Learning with Noise: Improving Distantly-Supervised Fine-grained Entity Typing via Automatic Relabeling

Haoyu Zhang[1,2*], **Dingkun Long**[2*], **Guangwei Xu**[2]
Muhua Zhu[2], **Pengjun Xie**[2], **Fei Huang**[2], **Ji Wang**[1†]
[1]State Key Laboratory of High Performance Computing, National University of Defense Technology
[2]Alibaba Group
{zhanghaoyu10, wj}@nudt.edu.cn, {zhumuhua}@gmail.com
{dingkun.ldk, kunka.xgw, chengchen.xpj, f.huang}@alibaba-inc.com

Abstract

Fine-grained entity typing (FET) is a fundamental task for various entity-leveraging applications. Although great success has been made, existing systems still have challenges in handling noisy samples in training data introduced by distant supervision method. To address these noises, previous studies either focus on processing the clean samples (i.e., have only one label) and noisy samples (i.e., have multiple labels) with different strategies or filtering the noisy labels based on the assumption that the distantly-supervised label set certainly contains the correct type label. In this paper, we propose a probabilistic automatic relabeling method which treats all training samples uniformly. Our method aims to estimate the pseudo-truth label distribution of each sample, and the pseudo-truth distribution will be treated as part of trainable parameters which are jointly updated during the training process. The proposed approach does not rely on any prerequisite or extra supervision, making it effective on real applications. Experiments on several benchmarks show that our method outperforms previous competitive approaches and indeed alleviates the noisy labeling problem.

1 Introduction

Fine-grained entity typing (FET) is a task which aims to find a proper fine-grained semantic type given an entity mention and its corresponding context text. Knowledge acquired through FET is informative and can benefit a wide range of natural language processing (NLP) applications, such as relation extraction [Liu *et al.*, 2014], knowledge expansion [Dong *et al.*, 2014], factoid question answering [Dong *et al.*, 2015], and entity linking [Onoe and Durrett, 2019a].

Due to the lack of manually annotated fine-grained labels, distant supervision [Mintz *et al.*, 2009] method is popularly adopted by recent FET systems. This method links the entity mention to an entity in the knowledge base and annotates all associated types as distantly-supervised labels. Although distant supervision is adequate to label data automatically, it suffers from the noisy labeling problem severely. As illustrated in Figure 1, the entity mention "**Amazon**" in two different sentences will be labeled with same entity type set, in which some types are inappropriate given the context. Obviously, direct distant supervision produces noisy training data, which will hurt the performance of the FET systems [Ren *et al.*, 2016a].

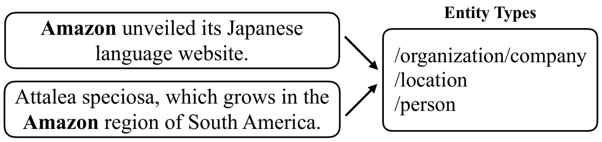
Amazon unveiled its Japanese language website.
Attalea speciosa, which grows in the **Amazon** region of South America.
Entity Types
/organization/company
/location
/person

Figure 1: Noisy samples produced by distant supervision.

To address the issue of noisy labeling, most of previous studies try to model the samples with only one label (treated as "clean" samples) and samples with multiple labels (treated as "noisy" samples) separately to improve the prediction performance [Ren *et al.*, 2016a; Abhishek *et al.*, 2017; Xu and Barbosa, 2018; Chen *et al.*, 2019]. In fact, we find that samples with only one distant label are not absolutely correct. [Wu *et al.*, 2019] proposed to detect and weight out noise based on the assumption that the distantly-supervised label set must contain the correct type for each training sample, which is overly strong. [Onoe and Durrett, 2019b] proposed to refine the distantly-labeled data with a learned model trained on human-annotated data. Consequently, previous studies suffer from two limitations: 1) Still rely on some prerequisites (e.g. human-curated clean dataset), making them inefficient in real applications; 2) Incapable of eliminating the impact of samples with only one type label but are false-positive (Table 4 shows more details).

In this paper, to handle the above two limitations simultaneously, we propose a probabilistic automatic relabeling method. As the ground-truth label distribution is not available, our method aims at estimating the pseudo-truth label distribution during the training process. In detail, each sample is assigned a continuous label distribution $\tilde{p}$ over all candidate labels, and $\tilde{p}$ will be jointly updated as trainable parameters through the back-propagation algorithm. The learning

[*] Indicates equal contribution.

[†] Corresponding author.

purpose is minimizing the Kullback-Leibler (KL) divergence between the predicted distribution and the pseudo-truth label distribution. Finally, we take the label with the highest value in $\tilde{p}$ as the only one pseudo-truth label. In order to ensure the rationality of the final estimated pseudo-truth distribution, we integrate the golden-noisy information during the training process with two specific designed constraints. In this way, our method can effectively **relabel** the noisy distant samples during training, thus improving the predictive performance.

To show the effectiveness and robustness of our approach, we conduct experiments on three benchmarks. Experimental results show that our approach achieves state-of-the-art results, significantly outperforms previous methods. Furthermore, we design additional subsidiary experiments to demonstrate that the above-mentioned problems are alleviated on FET tasks trained with noisy data.

