, 62176076), the Guangdong Province Covid-19 Pandemic Control Research Funding (2020KZDZX1224), the Shenzhen Foundational Research Funding (JCYJ20180507183527919 and JCYJ20200109113441941), China Postdoctoral Science Foundation (2020M670912), Joint Lab of HITSZ and China Merchants Securities, Youth Innovation Promotion Association of CAS China (No. 2020357), and Shenzhen Science and Technology Innovation Program (Grant No. KQTD20190929172835662).

References

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *ICML*, pages 41–48.

Zhiyuan Chen, Arjun Mukherjee, and Bing Liu. 2014. Aspect extraction with automated prior knowledge learning. In *ACL (Volume 1: Long Papers)*, pages 347–358.

Zhuang Chen and Tieyun Qian. 2020. Enhancing aspect term extraction with soft prototypes. In *EMNLP*, pages 2107–2117.

Maryna Chernyshevich. 2014. Ihs r&d belarus: Cross-domain extraction of product features using conditional random fields. *SemEval*, page 309.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Jingfei Du, Edouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Ves Stoyanov, and Alexis Conneau. 2020. Self-training improves pre-training for natural language understanding. *arXiv preprint arXiv:2010.02194*.

Junxian He, Jiatao Gu, Jiajun Shen, and Marc'Aurelio Ranzato. 2019. Revisiting self-training for neural sequence generation. In *ICLR*.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pages 507–517.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *ICLR*.

Kun Li, Chengbo Chen, Xiaojun Quan, Qing Ling, and Yan Song. 2020. Conditional augmentation for aspect term extraction via masked sequence-to-sequence generation. In *ACL*, pages 7056–7066.

Xin Li, Lidong Bing, Piji Li, and Wai Lam. 2019. A unified model for opinion target extraction and target sentiment prediction. In *AAAI*, volume 33, pages 6714–6721.

Xin Li, Lidong Bing, Piji Li, Wai Lam, and Zhimou Yang. 2018. Aspect term extraction with history attention and selective transformation. In *IJCAL*, pages 4194–4200.

Xin Li and Wai Lam. 2017. Deep multi-task learning for aspect term extraction with memory interaction. In *EMNLP*, pages 2886–2892.

Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *EMNLP*, pages 1346–1356.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *EMNLP*, pages 1433–1443.

Ilya Loshchilov and Frank Hutter. 2018. Fixing weight decay regularization in adam. In *ICLR*.

Dehong Ma, Sujian Li, Fangzhao Wu, Xing Xie, and Houfeng Wang. 2019. Exploring sequence-to-sequence learning in aspect term extraction. In *ACL*, pages 3538–3547.

Yue Mao, Yi Shen, Chao Yu, and Longjun Cai. 2021. A joint training dual-mrc framework for aspect based sentiment analysis. *arXiv preprint arXiv:2101.00816*.

Subhabrata Mukherjee and Ahmed Awadallah. 2020. Uncertainty-aware self-training for few-shot text classification. *NIPS*, 33.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *EMNLP*, pages 1532–1543.

Maria Pontiki, Dimitrios Galanis, Harris Papageorgiou, Ion Androutsopoulos, Suresh Manandhar, AL-Smadi Mohammad, Mahmoud Al-Ayyoub, Yanyan Zhao, Bing Qin, Orphee De Clercq, et al. 2016. Semeval-2016 task 5: Aspect based sentiment analysis. In *SemEval*, pages 19–30.

Maria Pontiki, Dimitrios Galanis, Harris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *SemEval*, pages 486–495.

Maria Pontiki, Dimitrios Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval*, pages 27–35.

Inaki San Vicente, Xabier Saralegi, Rodrigo Agerri, and Donostia-San Sebastián. 2015. Elixa: A modular and flexible absa platform. *SemEval*, page 748.

Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*, 11(3):363–371.

Zhiqiang Toh and Jian Su. 2016. Nlangp at semeval-2016 task 5: Improving aspect based sentiment analysis using neural network features. In *SemEval*, pages 282–288.

Zhiqiang Toh and Wenting Wang. 2014. Dlirec: Aspect term extraction and term polarity classification system. *SemEval*, page 235.

Xinshao Wang, Yang Hua, Elyor Kodirov, David A. Clifton, and Neil M. Robertson. 2021. Proselflc: Progressive self label correction for training robust deep neural networks. In *CVPR*, pages 752–761.

Yuanbin Wu, Qi Zhang, Xuan-Jing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *EMNLP*, pages 1533–1541.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2018. Double embeddings and cnn-based sequence labeling for aspect extraction. In *ACL (Volume 2: Short Papers)*, pages 592–598.

Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *NAACL: Human Language Technologies, Volume 1*, pages 2324–2335.

Liang Xu, Qianqian Dong, Cong Yu, Yin Tian, Weitang Liu, Lu Li, and Xuanwei Zhang. 2020. Cluener2020: Fine-grained name entity recognition for chinese. *arXiv preprint arXiv:2001.04351*.

Yunyi Yang, Kun Li, Xiaojun Quan, Weizhou Shen, and Qinliang Su. 2020. Constituency lattice encoding for aspect term extraction. In *COLING*, pages 844–855.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *NIPS*, 28:649–657.

Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. 2020. Rethinking pre-training and self-training. *NIPS*, 33.

A Appendix

Effect of the Number of Progressive Subsets on Performance In the above experiments, we split the unlabeled data into four progressive subsets (i.e., $T = 4$). Then a question may arise whether the number of progressive subsets has a significant impact on the method performance. To probe this question, we divide the unlabeled data into different number of progressive subsets for comparison. As shown in Table 11, different numbers of progressive subsets makes only a slight difference to the model performance (e.g., 83.98%→84.14%→84.07%→84.15%→84.32%).

