Automatic Noisy Label Correction for Fine-Grained Entity Typing

Weiran Pan[1,2] , **Wei Wei**[1,2*] , **Feida Zhu**[3]
[1]Cognitive Computing and Intelligent Information Processing (CCIIP) Laboratory, School of Computer Science and Technology, Huazhong University of Science and Technology, China
[2]Joint Laboratory of HUST and Pingan Property & Casualty Research (HPL), China
[3]School of Computing and Information Systems, Singapore Management University, Singapore
{panwr, weiw}@hust.edu.cn, fdzhu@smu.edu.sg

Abstract

Fine-grained entity typing (FET) aims to assign proper semantic types to entity mentions according to their context, which is a fundamental task in various entity-leveraging applications. Current FET systems usually establish on large-scale weakly-supervised/distantly annotation data, which may contain abundant noise and thus severely hinder the performance of the FET task. Although previous studies have made great success in automatically identifying the noisy labels in FET, they usually rely on some auxiliary resources which may be unavailable in real-world applications (*e.g.,* pre-defined hierarchical type structures, human-annotated subsets). In this paper, we propose a novel approach to automatically correct noisy labels for FET without external resources. Specifically, it first identifies the potentially noisy labels by estimating the posterior probability of a label being positive or negative according to the logits output by the model, and then relabel candidate noisy labels by training a robust model over the remaining clean labels. Experiments on two popular benchmarks prove the effectiveness of our method. Our source code can be obtained from https://github.com/CCIIPLab/DenoiseFET.

1 Introduction

Fine-grained entity typing (FET) [Ling and Weld, 2012] aims to predict fine-grained semantic types for entity mentions according to their contexts. Recently, [Choi *et al.*,] further proposed the ultra-fine entity typing introducing a richer type set. The fine-grained entity types obtained by FET is beneficial for many downstream NLP tasks like entity linking [Onoe and Durrett, a], relation extraction [Koch *et al.*, ; Shang *et al.*, ; Shang *et al.*, 2022], question answering [Wei *et al.*, a; Wei *et al.*, d], dialogue [Wei *et al.*, b; Wei *et al.*, c] and coreference resolution [Onoe and Durrett, b].

A fundamental challenge of FET is handling the noisy labels during training. Current FET systems usually generate training data via crowdsourcing or distant supervision, both of which may introduce noisy labels. With the large-scale and fine-grained type sets, it is extremely difficult for humans to annotate samples accurately. The context-agnostic distant supervision method also introduces noise inevitably. Figure 1 shows some noisy labels from the training sets of Ultra-fine and Ontonotes. Previous denoising work in FET usually relies on pre-defined hierarchical type structures or manually labeled subsets to identify noisy labels. However, these auxiliary resources may be unavailable in real-world applications, which limits the application of such methods.

Figure 1: Noisy label examples, the false-positive label is present in red, false-negative labels are present in grey. (a): A manually-annotated example selected from the Ultra-Fine dataset. The annotator failed to cover all types. (b): A distantly-labeled example chosen from the OntoNotes dataset. Context-agnostic distant supervision generated mislabeled sample.

To automatically correct noisy labels without the help of auxiliary resources, an intuitive way is to treat the potentially noisy labels as unlabeled data and train the FET model with the rest clean labels. Then we can relabel candidate noisy labels to generate a cleaner dataset. Intuitively, excluding the potentially noisy labels during training makes the model more robust, and learning from clean samples helps the model to make correct predictions on similar but mislabeled samples. The main challenge is to identify the potentially noisy labels precisely. This problem also receives widespread attention in robust learning from noisy labels. Recent studies reveal that deep neural networks (DNNs) are prone to first learn simple patterns before overfitting noisy labels [Arpit *et al.*,]. Based on this memorization effect, there are two main approaches to filter noisy labels. Methods like Co-teaching [Han *et al.*, 2018] treat the labels with the top k% loss as possible noisy

*Corresponding author.

labels (k% is the estimated noise rate). Some other methods like SELF [Nguyen *et al.*, 2020] record the model output on different training epochs and filter noisy labels progressively. Despite their success in other fields, those methods are difficult to directly apply to FET tasks for two reasons: i) Firstly, the noise rate can differ across FET types. Manually estimating the noise rate for each type is time-consuming. ii) Secondly, existing FET datasets usually provide a massive training set annotated by distant supervision. Current FET models can even converge before iterating through all data. Thus the multi-round methods are infeasible in practice.