2 Our Approach

The overall architecture of our proposed model is illustrated in Figure 2. Concretely, our model consists of a feature encoder (which has the same structure and objective function with **NFETC** model proposed by [Xu and Barbosa, 2018]) and a Probabilistic **A**utomatic **R**elabeling module, along with a three-phase training strategy.

2.1 Problem Definition

Given a training corpus $\mathcal{D}$ labeled with entity type hierarchy of knowledge base by distant supervision, we define $\Gamma = \{t_1, t_2, \cdots, t_{|T|}\}$ as all candidate type labels, where $|T|$ is the total number of types. Each type label t_i is a path from root node to the terminal node (e.g. *artist* represents */person/artist*), and a terminal node could be either a leaf node or a non-leaf node. In this paper, we use the terminal types as the predicted targets following settings in previous studies [Xu and Barbosa, 2018; Chen *et al.*, 2019]. In most part of this paper, we use type to refer to terminal type for simplicity.

Training corpus D constructed by distant supervision consists of triplets with form $(m_i, c_i, y_i), i = 1, 2, \cdots, N$, where $y_i \in \mathbb{R}^{|T|}$ is the label vector (also denoted as noisy label as it may contain types which are not appropriate). For each training sample, we denote the context sentence as a word sequence $c_i = \{w_1, w_2, \cdots, w_n\}$, and entity mention $m_i = \{w_{m_1}, w_{m_2}, \cdots, w_{m_l}\}$ as a continuous sub-sequence from the context sentence. Given the mention-context input pair, the FET task aims at predicting *the most appropriate* type y_i^* from the pre-defined candidate set Γ.

2.2 Feature Encoder

For fair comparison, we adopt the feature encoder used in [Xu and Barbosa, 2018]. For each sample triple $(m_i, c_i, y_i) \in \mathcal{D}$, each word w_j in c_i is first mapped into word embedding $e_j^w \in \mathbb{R}^{d_w}$ with a word embedding matrix $W \in \mathbb{R}^{d_w \times |V|}$, where $|V|$ is the vocabulary size and d_w is the embedding size. Analogously, word position embedding [Zeng *et al.*, 2014] $e_j^p \in \mathbb{R}^{d_p}$ is incorporated to reflect relative distances between each word and the target mention. Thus, the final embedding of the j-th word can be represented as concatenation of the two parts $e_j = [e_j^w, e_j^p]$.

Context Representation. The Bidirectional LSTM (Bi-LSTM) [Hochreiter and Schmidhuber, 1997; Graves and Schmidhuber, 2005] is then applied to model contextualized representations of context c_i. The sequence of embedding $\{e_1, e_2, ..., e_j, ..., e_n\}$ will be fed into the Bi-LSTM network. By combining the last hidden state of forward and backward pass, we get the contextualized representation of word w_j as $h_j = [\overrightarrow{h_j} \oplus \overleftarrow{h_j}]$, where $\oplus$ represents the element-wise sum operation. The hidden state size of Bi-LSTM notes as d_s. Following [Zhou *et al.*, 2016], a word-level attention module is adopted to attend to the most influenced words, and generate the final context representation r_{c_i}.

Mention Representation. The mention encoder contains two parts. Average encoder: given $\{e_{m_1}, ...e_{m_k}, ..., e_{m_l}\}$ be the sequence word embeddings of m_i, the average encoder simply averages value of each word embedding vector: $r_a = \frac{1}{l}\sum_{k=1}^{l} e_{m_k}$. LSTM encoder: by applying an LSTM over the extended mention embeddings, the context-aware representation $\{h_{m_1-1}, ...h_{m_k}, ..., h_{m_l+1}\}$ is achieved. The last hidden state $r_l = h_{m_l+1} \in \mathbb{R}^{d_s}$ is concatenated with r_a to from the final mention representation $r_{m_i} = [r_a, r_l]$.

2.3 Cross-Entropy Objective Function

As illustrated by left dash box of Figure 2, the feature vector is represented using $r_i = [r_{m_i}, r_{c_i}]$ and fed into a MLP layer. Then we use a softmax classifier to predict $\bar{y}_i$ over the candidate set Γ, where $W \in \mathbb{R}^{d_r \times |T|}$ and $b \in \mathbb{R}^{|T|}$ are trainable parameters. The cross-entropy loss function is denoted as Eq. 2, where θ denotes for trainable parameters of entire model.

$$p(y_i|m_i, c_i) = \text{softmax}(Wr_i + b) \tag{1}$$

$$\mathcal{L}_{ce}(\theta) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log\left(p(y_i|m_i, c_i; \theta)\right) \tag{2}$$

Clearly, optimizing the FET model with standard cross-entropy loss function has limitations for learning with noisy labels. The one-hot labels y for training samples with multiple entity types are not reasonable distribution, which will lead the model over-fitting to noisy labels and thus degrade the model performance. In the next section, we propose a probabilistic automatic relabeling method which helps to estimate the pseudo-truth label of each training sample and then address the noisy labeling problem.

2.4 Automatic Relabeling Module

In this section, we propose the probabilistic Automatic Relabeling (AR) module, an approach that strikes a balance between sufficient learning and robustness to noisy labels.