		Lap14	Res14
BERT-TC		80.32	84.45
our method	$T=3$	83.98	87.71
	$T=4$	84.14	87.65
	$T=5$	84.07	87.60
	$T=6$	84.15	87.60
	$T=7$	84.32	87.72

Table 11: Comparison (F1 scores) of different number of progressive subsets. The value T is the number of progressive subsets. **Note that** regardless of the value of T, we try to keep the size of the unlabeled data (i.e., $||D_u||$) consistent for a fair comparison.

Effect of Different Incremental Magnitude on Performance In this paper, we set the incremental magnitude to $1k$ for simplicity, i.e., $D_{u/i} = i * 1k$. We assume that an excessive incremental magnitude should have a positive impact on the model performance in that the progressive subsets are not increasing in size once the magnitude drops to zero. The experimental scores in Table 12 validate our assumptions. Overall, the subset with larger incremental magnitude boosts the model performance more compared to that with smaller incremental magnitude.

		Lap14	Res14
BERT-TC		80.32	84.45
our method	$0.75k$	83.91	87.16
	$1k$	84.14	87.63
	$1.25k$	84.42	87.76

Table 12: Comparison (F1 scores) of different incremental magnitude. **Note that** regardless of the magnitude of the increment, we try to keep the size of the unlabeled data (i.e, $||D_u||$) consistent for a fair comparison.

Fine-grained Named Entity Recognition Experiments To demonstrate our method can be applied to other sequence labeling tasks, we experiment on the fine-grained named entity recognition task (Xu et al., 2020). We consider the first $1k$ samples in the original training set as labeled data and the rest of the samples (9,747) as unlabeled data. Table 13 shows that our method also has advantages over conventional self-training (71.94% vs 69.64%) on the fine-grained named entity recognition task, which confirms the generalizability of our method.

Entities	bert	bert+ST	bert+our	bert*
Address	51.25	52.21	*55.80*	**61.11**
Book	61.64	64.31	*70.83*	**79.23**
Company	68.15	70.67	*72.45*	**78.58**
Game	76.82	75.47	*77.26*	**83.44**
Government	67.78	67.52	*71.59*	**75.05**
Movie	66.86	65.67	*69.31*	**79.73**
Person name	82.04	82.54	*84.15*	**85.63**
Origanization	65.89	67.62	*68.21*	**73.92**
Position	70.30	*74.86*	74.75	**78.38**
Scene	56.16	*65.60*	**66.06**	65.28
Overall	67.67	69.64	*71.94*	**76.27**

Table 13: Comparison (F1 scores) on the fine-grained named entity recognition task (validation set). Here, bert refers to Chinese BERT$_{\text{base}}$; $*$ indicates the use of all annotated data; ST denotes the conventional self-training method. The best scores are in **bold** and the second-best scores are in *italics*.

Comparison of the Parameter Amount and the Computational Complexity The significant time cost of our method is mainly attributed to two aspects: ATE model and discriminator need to be retrained after each subset is used (line 10 and 11 of Algorithm). For clarity of exposition, we conduct relevant experiments on the Res15 dataset and 10k unlabeled data. The parameter amounts for our method and the conventional self-training (ST) method are $218M$ and $109M$, respectively. The main reason for this large difference is that our method includes a discriminator to filter out noise in the pseudo-labels. In addition, training our method and ST method requires $56min$ and $17min$ respectively (both have the same hyper-parameters). We can see that our model takes several times as many hours as the ST method because of requiring retraining the baseline several times. However, it is worth noting that both take the same time during the inference phase. This is because our method involves only ATE model in practical inference. For example, both our method and ST method take $4s$ to infer Res15 test set (685 samples).

Unlabeled Data	Sentence
D_u	they look good and stick good !
	i just do n ' t like the rounded shape because i was always bumping it and siri kept popping up and it was irritating .
	these stickers work like the review says they do .
	however , i ordered these buttons because they were a great deal and included a free screen protector .
	especially having nails , it helps to have an elevated key .
	these make using the home button easy .
	people ask where i got them from it ' s great when driving .
	battery charges with full battery lasts me a full day .
	easy access to all buttons and features , without any loss of phone reception .
	it is a genuine blackberry charger .

(a) Examples of pseudo-labeled data of the conventional self-training method.

Unlabeled Data Subset	Sentence	Discriminate
$D_{u/1}$	the igo bluetooth keyboard works great .	✓
	good headset , good sound , great price .	✓
	good headset , good sound , great price .	✓
	good headset , good sound , great price .	✓
$D_{u/2}$	no issues at all with this battery order .	✗
	it has loud speakers and eliminates background noises .	✓
	you can turn the ear piece off then power on to answer in order to keep your fav ring tone .	✓
	this thing works good , but its not all fireworks and hotel parties .	✗
$D_{u/3}$	also i have n ' t had any complaints from other friends i ' ve talked to with the headset .	✗
	i have owned 2 of these and its the best bluetooth i have used .	✓
	there is a difference in usb cables .	✓
	it does nothing to improve your signal .	✓
$D_{u/4}$	battery lasts a couple of weeks without recharging .	✓
	it was comfortable and transmission was good .	✓
	i never leave home without my ipad and this most useful stylus .	✓
	this is a nice charger but you can tell it was made cheaply in china .	✓

(b) Examples of pseudo-labeled data of our method. The last column is the results of the discriminator.

Table 14: The phrase with color indicates the pseudo-labeled aspect terms; The green and red (manual inspection) indicate correct and incorrect pseudo-labels respectively.