STAD: Self-Training with Ambiguous Data for Low-Resource Relation Extraction

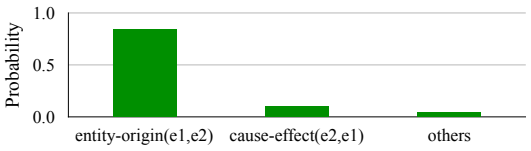
Junjie Yu[1] **Xing Wang**[2] **Jiangjiang Zhao**[3] **Chunjie Yang**[3] **Wenliang Chen**[1]
[1]Institute of Artificial Intelligence, School of Computer Science and Technology, Soochow University, China
[2]Tencent AI Lab, Shenzhen, China
[3]China Mobile Online Marketing and Services Center, China
jjyu@stu.suda.edu.cn, brightxwang@tencent.com, wlchen@suda.edu.cn
{zhaojiangjiang, yangchunjie}@cmos.chinamobile.com

Abstract

We present a simple yet effective self-training approach, named as STAD, for low-resource relation extraction. The approach first classifies the auto-annotated instances into two groups: confident instances and uncertain instances, according to the probabilities predicted by a teacher model. In contrast to most previous studies, which mainly only use the confident instances for self-training, we make use of the uncertain instances. To this end, we propose a method to identify ambiguous but useful instances from the uncertain instances and then divide the relations into candidate-label set and negative-label set for each ambiguous instance. Next, we propose a set-negative training method on the negative-label sets for the ambiguous instances and a positive training method for the confident instances. Finally, a joint-training method is proposed to build the final relation extraction system on all data. Experimental results on two widely used datasets SemEval2010 Task-8 and Re-TACRED with low-resource settings demonstrate that this new self-training approach indeed achieves significant and consistent improvements when comparing to several competitive self-training systems.[1]

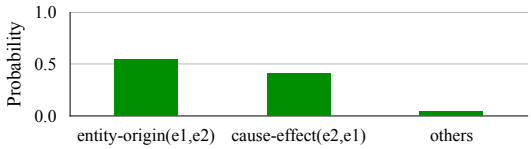
1 Introduction

Relation Extraction (RE) is a fundamental task in Information Extraction, which aims to obtain a predefined semantic relation between two entities in a given sentence (Zhou et al., 2005). In recent years, fine-tuning on the downstream RE tasks with pre-trained models (Soares et al., 2019) has achieved significant progress with the rapid development of the "Pre-train and Fine-tune" Paradigm (Devlin et al., 2019) which leverages large-scale unlabeled data. However, RE still suffers from the data scarcity problem. For most RE tasks, due to the task-specific definition of relations, the lack of customized annotation data poses great challenge for the supervised RE (Hendrickx et al., 2010; Zhang et al., 2017). Meanwhile, manually labeling large-scale RE data is extremely time-consuming, expensive, and laborious. As an alternative, automatically building annotated data for RE attracts a lot of attention in the research community (Mintz et al., 2010; Luo et al., 2019; Yu et al., 2020).



Most of birdtown's earliest residents hailed from eastern european countries.

(a) Confident instance



The taste is from the ginseng , not alcohol .

(b) Uncertain instance

Figure 1: Two examples of auto-annotated instances. For simplicity, we list two detailed relations and use "others" to represent the other relations.

Self-training is a simple and effective approach to build auto-annotated data (Xie et al., 2020; Du et al., 2021). The idea is to use a teacher model trained on human-annotated data to annotate the additional unlabeled data. Then the human-annotated data is combined with some instances selected from the auto-annotated data to train a student model. In this paper, we follow the self-training framework to improve **Low-Resource RE**, which is closer to practical situations where the task starts with a small seed set of human-annotated data.

In previous studies, researchers often select the auto-annotated instances with high confidence,

[1] Code is publicly available at https://github.com/jjyunlp/STAD

Proceedings of the 29th International Conference on Computational Linguistics, pages 2044–2054
October 12–17, 2022.

named as *confident instance*, and have achieved a certain success (Qian et al., 2009; Du et al., 2021). Figure 1(a) shows an example of confident instance, where the teacher model can easily classify it as relation "entity-origin(e1,e2)" with the clue offered by "hailed from". Therefore, we first follow this kind of self-training solutions to train the RE system. However, in the preliminary experiments, we find that *some relations might have similar expressions among instances*, which makes the teacher model confused. As a result, for the uncertain instances, the teacher model gives similar high probabilities to some relations or assigns low probabilities to all the relations. An example of uncertain instance is shown in Figure 1(b), where the teacher model predicts the instance as relation "entity-origin(e1,e2)" with a probability of 56%, as "cause-effect(e2,e1)" (the ground truth label) with 42%, and as other relations with only 2%. It is hard to distinguish between the first two relations as the expression "... is from ..." is often used for both. In previous studies, the uncertain instances are often discarded due to the confusion. However, we argue that ignoring all the uncertain instances may not be appropriate since they may contain useful information. For example, it is a good clue that the answer is one of the first two relations with a probability 98% for the instance in Figure 1(b).