Our method aims to investigate the underlying truth label for each sample. The basic assumption of our idea is: for each mention-context pair (m, c), there exists only one most appropriate type given the candidate set Γ. Hence, the de-noising process over training set $\mathcal{D}$ implies that we seek

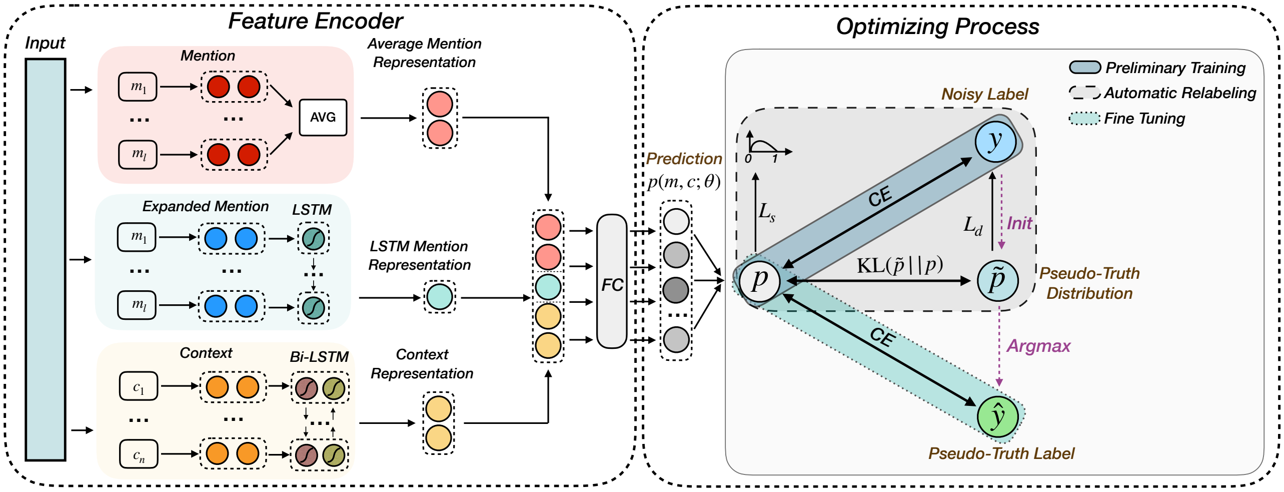

Figure 2: An overall illustration of our proposed **NFETC**-**A**utomatic-**R**elabeling (**NFETC-AR**) model.

to find the pseudo-truth label distribution $\tilde{p}$ over Γ of each sample. From the probabilistic perspective, how can we estimate this distribution? We realize it by treating $\tilde{p}$ as part of trainable parameters and $\tilde{p}$ will be jointly updated during the learning process. Therefore, when training with cross-entropy loss without automatic relabeling, the optimization problem of FET task can be defined as:

$$\theta^* = \mathrm{argmin}_{\theta}\mathcal{L}(m, c, y; \theta) \tag{3}$$

In our automatic relabeling module, the optimization objective is formulated as:

$$\theta^*, \tilde{p}^* = \mathrm{argmin}_{\theta,\tilde{p}}\mathcal{L}(m, c, y; \theta, \tilde{p}) \tag{4}$$

As shown in Figure 2, the proposed automatic relabeling module contains a pseudo-truth label distribution estimation term accompanied by two specifically designed constraints.

Pseudo-truth Label Distribution Estimation

The ground truth label distribution is not available, which must be estimated under proper assumptions. Although the training set is constructed through distant supervision method, the label noises are correlated and context-aware. This basic observation makes it possible to estimate the pseudo-truth label distribution $\tilde{p}$ using a self-supervised learning approach, which has also been verified by previous work [Wu *et al.*, 2019]. Concretely, we target on maximizing the information-theoretic dependency between the context and the pseudo-truth label associated. Although the original predicted distribution p is not completely correct, it is still a high-quality prior knowledge to guide the distribution estimation process of $\tilde{p}$. Inspired by [Hu *et al.*, 2017; Wang and Wu, 2019], we could directly estimate the pseudo-truth label distribution $\tilde{p}$ via minimizing the KL-divergence between $\tilde{p}$ and $p(m, c; \theta)$.

Formally, we assign a continuous label distribution $\tilde{p}_i$ to each training sample, where $\tilde{p}_i = \{\tilde{p}_{ij}: \tilde{p}_{ij} \in [0, 1], \sum_j \tilde{p}_{ij} = 1\}$. The KL-divergence loss function can be represented as:

$$\begin{aligned} \mathcal{L}_{kl} &= \frac{1}{N}\sum_{i=1}^{N}\mathrm{KL}(\tilde{p}_i \| p(y_i|m_i, c_i; \theta)) \\ &= \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|T|}\tilde{p}_{ij}\log\big(\frac{\tilde{p}_{ij}}{p_j(y_i|m_i, c_i; \theta)}\big), \end{aligned} \tag{5}$$

where θ denotes for all trainable parameters in the basic model. Additionally, $\tilde{p} \in \mathbb{R}^{N\times|T|}$ will be updated as other trainable parameters.

However, using the KL-divergence loss alone is not enough to get a well-estimated pseudo-truth label distribution. As we take the prediction p as objective, the $\mathcal{L}_{kl}$ can't deal with the wrong predictions produced by the basic typing model. These errors will continue to accumulate during the training process and then cause confirmation bias. Moreover, we empirically observe that the final estimated distribution $\tilde{p}$ is very smooth when employing only $\mathcal{L}_{kl}$, which is unhelpful to optimize it towards the ground-truth one-hot label distribution. Thus, we add two additional constraints to guide the pseudo-truth label distribution estimation process.

Distant label Constraint

As illustrated in Table 2, the majority of training data (64.46% of Wiki, 73.13% of OntoNotes and 75.92% of BBN) constructed by distant supervision contains only one type label, which is treated as clean sample in previous studies [Xu and Barbosa, 2018; Chen *et al.*, 2019]. The distantly-supervised labels still contain valuable information about the ground truth label distribution, although not all training samples with only one label are absolutely correct. By using this information, we can constrain the distribution estimation process and reduce the learning difficulty.

Considering all above, we make use of the distant noisy labels y with three operations: 1) At the beginning of the automatic relabeling process, we initialize the trainable distribution $\tilde{p}$ with the normalized version of original noisy labels y, i.e., $\tilde{p} = \mathrm{softmax}(y)$. 2) We also maintain the cross-entropy loss $\mathcal{L}_{ce}$ of the basic model as a part of the final optimization objective to ensure the rationality of the predicted distribution $p(m, c; \theta)$. 3) To avoid the estimated pseudo-truth label distribution $\tilde{p}$ from being totally different from the original noisy label distribution y, we add a cross-entropy loss between $\tilde{p}$ and y, which can be denoted as:

$$\mathcal{L}_d = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|T|}\tilde{p}_{ij}\log\, y_{ij} \tag{6}$$

Distribution Sharpen Constraint

As mentioned above, the FET task assumes that each sample has only one most appropriate true type label [Wu *et al.*, 2019]; thus we want to keep the distribution $\tilde{p}$ obeying the same property. We try to control the pseudo distribution by encouraging the predicted label distribution to be sharpen for each sample. Under this purpose, another regularization term is introduced: the distribution sharpen constraint, which pushes the predicted probability of each type to nearly 0 or 1, as shown in Eq. 7.

$$\mathcal{L}_s = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{|T|}p_j(y_i|m_i, c_i; \theta)\log\, p_j(y_i|m_i, c_i; \theta) \tag{7}$$

2.5 Training Strategy

To this end, we can train the AR module in an end-to-end manner with the standard back-propagation algorithm. The overall loss function can be formulated as:

$$\mathcal{L}_{ar} = \beta \cdot \mathcal{L}_{ce} + \gamma \cdot \mathcal{L}_{kl} + \omega \cdot \mathcal{L}_d + \delta \cdot \mathcal{L}_s, \tag{8}$$

where β, γ, ω and δ are hyper-parameters.

We divide the entire optimization process into three phases: 1) **Preliminary training**. Since the proposed approach does not rely on any extra supervision, at the beginning of the entire training process, we first train a basic entity typing model with parameters θ only using the original noisy labels (can be treated as warm-up training). 2) **Probabilistic Automatic Relabeling**. After a preliminary network is trained, we can step into the second phase and jointly estimate the pseudo label distribution $\tilde{p}$. 3) **Fine tuning**. Finally, we normalize the estimated distribution $\tilde{p}$ to one-hot labels $\hat{y}$ by choosing one label with maximum value for each sample, and then the pseudo labels $\hat{y}$ are used for continuous fine-tuning the basic typing model. More details about the three-phase optimization process are described in Algorithm 1. The training epoch number for each phase e_1, e_2, e_3 are hyper-parameters and chosen by grid search.

3 Experiments

We thoroughly evaluate the performance of our method on three benchmarks using several comparable approaches as baselines. Furthermore, we conduct several auxiliary experiments to analyze the effectiveness of our proposed method.

3.1 Settings

Datasets. We conduct the experiments on three standard and publicly available datasets. **Wiki** [Ling and Weld, 2012], **OntoNotes** [Weischedel *et al.*, 2013] and **BBN** [Weischedel and Brunstein, 2005]. We follow the same hierarchy refinement pre-processing of datasets used in [Abhishek *et al.*, 2017; Xu and Barbosa, 2018]. The detailed statistics of the datasets are shown in Table 2.

Algorithm 1: Training Procedure

```
Input:
- $\mathcal{D} = \{(m_i, c_i, y_i), \cdots, \}$)
- $\mathcal{X}_b \leftarrow Batch(\mathcal{D})$
- $\mathcal{Y}_b \leftarrow Batch(\mathcal{D})$
Parameters:
- Model parameters: $\theta$, Pseudo distribution: $\tilde{p}$
Randomly initialize model parameters $\theta$
for $epoch \leftarrow 1$ to $e_1$ do
    Update $\theta$ w.r.t. $\mathcal{L}_{ce}$ $(\mathcal{X}_b, \mathcal{Y}_b; \theta)$
Initialize $\tilde{p} \leftarrow softmax(y)$
for $epoch \leftarrow 1$ to $e_2$ do
    Update $\theta, \tilde{p}$ w.r.t. $\mathcal{L}_{ar}$ $(\mathcal{X}_b, \mathcal{Y}_b; \theta, \tilde{p})$
$\hat{y} \leftarrow argmax(\tilde{p})$
$\hat{\mathcal{Y}}_b \leftarrow Batch(\hat{y})$
for $epoch \leftarrow 1$ to $e_3$ do
    Update $\theta$ w.r.t. $\mathcal{L}_{ce}$ $(\mathcal{X}_b, \hat{\mathcal{Y}}_b; \theta)$
```

Implementation Details. Table 3 shows the hyper-parameters used in our method. The model is trained using mini-batched back-propagation, and Adam optimizer [Kingma and Ba, 2014] is used for optimization.