Ideally, we would wish to make full use of all the auto-annotated instances to improve the RE system. But it is very hard due to the confusion problem. Therefore, we split the uncertain instances into two groups: *ambiguous set* and *hard set*. The ambiguous set contains the instances for which the teacher model predicts similar high probabilities on some relations, while the hard set contains those that the teacher model assigns low probabilities to all the relations. In this paper, we focus on the ambiguous set and propose an approach to use the ambiguous instances and the confident instances to improve the system of low-resource RE.

In our approach, we tackle two main issues when exploiting the ambiguous instances: 1) how to identify the candidate labels of ambiguous instances; 2) how to train a new model with the ambiguous instances. For the first issue, we adopt a probabilistic accumulation approach to obtain a set of relations, that is, a set of candidate labels containing the vast majority of probabilities, and then label ambiguous instances with the candidate labels in the set. To deal with the second issue, we make an assumption: *For the ambiguous instances, the teacher model does not know which relation is the exact answer, but it does know that #1) the answer is (with high probability) in the candidate-label set (likes the first two relations in Figure 1(b)) and #2) the answer is not one of the relations which are with very low probabilities (likes "others" in Figure 1(b)).* Under this assumption, we treat the ambiguous instances as partially-labeled instances (Yan and Guo, 2020), where the relations are divided into a candidate-label set and a negative-label set for each ambiguous instance. Then, we propose a novel set-negative training method to learn from the ambiguous instances by the negative-label set. In order to build the final relation extraction system, we further propose a joint training method which supports both positive and set-negative training.

Our main contributions are as follows:

- We propose STAD, a self-training framework, which supports learning from ambiguous data.
- We propose a method to classify the auto-annotated instances generated from teacher model into confident set, ambiguous set and hard set. The ambiguous instances are then treated as partially-labeled, which can reduce the effect of confused expressions. To our best knowledge, it is the first time that partial labeling is used to tag the auto-annotated instances in RE.
- In order to exploit the auto-annotated instances properly, we propose set-negative training for the ambiguous instances and positive training for the confident instances. The set-negative training method can utilize the information that the answer is not in the set of the relations which have very low probabilities predicted by the teacher model. And we further propose a joint training method to combine positive and set-negative training methods to build the final RE system.

To verify the effectiveness of our approach, we conduct experiments on two widely used datasets SemEval2010 Task-8 and Re-TACRED with low-resource settings. Experimental results show that our proposed approach significantly outperforms the conventional self-training system which only samples confident instances and other baseline systems.

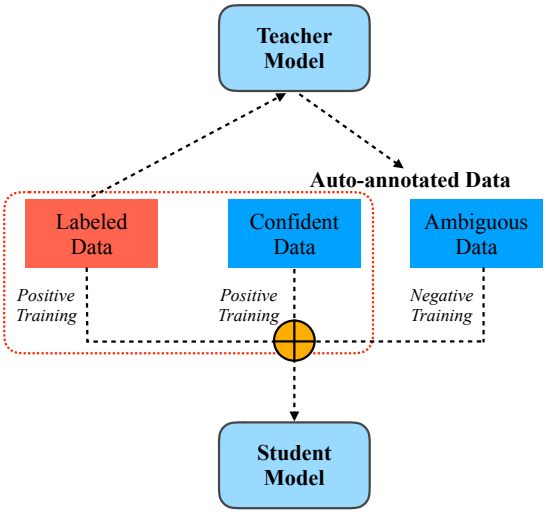


Figure 2: Framework of our approach. The red dotted rectangle is the conventional self-training approach that neglects the ambiguous data in auto-annotated data.

2 Our Approach

We first briefly introduce the relation extraction task as well as the self-training framework frequently used in the previous studies (Zhou et al., 2008). Then, we propose an algorithm to classify the auto-annotated instances into three groups: confident data, ambiguous data and hard data. Next, we introduce three tagging modes for ambiguous data. Finally, we propose the training method to use both the confident and ambiguous data. The framework of the proposed approach is shown in Figure 2.

2.1 Self-Training for Relation Extraction

2.1.1 Relation Extraction

Fine-tuning on a pre-trained model, e.g., BERT, with a task-specific classifier is a common practice for downstream NLP tasks (Devlin et al., 2019). Following Soares et al. (2019), the RE model is composed of a BERT encoder and a relation classification layer. Entity markers ('[E1] head entity [/E1]' and '[E2] tail entity [/E2]') are inserted into input tokens to wrap the entities. Concretely, the output representation of two start entity markers ('[E1]' and '[E2]') are concatenated as the representation of entities. Finally, the representations are used as input for the relation classification layer.