Evaluation Metrics. We adopt three commonly used evaluation metrics: Strict Accuracy (Strict Acc), Micro-averaged F1 (Micro-F1) score and Macro-averaged F1 (Macro-F1) [Ling and Weld, 2012]. On all datasets, we use the same training/development/test sets settings with previous studies, and the development sets are used to select the model with the best performance among all epochs.

Baselines. We compare our method with several state-of-the-art FET systems, including **AFET** [Ren *et al.*, 2016a], **AAA** [Abhishek *et al.*, 2017], **Attentive** [Shimaoka *et al.*, 2016], **NDP** [Wu *et al.*, 2019], **NFETC** [Xu and Barbosa, 2018] and **NFETC-CLSC** [Chen *et al.*, 2019]. [Xu and Barbosa, 2018] propose to model the type hierarchy with the hierarchical loss, which has been proven effective for FET task. Thus, for all **NFETC** based methods, we report results produced by $\mathbf{NFETC}$ and $\mathbf{NFETC}_{hier}$ respectively.

3.2 Overall Results

Table 1 presents the final results of our method with three metrics. The scores of each metric are calculated by running the model for five times and computing the mean and standard deviation values. We highlight the best performances of each metric in bold. We find that our approach achieves the best performance among all the baselines.

Our model improves the performances by a large margin compared to the basic model (NFETC) on all three datasets. Specifically, the auto-relabeling enhanced model improves the strict accuracy on the three datasets from 68.9 to 70.1,

Model	**Wiki**			**OntoNotes**			**BBN**		
	Strict Acc	**Macro F1**	**Micro F1**	**Strict Acc**	**Macro F1**	**Micro F1**	**Strict Acc**	**Macro F1**	**Micro F1**
AFET	53.3	69.3	66.4	55.3	71.2	64.6	68.3	74.4	74.7
AAA	65.8	81.2	77.4	52.2	68.5	63.3	65.5	73.6	75.2
Attentive	59.7	80.0	75.4	51.7	71.0	64.91	48.4	73.2	72.4
NDP	67.7	81.8	78.0	58.0	71.2	64.8	72.7	76.4	77.7
NFETC	56.2 ± 1.0	77.2 ± 0.9	74.3 ± 1.1	54.8 ± 0.4	71.8 ± 0.4	65.0 ± 0.4	73.8 ± 0.6	78.4 ± 0.6	78.9 ± 0.6
NFETC$_{hier}$	**68.9 ± 0.6**	**81.9 ± 0.7**	**79.0 ± 0.7**	60.2 ± 0.2	76.4 ± 0.1	70.2 ± 0.2	**73.9 ± 1.2**	78.8 ± 1.2	**79.4 ± 1.1**
NFETC-CLSC	-	-	-	59.6 ± 0.3	75.5 ± 0.4	69.3 ± 0.4	**74.7 ± 0.3**	**80.7 ± 0.2**	**80.5 ± 0.2**
NFETC-CLSC$_{hier}$	-	-	-	**62.8 ± 0.3**	**77.8 ± 0.3**	**72.0 ± 0.4**	73.0 ± 0.3	79.8 ± 0.4	79.5 ± 0.3
NFETC-AR	58.1 ± 1.1	79.0 ± 0.4	76.1 ± 0.4	62.8 ± 0.4	77.8 ± 0.4	71.8 ± 0.5	**76.7 ± 0.2**	**81.4 ± 0.3**	**81.5 ± 0.3**
NFETC-AR$_{hier}$	**70.1 ± 0.9**	**83.2 ± 0.7**	**80.1 ± 0.6**	**64.0 ± 0.3**	**78.8 ± 0.3**	**73.0 ± 0.3**	74.9 ± 0.6	80.4 ± 0.6	80.3 ± 0.6

Table 1: Performance results on three benchmark datasets.

	Wiki	**OntoNotes**	**BBN**
types	113	89	47
hierarchy depth	2	3	2
mentions-train	2009898	253241	86078
mentions-test	563	8963	12845
one label train data (%)	64.46	73.13	75.92
one label test data (%)	88.28	94.00	100

Table 2: Statistics of datasets.

Hyper-parameters	**Wiki**	**OntoNotes**	**BBN**
Learning rate	0.0002	0.0006	0.0007
Batch size	512	512	512
LSTM layer	0	2	1
hidden size (d_s)	-	700	560
Word emb size (d_w)	300	300	300
Pos emb size (d_p)	85	70	20
Epochs (e_1, e_2, e_3)	(5,5,10)	(10,10,10)	(5,5,20)
β	0.8	0.8	0.8
γ	0.3	0.3	0.1
ω	0.3	0.3	0.4
δ	0.1	0.1	0.4

Table 3: Hyper-parameters chosen for the three datasets.