Formally, the output representation of an input instance x after BERT is $\mathbf{h} = \mathbf{h}_{[E1]} \oplus \mathbf{h}_{[E2]}$. Then, the output probability distribution for M relations $p = [p_1, p_2, ..., p_M]$ is computed by the relation classification layer:

$$p = f(x) = \text{Softmax}(\mathbf{W}\mathbf{h} + \mathbf{b}), \quad (1)$$

where $\mathbf{W}$ and $\mathbf{b}$ are model parameters.

During training, each instance x from the human-annotated data is labeled with a one-hot label vector y: a single 1 value for the ground-truth label and 0 values for other labels. Then, the positive training is performed to calculate the cross entropy loss:

$$\mathcal{L}_{PT}(f(x), y) = -\sum_{i=1}^{M} y_i \log p_i, \quad (2)$$

where M is the number of relations, and y_i and p_i are the label and prediction probability of ith relation, respectively.

2.1.2 Self-Training

Generally, in the self-training framework, unlabeled instances are labeled by the teacher model to build the auto-annotated data. As shown in Figure 2, the common-used flow of self-training is performed in the following steps: (1) use the human-annotated data to train a teacher model; (2) use the teacher model to conduct label prediction for unlabeled data; (3) select confident auto-annotated instances via a pre-defined probability threshold (described in Sec. 2.2) and the rest are uncertain instances; (4) combine confident auto-annotated data and human-annotated data to train a student model (the red dotted rectangle in Figure 2).

And in step (3), the remaining uncertain instances are considered to be useless. However, as described in Sec. 1, the uncertain instances (e.g., the example in Figure 1(b)) might contain useful information.

2.2 Instance Classification

As shown in Algorithm 1, we propose a method to classify the auto-annotated data into confident data (Line 14), ambiguous data (Line 16) and hard data (Line 18). In detail, considering a classification task with M classes, we first set a probability threshold T. After sorting the classes by the predicted probability from large to small, we dynamically accumulate the probability of top classes until the score is larger than T (Line 6-10 in Algorithm 1). Then, we move on to the classification of instances:

Confident instance. Using a probability threshold to select confident instances is a common practice in self-training. We also add an instance into the confident data when the highest prediction probability exceeds the threshold T, as the teacher

Algorithm 1 Instance Classification

Input: auto-annotated data $\mathbf{D_{auto}} = \{x, P, Y\}$ containing sentence x, prediction distribution P and its corresponding relations Y.
Hyper Parameter: probability threshold T.
Output: confident data $\mathbf{D_{con}}$, ambiguous data $\mathbf{D_{amb}}$ and hard data $\mathbf{D_{hard}}$

1: **for** $(x, P) \in \mathbf{D_{auto}}$ **do**
2: Let $score = 0.0$.
3: Let candidate-label set $C^+ = \{\}$
4: Sort P from large to small
5: Sort Y by the order in P
6: **for** $(p, y) \in P, Y$ **do**
7: $score \leftarrow score + p$.
8: Append y to C^+.
9: **if** $score > T$ **then**
10: **break**
11: **end if**
12: **end for**
13: **if** len(C^+) == 1 **then**
14: Append (x, C^+) to $\mathbf{D_{con}}$
15: **else if** len(C^+) $\leq M - 1$ **then**
16: Append (x, C^+) to $\mathbf{D_{amb}}$
17: **else**
18: Append x to $\mathbf{D_{hard}}$
19: **end if**
20: **end for**
21: **return** $\mathbf{D_{con}}, \mathbf{D_{amb}}, \mathbf{D_{hard}}$

makes a certain prediction. In Line 13-14 of Algorithm 1, instances with only one candidate label are added into the confident data $\mathbf{D_{con}}$.

Ambiguous instance. Besides the confident data, according to our observations, the teacher model may give relatively high probabilities to multiple relations. In order to make full use of ambiguous instances to improve the RE system, we adopt a greedy probability accumulation method to identify ambiguous data from uncertain data. Specifically, the size of candidate-label set is at least two and can be at most $M - 1$ (Line 15-16 in Algorithm 1). Obviously, as the size of candidate-label set increases, instances carry less information.

Hard instance. We also consider an extreme situation that the sum probability of top $M - 1$ relations is still lower than T. In other words, the lowest predicted probability is larger than $1 - T$. We treat such examples as hard instances because the teacher model is confused about all relations.

Mode	Ent-Ori	Cau-Eff	Others
Probability	0.56	0.42	0.02
Hard Label	1	0	0
Soft Label	0.56	0.42	0.02
Partial Label	1	1	0

Table 1: An example of three tagging modes with given predicted probability distribution.

2.3 Instance Label Tagging Mode

After identifying confident and ambiguous data from the auto-annotated data, we now have three training sets: a small seed set of human-annotated data $\mathbf{D_{hum}}$, a confident auto-annotated data $\mathbf{D_{con}}$, and an ambiguous auto-annotated data $\mathbf{D_{amb}}$. For the human-annotated data and confident data, we take the one-hot label vector as described in Sec. 2.1.1 to tag the data. For the ambiguous data, a variety of methods can be used to tag the instance. As shown in Table 1, given the probability distributions predicted by the teacher model, hard label mode assigns an exact label (the label with highest prediction probability) with a one-hot vector (Lee, 2013) while soft label mode (Xie et al., 2020) adopts probability distributions as labels.

In this work, we propose to use the partial label mode (Yan and Guo, 2020) to tag ambiguous instances. As the teacher model gives relatively high probabilities to relations in the candidate-label set C^+ (Line 3 in Algorithm 1), partial label mode assigns each ambiguous instance with a set of candidate relations and treats each candidate relation equally to form a multi-hot label vector. As the example shown in Table 1, the instance is labeled as [1, 1, 0] because the first two relations are in candidate-label set.

2.4 Model Training

As described in Sec. 2.1.1, we can directly utilize Eqn. 2 to train data tagged in the hard or soft label mode. In this section, we focus on training ambiguous data under partial label mode and describe how to train model on a mixed data.

2.4.1 Set-Negative Training

Inspired by negative training (Kim et al., 2019) which trains noisy data in a negative way by selecting a random label excepting the tagged label, we propose the set-negative training method to train ambiguous data. For the ambiguous data, as described in Sec. 2.2, we first split relations into two

sets: candidate-label set C^+ and negative-label set C^-, where C^- includes the relations that are not in C^+. With the Assumption #2 described in Sec. 1, we are confident that the answer is not in C^-. Hence, we randomly select one label from C^- as a negative label for each ambiguous instance in every iteration to update the model in a negative way.

Formally, the loss function for the ambiguous instances under set-negative training is:

$$\mathcal{L}_{NT}(f(x), y) = -\sum_{i=1}^{M} y_i \log(1 - p_i), \quad (3)$$

where the one-hot label y is dynamically changed by randomly selecting a negative label from C^- for every iteration.

2.4.2 Joint Training

After adopting the set-negative training for ambiguous data, another problem is how to train on $\mathbf{D_{hum}}$, $\mathbf{D_{con}}$, and $\mathbf{D_{amb}}$ simultaneously. Formally, we need to design a unified training framework to support both positive training (Eqn. 2) and set-negative training (Eqn. 3). To this end, we first introduce a flag variable z to represent whether current input instance is partially labeled or not:

$$z = \begin{cases} 1 & \text{if partially labeled,} \\ 0 & \text{others.} \end{cases} \quad (4)$$

Then, the following unified loss function can be directly used for joint training:

$$\mathcal{L}(f(x), y) = -\sum_{i=1}^{M} y_i \log |z - p_i|, \quad (5)$$

where $|*|$ is the absolute value. When the input instance is partially labeled, this function is equivalent to Eqn. 3, otherwise it is equivalent to Eqn. 2.

3 Experiments

In this section, we carry out experiments to show the effectiveness of the proposed approach. We first introduce settings of datasets and parameters. Then, we describe comparison systems used in this work. After that, we present the overall evaluation results, followed by further analysis.

3.1 Datasets

We conduct our experiments on two widely used relation extraction datasets: SemEval 2010 Task-8 (SemEval) and Re-TACRED. The brief information of two datasets are as follows:

- SemEval: A classical dataset in relation extraction which contains 10,717 annotated sentences covering 9 relations with two directions and one special relation "no_relation" (Hendrickx et al., 2010).
- Re-TACRED: A repaired version of TACRED (Zhang et al., 2017) proposed by Stoica et al. (2021) who re-annotated the data and refined relation definitions. In total, it contains 91,467 sentences covering 40 relations (also including a "no_relation" class).

To test on the **low-resource scenario**, we randomly sample 20 instances for each relation from the original training set as the human-annotated data and use the remaining instances as unlabeled data. [2] Meanwhile, we also rebuild the development set by sampling 10 instances for each relation from the original development set which is more realistic. Besides, the special "no_relation" type in the Re-TACRED is excluded as it occupies more than 66% of instances. The statistics of two low-resource datasets are shown in Table 2.

Data	Rel	Train	Dev	Test	Unlabel
SemEval	19	380	190	2,717	6,076
Re-TACRED	39	780	390	5,648	18,938

Table 2: Statistics of SemEval and Re-TACRED with the low-resource setting.

3.2 Parameter Settings

Relation Extraction Model Training To train the relation extraction model, we use the "Entity marker+Entity start state" architecture proposed in Soares et al. (2019) with $\texttt{BERT}_{\texttt{base}}$ (Devlin et al., 2019). During training, we follow Devlin et al. (2019) to select the learning rate among {2e-5, 3e-5, 5e-5}, batch size among {16, 32} and adopt the hyper parameters with the best performance on development set. For other parameters, we simply follow the settings used in Soares et al. (2019) to conduct experiments. We train the model in 20 epochs with early stop strategy to relieve the overfitting problem. Besides, we run each system

[2] For any relation contains less than 20 instances in the original training set, we directly repeat some instances to keep the data balanced.