60.2 to 64.0 and 73.9 to 76.7, respectively. This observation illustrates the necessity of noise reduction in the distantly-supervised FET task and the effectiveness of our proposed auto-relabeling method.

Comparing our NFETC-AR method with previous methods which process "clean" and "noise" data separately(e.g. NFETC-CLSC), NFETC-AR performs much better. The reason might be that the separate processing method can reduce the impact of noisy samples to some degree, but it also neglects the valuable information contained in noisy samples. These results further verify the intuition for relabeling the noisy samples and converting them into helpful training data. Also, there is an obvious margin between NFETC-AR and NDP. This result demonstrates that our model is more effective in finding correct type from the distantly-supervised type set, and the ability to handle false-positive samples is quite effective in improving the predictive performance.

3.3 Discussion and Analysis

Learning Curve of NFETC-AR. To better analyze the advantages of our model in tackling the noisy labeling problem, we compute the strict accuracy of our model on development set and test set after each training epoch, as shown in Figure 3. Note that the horizontal dotted lines in Figure 3 represent test set performances of NFETC and NFETC-CLSC respectively. We can see that: 1) In phase 1, our model is actually the basic NFETC; thus, the performance of our model cannot exceed the NFETC dotted lines by a meaningful margin. 2) In phase 2, as the loss function of the automatic relabeling module is adopted, the NFETC-AR model is able to reduce the negative effect of noisy data. As a result, the strict accuracy on the test set is further improved and then exceeds the NFETC model significantly. 3) In the third phase, the estimated pseudo-label distribution (soft-labels) in phase 2 is transformed into one-hot labels (hard-labels). Then we use these hard-labels (after relabeling) to continue fine-tuning the basic model. As a result, the performance of our model keeps improving and then surpasses the previous state-of-the-art model NFETC-CLSC.

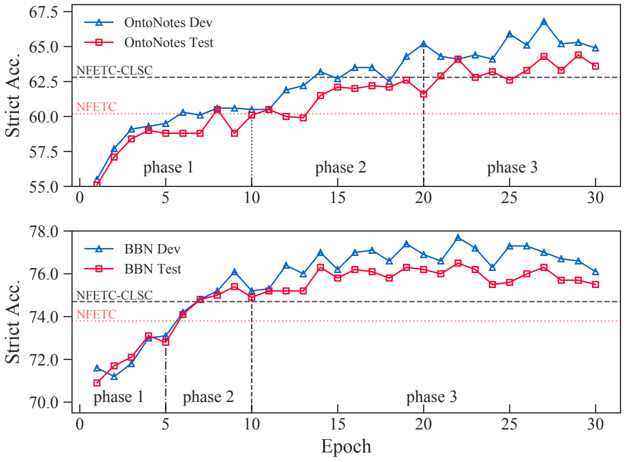

Figure 3: Strict Acc evolution on OntoNotes and BBN. We report the best performances of model variants (with or without the hierarchical loss) and hide the suffix $_{hier}$ in the plot for brevity.

Ablation Study of AR. We further evaluate the influence of each component in the automatic relabeling (AR) module by conducting an ablation study. From Table 5, we observe that each component statistically improves the model performance. Specifically, the KL-divergence loss is the key component of the AR process. Without $\mathcal{L}_{kl}$, the model produces similar performance with the basic model (shown in w/o AR

Source	Type	Context & Mention	Original Label	After Relabeling
OntoNotes	Multi-to-one-in	Jennifer Laden , **NPR News** , Jerusalem	{/org/company/news, /person}	/org/company/news
Wiki	Multi-to-one-in	the Plains of Abraham in Quebec City , Quebec , **Canada** .	{/location/country, /person/director}	/location/country
BBN	Multi-to-one-in	**Federal researchers** said lung-cancer mortality rates for people under 45	{/person, /org/government}	/person
OntoNotes	Multi-to-one-out	Because along with Haditha comes **Jesse Macbeth** , allegedly ...	{/other/art/writing, /other/art/stage}	/person
OntoNotes	One-to-one-out	... Alcee Hastings of Florida of eight **impeachment articles** ,...	/other/health/malady	/other/art/writing

Table 4: Cases of the automatic relabeling results on the three datasets.

	Acc	Macro F1	Micro F1
NFETC-AR$_{hier}$	**64.0 ± 0.3**	**78.8 ± 0.3**	**73.0 ± 0.3**
w/o $\mathcal{L}_{kl}$	61.2 ± 0.5	76.6 ± 0.4	70.4 ± 0.5
w/o $\mathcal{L}_{d}$	63.8 ± 0.3	78.4 ± 0.2	72.6 ± 0.3
w/o $\mathcal{L}_{ce}$	61.1 ± 0.3	76.1 ± 0.3	69.9 ± 0.5
w/o noisy label init	55.0 ± 0.3	67.1 ± 0.4	60.3 ± 0.4
w/o $\mathcal{L}_{s}$	63.7 ± 0.6	78.3 ± 0.4	72.3 ± 0.6
w/o AR	60.2 ± 0.2	76.4 ± 0.1	70.2 ± 0.2

Table 5: Ablation study on OntoNotes dataset.