#	System	Auto-annotated		Micro F1		
		$\mathbf{D_{con}}$	$\mathbf{D_{amb}}$	SemEval	Re-TACRED	Avg. Score
1	SUPERVISED	×	×	$77.3_{\pm 0.8}$	$77.5_{\pm 1.3}$	77.4
2	SELF-TRAINING	✓	×	$79.2_{\pm 1.5}$	$80.5_{\pm 0.4}$	79.9
3	HARD-LABEL	✓	✓	$78.0_{\pm 1.2}$	$78.2_{\pm 1.4}$	78.1
4	SOFT-LABEL	✓	✓	$79.1_{\pm 0.8}$	$79.8_{\pm 2.0}$	79.5
5	STAD (Ours.)	✓	✓	$\mathbf{80.0}_{\pm 1.0}$	$\mathbf{81.6}_{\pm 0.3}$	**80.8**

Table 3: Results on the test set of SemEval and Re-TACRED with low-resource settings in terms of micro F1 score. $\mathbf{D_{con}}$ and $\mathbf{D_{amb}}$ denote confident data and ambiguous data, respectively. We run experiment five times with different random seeds and report the mean of the micro F1 scores (Micro-F1) and its standard deviation score. Some additional experiments with random data samples are included in Appendix A.

five times with random seeds and report the mean of the micro F1 score (Micro-F1) with a standard deviation score.

Auto-Annotated Data Building The probability threshold is an important hyper parameter which is used to select confident data and determine the candidate-label sets of ambiguous data. In detail, we search probability threshold T among {0.95, 0.90, 0.85, 0.80} on development set to select the best confident data.

3.3 Comparison Systems

In order to make a fair comparison, all systems in this work use the BERT-based fine-tuning model proposed by Soares et al. (2019) as relation extraction model. Based on the difference of data parts and tagging strategies, we conduct following models for comparison.

SUPERVISED: A supervised baseline system for our relation extraction task (Soares et al., 2019). This system is trained only on human-annotated data. We also use this system as the teacher model to tag unlabeled data for the following approaches.

SELF-TRAINING: A common-used self-training method (Du et al., 2021), which combines human-annotated and confident data. We re-produce this self-training framework with the same relation extraction model of SUPERVISED system.

HARD-LABEL: A direct comparison of the system proposed by Lee (2013), which extends the SELF-TRAINING system to involve all auto-annotated data. We obtain the class with the highest prediction probability as the label for all auto-annotated data.

SOFT-LABEL: Another extension of SELF-TRAINING system which utilizes the remaining auto-annotated data by assigning probability distributions comes from teacher model as labels (Xie et al., 2020).

3.4 Overall Evaluation Results

Table 3 shows the overall evaluation results on SemEval and Re-TACRED with low-resource settings. Our observations are:

- System SELF-TRAINING significantly and consistently outperforms SUPERVISED with +2.5 on Micro-F1 (79.9 vs. 77.4) in average. The results indicate that self-training with sampling confident data is effective for relation extraction in low-resource scenarios.
- Systems HARD-LABEL and SOFT-LABEL employ the ambiguous data outperform the SUPERVISED system, but can not achieve improvement over the SELF-TRAINING system, demonstrating that it is challenging to achieve improvement with the ambiguous data.
- Our final system STAD significantly outperforms the SUPERVISED system on both datasets with +3.4 (80.8 vs. 77.4) in average. And it also performs better than the SELF-TRAINING system, demonstrating the the effectiveness and the versatility of the proposed approach.

3.5 Extremely Low-resource Scenario

In addition, we conduct additional experiments on **extremely low-resource scenario** with 15, 10 and 5 instances per relation to investigate the effect of the proposed approach. Experimental results are

System	Data Size			
	20	15	10	5
SUPERVISED	77.3	72.0	60.6	45.1
SELF-TRAINING	79.2	77.1	71.3	47.4
STAD	**80.0**	**78.1**	**72.6**	**53.5**
$\triangle_1$	2.7	6.1	12.0	8.4
$\triangle_2$	0.8	1.0	1.3	6.1

Table 4: Results on the test set of SemEval with low-resource settings in terms of Micro F1 score. $\triangle_1$ and $\triangle_2$ represent the absolute improvement when comparing ours with supervised baseline and self-training baseline, respectively.