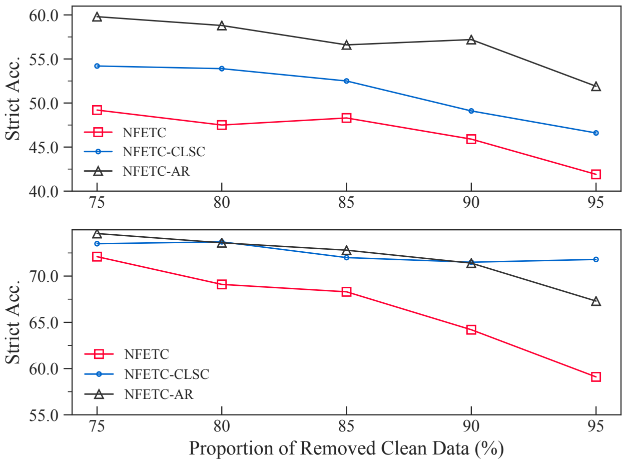

Figure 4: Strict Acc on OntoNotes (upper) and BBN (lower) when different proportions of “clean” training data are removed.

line) on Marco-F1 and Micro-F1. Moreover, among the three operations in the distant label constraint, we find that noisy label initialization has the greatest impact (switching to random initialization will degrade the accuracy to 55.0). This result again proves that the original noisy information is valuable to some extent, and fully making use of this information is very important to the automatic relabeling process.

Statistical Analysis of AR. In order to examine the relabeling results of our proposed model, we statistically analyze the differences between the noisy labels y and the pseudo labels $\hat{y}$ after automatic relabeling. We find that there are three types of modification. **Multi-to-one-in**: samples with multiple distantly-supervised labels are modified into one of them; In contrast, **Multi-to-one-out** represents samples with multiple labels are finally modified into one label out of the original ones. Besides, we find that a part of the samples with only one label has also been modified, and it is noted as **One-to-one-out**. Taking OntoNotes dataset as an example, **27.52%** of the training samples are modified. Among the modified data, the percentages of the three different types are **97.49%**, **0.12%** and **2.39%** respectively. Similar results are obtained on both Wiki and BBN datasets. After careful inspection of the corrected labels $\hat{y}$, we find: 1) not all samples with only one type label are correctly labeled; 2) the assumption that the ground truth label must exist in the distantly-supervised labels is also biased. In Table 4, we select some samples of each correction type for better understanding. The result of statistical analysis and case studies illustrates the ability of our model in automatic relabeling. Moreover, our proposed method does not rely on any pre-set assumptions that are actually problematic.

Robustness to Noisy Data. To study the robustness of our model in handling noisy data, we compare our method with previous start-of-the-art systems after removing “clean” samples on the training set. For each dataset, we split the training set into “clean” and “noisy” samples, and “clean” samples are removed by different proportions (75% - 95%). Here, we treat samples with one type label as **clean data** following [Chen *et al.*, 2019], as most of them contain correct information. We compute the strict accuracy given remaining clean samples and all noisy samples to evaluate the model's robustness to more noisy data. According to Figure 4, our model surpasses all baselines consistently on the OntoNotes dataset. On the BBN dataset, our model achieves comparable performances with NFETC-CLSC in most cases except when removing 95% clean data. The reason might be that NFETC-CLSC applies an unsupervised cluster-based noise reduction method and thus is more robust with extremely limited clean data. Overall, the experiment proves that our proposed approach makes the basic model much more robust.

4 Related Work

Fine-grained entity typing (FET) is a common task in NLP. In general, a large amount of labeled data is required to train a FET model, which is quite expensive. To address this issue, [Ling and Weld, 2012] first proposed building the training data via distant supervision [Mintz *et al.*, 2009]. Embedding-based models were applied to FET since then [Yogatama *et al.*, 2015; Dong *et al.*, 2015]. Recently, deep neural methods have been widely applied to FET. [Shimaoka *et al.*, 2016] proposed to use LSTMs to encode the context information and then use attention mechanism to select informative information. [Ma *et al.*, 2016; Lin and Ji, 2019] exploited to improve the model performance by leveraging the entity type information. [Xin *et al.*, 2018] proposed a knowledge attention model by jointly considering the information from knowledge bases (KBs).

Despite the success of distance supervision, it still suffers from the noisy labeling problem. To alleviate this issue, [Gillick *et al.*, 2014] refined the training data by applying a set of heuristics to prune types, [Ren *et al.*, 2016a; Ren *et al.*, 2016b] proposed the AFET model which separates the loss function for clean and noisy entity separately, and incorporates the imperfect annotation by partial-label loss. Additionally, [Abhishek *et al.*, 2017; Xu and Barbosa, 2018] proposed two variants of partial-label loss. Nevertheless, these methods still struggle with confirmation bias. Based on the assumption that the distantly-supervised label set must contain the ground-truth label, [Wu *et al.*, 2019] proposed to model the structured, noisy labels directly and then weight out noisy labels during training with the help of random walking process. [Chen *et al.*, 2019] leveraged the noisy samples as regularization via compact latent space clustering method. [Onoe and Durrett, 2019b] tried to filter and relabel the noisy samples with the help of additional human-labeled data. Unlike these methods, we propose a unified framework that treats the clean and noisy samples equally and solves the noise labeling problem without extra supervision.