System	Data Size			
	20	15	10	5
SUPERVISED	77.5	74.1	62.0	47.0
SELF-TRAINING	80.5	79.3	68.4	50.1
STAD	**81.6**	**81.1**	**72.3**	**60.0**
$\triangle_1$	4.1	7.0	10.3	13.0
$\triangle_2$	1.1	1.8	3.9	9.9

Table 5: Results on the test set of Re-TACRED with low-resource settings in terms of Micro F1 score. $\triangle_1$ and $\triangle_2$ represent the absolute improvement when comparing ours with supervised baseline and self-training baseline, respectively.

shown in Table 4 and Table 5. From the two tables, we can find that:

- The performance of SUPERVISED drops rapidly when the amount of training data decreases.
- SELF-TRAINING system works well on 20, 15 and 10, but it only achieves a slight improvement on 5. This indicates that it becomes hard to learn from confident data when teacher model is weak.
- The results of $\triangle_1$ and $\triangle_2$ show that our STAD system outperforms both SUPERVISED and SELF-TRAINING on all settings, especially for the improvement obtained on data size is 5 which indicating the robustness of our system. We think the reason is that when the teacher model becomes weak, there is more and more ambiguous data in the auto-annotated data.

3.6 Ablation Study

Method	Mirco F1
STAD	81.6
– Partial Labeling	-1.4
– Set-Negative Training	-2.3
– Both	-3.4

Table 6: An ablation study for using partial labeling and set-negative training on the test set of Re-TACRED.

To further analyze the effect of partial labeling and set-negative training on the ambiguous data in our proposed approach, we perform ablation studies on the low-resource Re-TACRED dataset and list the results in Table 6.

Without the partial labeling, we label the ambiguous data in the hard label mode and use the negative training to train the model, the performance of our proposed approach degrades by 1.4 F1 score. Without the set-negative training, we label the ambiguous data in partial label mode but with positive training. The strategy results in a drastic drop in F1 score. Removing both the partial labeling and set-negative training, the proposed system deals the ambiguous data in the same way as in HARD-LABEL system in Table 3 and suffers from performance degradation (78.2-81.6=-3.4).

These results demonstrate that the partial labeling and the set-negative training on the ambiguous data indeed contribute to the final performance.

3.7 Distribution of Auto-Annotated Data

Figure 3 shows the distribution of auto-annotated data under our probability accumulation method on Re-TACRED. From the figure we can find that although the number of confident data (the first bar, 44.5%) is much larger than the others, the sum of ambiguous data (others, 55.5%) is also considerable. This indicates that ignoring such a large number of ambiguous data is not appropriate and how to make full use of them is the key to improving self-training. In addition, we find that there is no hard instances in our experiments due to the large number of relation types and the setting of probability threshold.

3.8 Top-N Evaluation

Our method of modeling ambiguous data under negative training helps the model compress the scope of answers. Intuitively, there should be

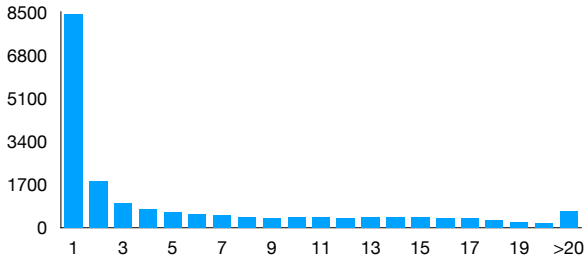


Figure 3: The number of auto-annotated instances according to size of candidate-label sets on Re-TACRED.

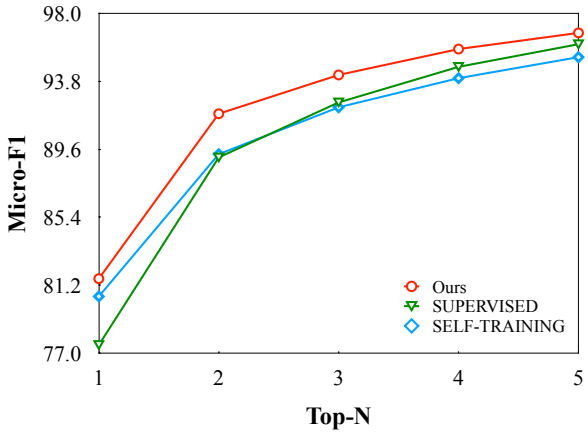


Figure 4: The top-N evaluation on Re-TACRED.

more significant improvements in top-N evaluation. Therefore, we evaluate top-n predictions on Re-TACRED. The results are presented in Figure 4. The figure shows that the SELF-TRAINING method shows a disadvantage in top-N evaluation as its performance gradually degrades when compared to the SUPERVISED baseline. After modeling ambiguous data in a partially annotated and negative training mode, however, our method achieves significant and consistent improvement. In conclusion, our method of modeling ambiguous data can learn the information contained in the ambiguous data.

System	Data Sampling		
	Data 1	Data 2	Data 3
SUPERVISED	$80.3_{\pm 1.6}$	$79.7_{\pm 1.3}$	$79.0_{\pm 1.5}$
SELF-TRAINING	$82.5_{\pm 0.6}$	$84.0_{\pm 1.5}$	$81.4_{\pm 1.7}$
STAD	$\mathbf{84.6}_{\pm 1.5}$	$\mathbf{85.3}_{\pm 1.0}$	$\mathbf{83.9}_{\pm 1.6}$

Table 7: Results on Re-TACRED with different data samples for training and validation sets on the low-resource setting.

3.9 Experiments with Random Data Samples on Re-TACRED

In order to further study the stability of our approach, we conduct extra experiments on Re-TACRED which contains much more data than SemEval. For each experiment, we randomly sample a new split of training and validation sets for the low-resource setting. We report the mean of the micro F1 scores and its standard deviation score of five runs with different random seeds.

3.10 Qualitative Results: Case Study

Table 8 shows examples of ambiguous data in Re-TACRED which are predicated by the teacher model. We present the sentence as well as its relations with two highest predicted probabilities. The teacher model gives relatively high probabilities to the two relations and fails to predicate the ground-truth relation. For example, the teacher model can not distinguish "per:city_of_birth" from "per:country_of_birth" for the first sentence, and the conventional self-training system does not adopt this instance to the training set. But in our approach, this instance is tagged in a partially labeled mode and is used as an ambiguous instance to train the student model.

4 Related Work

Self-Training. Self-training is one of the most commonly used approaches for exploiting unlabeled data (Scudder, 1965; Yarowsky, 1995; McClosky et al., 2006; Lee, 2013). With the development of neural network models and the growth of demand for labeled data, self-training has become a very active field in research. It is widely used to improve the performance of neural machine translation (Zhang and Zong, 2016; Jiao et al., 2020, 2021), question answering (Sachan and Xing, 2018) and low resource dependency parsing (Rotman and Reichart, 2019). In this work, we apply self-training to exploit unlabeled data for low-resource relation extraction. The main difference is our work focuses on the uncertain instances which are often neglected in the previous studies.

Low-Resource Relation Extraction Recently, low-resource relation extraction (LRE) has attracted much attention due to data scarcity problems. There are two main scenarios of LRE: low-shot RE and low-resource supervised RE. Low-shot RE, including few-shot RE (Han et al., 2018) and

Sentence	Relation	Probability
It 's website , lists [Abu Zubaydah]$_{e1}$'s birthplace as [Riyadh]$_{e2}$, Saudi Arabia	per:country_of_birth per:city_of_birth	0.55 0.40
[Parliament]$_{e2}$ speaker [Ali Larijani]$_{e1}$, Iran 's former chief nuclear negotiator.	per:title per:employee_of	0.79 0.16
[Paul Kim]$_{e1}$ (25) Currently lives in [Saratoga]$_{e2}$, CA	per:city_of_birth per:city_of_residence	0.74 0.20

Table 8: Examples of ambiguous data in Re-TACRED."e1" and "e2" are head entity and tail entity respectively. "Probability" is predicted probability given by the teacher model.

zero-shot RE (Levy et al., 2017), trains a model to identify unseen relation labels in a test set with a few support examples or even none of the examples. The low-resource supervised RE is designed to train a supervised model with small training data (Deng et al., 2022). In this work, we also focus on the low-resource supervised RE.

To solve the low-resource problem, one direction is to build a robust model which can make full use of small data and existing knowledge. For example, integrating semantic relation labels (Dong et al., 2021) and introducing knowledge graph (Zhang et al., 2019). Another direction is to enlarge annotated data automatically. Data augmentation is a widely used technology to enrich the data (Xu et al., 2016; Yu et al., 2020). Self-training is also a conventional way to obtain automatically annotated data (Rosenberg et al., 2005; Hu et al., 2021a,b).

Partial Label Learning. In this work, the definition of partial label is a candidate set of labels for an ambiguous instance in a multi-class classification task (Cour et al., 2011). This is different from that in sequence labeling tasks (Li et al., 2014) and multi-label multi-class classification tasks (Xie and Huang, 2018). In order to learn from partially labeled instances, various methods have been proposed to deal with the problem (Nguyen and Caruana, 2008; Cour et al., 2011). Recently, with the help of self-training, Feng and An (2019) proposes a self-guided retraining method to learn from partially labeled data. Besides, Yan and Guo (2020) also proposes to recalculate the confidence of labels in a candidate set by taking the current model as a teacher.

5 Conclusion

This paper proposes a novel self-training approach with ambiguous data (STAD) for the low-resource relation extraction, which fully uses the auto-annotated data. According to the probabilities predicted by the teacher model, we classify the auto-annotated data into three sets: confident set, ambiguous set and hard set. During training, we consider the ambiguous set, which is neglected by the previous studies, and adopt the set-negative training method to alleviate the noise problem. Experimental results show that our proposed system consistently outperforms the self-training systems.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (61876115 and Grant No. 61936010), the Priority Academic Program Development of Jiangsu Higher Education Institutions. The corresponding authors are Wenliang Chen and Jiangjiang Zhao. We thank the anonymous reviewers for their constructive comments.