5 Conclusion and Future Work

In this paper, we propose a probabilistic automatic relabeling method for fine-grained entity typing. The proposed approach is able to infer the pseudo label of each sample and use the relabelled samples to fine-tune the backbone network. The proposed approach does not rely on any prerequisite or extra supervision, making it robust and effective on real applications. Extensive experimental results show that the proposed method performs better than previous approaches, and can certainly alleviate the noisy labeling problem in fine-grained entity typing task. Our proposed method is independent of the backbone network; thus, in future work we plan to investigate the performance of our method under different backbone networks. In addition, the automatic relabeling approach can also be applied to other NLP tasks established on distantly-supervised datasets, such as relation extraction, lexicon-based named entity recognition. More research is underway to explore these two directions.

Acknowledgments

We thank all the anonymous reviewers for their valuable suggestions. This research is partially supported by the National key R&D program of China (No. 2017YFB1001802) and the National Nature Science Foundation of China (No. 61532007).

References

[Abhishek *et al.*, 2017] Abhishek, Ashish Anand, and Amit Awekar. Fine-grained entity type classification by jointly learning representations and label embeddings. In *EACL*, pages 797–807, 2017.

[Chen *et al.*, 2019] Bo Chen, Xiaotao Gu, Yufeng Hu, Siliang Tang, Guoping Hu, Yueting Zhuang, and Xiang Ren. Improving distantly-supervised entity typing with compact latent space clustering. In *NAACL-HLT*, pages 2862–2872, 2019.

[Dong *et al.*, 2014] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *KDD*, pages 601–610, 2014.

[Dong *et al.*, 2015] Li Dong, Furu Wei, Hong Sun, Ming Zhou, and Ke Xu. A hybrid neural model for type classification of entity mentions. In *IJCAI*, pages 1243–1249, 2015.

[Gillick *et al.*, 2014] Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. Context-dependent fine-grained entity type tagging. *arXiv preprint arXiv:1412.1820*, 2014.

[Graves and Schmidhuber, 2005] Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

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

[Hu *et al.*, 2017] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *ICML*, pages 1558–1567, 2017.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[Lin and Ji, 2019] Ying Lin and Heng Ji. An attentive fine-grained entity typing model with latent type representation. In *EMNLP-IJCNLP*, pages 6196–6201, 2019.

[Ling and Weld, 2012] Xiao Ling and Daniel S. Weld. Fine-grained entity recognition. In *AAAI*, 2012.

[Liu *et al.*, 2014] Yang Liu, Kang Liu, Liheng Xu, and Jun Zhao. Exploring fine-grained entity type constraints for distantly supervised relation extraction. In *COLING*, pages 2107–2116, 2014.

[Ma *et al.*, 2016] Yukun Ma, Erik Cambria, and Sa Gao. Label embedding for zero-shot fine-grained named entity typing. In *COLING*, pages 171–180, 2016.

[Mintz *et al.*, 2009] Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP*, pages 1003–1011, 2009.

[Onoe and Durrett, 2019a] Yasumasa Onoe and Greg Durrett. Fine-grained entity typing for domain independent entity linking. *arXiv preprint arXiv:1909.05780*, 2019.

[Onoe and Durrett, 2019b] Yasumasa Onoe and Greg Durrett. Learning to denoise distantly-labeled data for entity typing. In *NAACL-HLT*, pages 2407–2417, 2019.

[Ren *et al.*, 2016a] Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. AFET: automatic fine-grained entity typing by hierarchical partial-label embedding. In *EMNLP*, pages 1369–1378, 2016.

[Ren *et al.*, 2016b] Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. Label noise reduction in entity typing by heterogeneous partial-label embedding. In *KDD*, pages 1825–1834, 2016.

[Shimaoka *et al.*, 2016] Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. An attentive neural architecture for fine-grained entity type classification. In *AKBC@NAACL-HLT*, pages 69–74, 2016.

[Wang and Wu, 2019] Guo-Hua Wang and Jianxin Wu. Repetitive reprediction deep decipher for semi-supervised learning. *arXiv preprint arXiv:1908.04345*, 2019.

[Weischedel and Brunstein, 2005] Ralph Weischedel and Ada Brunstein. Bbn pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia*, 112, 2005.

[Weischedel *et al.*, 2013] Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. Ontonotes release 5.0 ldc2013t19. *Linguistic Data Consortium, Philadelphia, PA*, 23, 2013.

[Wu *et al.*, 2019] Junshuang Wu, Richong Zhang, Yongyi Mao, Hongyu Guo, and Jinpeng Huai. Modeling noisy hierarchical types in fine-grained entity typing: A content-based weighting approach. In *IJCAI*, pages 5264–5270, 2019.

[Xin *et al.*, 2018] Ji Xin, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Improving neural fine-grained entity typing with knowledge attention. In *AAAI*, 2018.

[Xu and Barbosa, 2018] Peng Xu and Denilson Barbosa. Neural fine-grained entity type classification with hierarchy-aware loss. In *NAACL-HLT*, pages 16–25, 2018.

[Yogatama *et al.*, 2015] Dani Yogatama, Daniel Gillick, and Nevena Lazic. Embedding methods for fine grained entity type classification. In *ACL*, pages 291–296, 2015.

[Zeng *et al.*, 2014] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344, 2014.

[Zhou *et al.*, 2016] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *ACL*, 2016.