References

Timothee Cour, Ben Sapp, and Ben Taskar. 2011. Learning from partial labels. *JMLR*.

Shumin Deng, Ningyu Zhang, Hui Chen, Feiyu Xiong, Jeff Z Pan, and Huajun Chen. 2022. Knowledge extraction in low-resource scenarios: survey and perspective. *arXiv preprint arXiv:2202.08063*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186.

Manqing Dong, Chunguang Pan, and Zhipeng Luo. 2021. Mapre: An effective semantic mapping approach for low-resource relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2694–2704.

Jingfei Du, Édouard Grave, Beliz Gunel, Vishrav Chaudhary, Onur Celebi, Michael Auli, Veselin Stoyanov,

and Alexis Conneau. 2021. Self-training improves pre-training for natural language understanding. In *NAACL*.

Lei Feng and Bo An. 2019. Partial label learning with self-guided retraining. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3542–3549.

Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4803–4809.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *SemEval*.

Xuming Hu, Chenwei Zhang, Fukun Ma, Chenyao Liu, Lijie Wen, and S Yu Philip. 2021a. Semi-supervised relation extraction via incremental meta self-training. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 487–496.

Xuming Hu, Chenwei Zhang, Yawen Yang, Xiaohe Li, Li Lin, Lijie Wen, and S Yu Philip. 2021b. Gradient imitation reinforcement learning for low resource relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2737–2746.

Wenxiang Jiao, Xing Wang, Shilin He, Irwin King, Michael Lyu, and Zhaopeng Tu. 2020. Data rejuvenation: Exploiting inactive training examples for neural machine translation. In *EMNLP*.

Wenxiang Jiao, Xing Wang, Zhaopeng Tu, Shuming Shi, Michael Lyu, and Irwin King. 2021. Self-training sampling with monolingual data uncertainty for neural machine translation. In *ACL-IJCNLP*.

Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. 2019. Nlnl: Negative learning for noisy labels. In *ICCV*.

Dong-Hyun Lee. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML*.

Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342.

Zhenghua Li, Min Zhang, and Wenliang Chen. 2014. Ambiguity-aware ensemble training for semi-supervised dependency parsing. In *ACL*, pages 457–467.

Fan Luo, Ajay Nagesh, Rebecca Sharp, and Mihai Surdeanu. 2019. Semi-supervised teacher-student architecture for relation extraction. In *Workshop on Structured Prediction*, pages 29–37.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *HLT-NAACL*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2010. Distant supervision for relation extraction without labeled data. In *ACL*.

Nam Nguyen and Rich Caruana. 2008. Classification with partial labels. In *KDD*, pages 551–559.

Longhua Qian, Guodong Zhou, Fang Kong, and Qiaoming Zhu. 2009. Semi-supervised learning for semantic relation classification using stratified sampling strategy. In *EMNLP*.

Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. 2005. Semi-supervised self-training of object detection models.

Guy Rotman and Roi Reichart. 2019. Deep contextualized self-training for low resource dependency parsing. *TACL*.

Mrinmaya Sachan and Eric Xing. 2018. Self-training for jointly learning to ask and answer questions. In *NAACL*.

Henry Scudder. 1965. Probability of error of some adaptive pattern-recognition machines. *IEEE Transactions on Information Theory*.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In *ACL*.

George Stoica, Emmanouil Antonios Platanios, and Barnabás Póczos. 2021. Re-tacred: Addressing shortcomings of the tacred dataset. In *AAAI*.

Ming-Kun Xie and Sheng-Jun Huang. 2018. Partial multi-label learning. In *AAAI*.

Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. 2020. Self-training with noisy student improves imagenet classification. In *CVPR*.

Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. 2016. Improved relation classification by deep recurrent neural networks with data augmentation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1461–1470.

Yan Yan and Yuhong Guo. 2020. Partial label learning with batch label correction. In *AAAI*, volume 34, pages 6575–6582.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *ACL*.

Junjie Yu, Tong Zhu, Wenliang Chen, Wei Zhang, and Min Zhang. 2020. Improving relation extraction with relational paraphrase sentences. In *COLING*, pages 1687–1698, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiajun Zhang and Chengqing Zong. 2016. Exploiting source-side monolingual data in neural machine translation. In *EMNLP*.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. 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 3016–3025.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. 2017. Position-aware attention and supervised data improve slot filling. In *EMNLP*.

Guodong Zhou, Junhui Li, Longhua Qian, and Qiaoming Zhu. 2008. Semi-supervised learning for relation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-I*.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting of the association for computational linguistics*, pages 427–434.