To this end, we propose a novel approach to select the potentially noisy labels for FET. Figure 2 briefly demonstrates the selection process. We note the model does not strictly separate positive and negative labels before overfitting, which means the model is "ambiguous" on some labels in the early stage of training. We argue this phenomenon is partially caused by noisy labels. Due to the existence of noisy labels, samples belonging to the same pattern may have different annotations. This inconsistency between annotation and underlying patterns brings difficult for the model to separate samples according to their annotations. Therefore, we propose to select potentially noisy labels by identifying the "ambiguous labels". Specifically, we first fit the logits output by the model on positive and negative labels using two Gaussian distributions, respectively. Then, we calculate the posterior probability of a label being positive or negative. Finally, we select the potential noisy labels according to the agreement between original annotations and the calculated posterior probabilities.

In summary, our main contributions are three-fold:

- We propose a novel method to automatically correct noisy labels in the FET training corpus without the help of auxiliary resources.
- We propose an effective method to automatically identify the potentially noisy labels by discriminating the "ambiguous labels" from the normal ones.
- Experiments on two widely-used benchmarks prove the effectiveness of the proposed method.

2 Related Work

2.1 Fine-grained Entity Typing

The noisy labeling problem in FET domain has been studied for years. [Gillick *et al.*, 2014] filter noisy labels by a set of heuristics rules, [Ren *et al.*, a; Ren *et al.*, b] propose the partial label loss (PLL) and the partial-label embedding method to alleviate the negative influence caused by noisy labels, [Onoe and Durrett, c] utilize a small set of manually labeled training data to train a filter module and a relabel module then denoise the distantly-labeled data. [Chen *et al.*,] treat the noisy labels as unlabeled data and performs Compact Latent Space clustering (CLSC) to regularize their representation. [Zhang *et al.*,] assume each sample only has one most appropriate label and treat the underling ground truth labels as trainable parameters for fine-tuning during training. [Wu *et al.*,] estimate the noise confusion matrix through hierarchical label structure. Other works on the FET task dedicate to mining label dependencies [Xiong *et al.*, ; Liu *et al.*,] designing more expressive embedding method [Onoe *et al.*, 2021] and exploring better automated data annotation method [Dai *et al.*,] to improve performance.

Most existing denoising methods in FET rely on prior auxiliary resources (pre-defined hierarchical type structures/manually labeled subsets) or assume each sample only has one most appropriate label. As compared, in this paper we consider the more restrictive setting where no auxiliary resources are available and one sample can have multiple labels simultaneously.

2.2 Robust Learning from Noisy Labels

Previous studies have pointed out that DNNs are susceptible to noisy labels [Zhang *et al.*, 2017]. Sample selection is a popular method to counteract noise, which only uses relatively clean labels to train the model. Here we briefly note several previous works in this field.

Co-teaching [Han *et al.*, 2018] simultaneously train two collaborating networks to filter out potentially noisy labels according to their loss. This method needs to control select how many small-loss instances as clean ones, which can be tricky in practice. To automatically select clean labels, DivideMix [Li *et al.*, 2020] leveraged a one-dimensional and two-component Gaussian Mixture Model (GMM) to model the loss distribution of clean and noisy labels. SELF [Nguyen *et al.*, 2020] observed that the network's prediction is consistent on clean samples while oscillating strongly on mislabeled samples. Therefore, they select the clean labels according to the agreement between annotated labels and the running average of the network's prediction.

In practice, we found that the loss does not form a bimodal distribution in FET tasks and fitting the GMM can not distinguish clean and noisy labels precisely. The multi-round methods are also infeasible. Training on the large-scale distantly annotated training set, current FET models can converge or even overfit noisy labels in the first epoch. To this end, we propose a novel method to select potentially noisy labels by estimating the posterior probability of a label being positive or negative according to the logits output by the model.

3 Methodology

Indeed, our method corrects the noisy labels with two major steps, namely: (1) Selecting potentially noisy labels by identifying "ambiguous labels"; (2) Re-labeling training sets by learning a robust entity typing model over the clean labels. Next, we will first introduce the problem definition and the entity typing model used in this paper, and then present our noisy label correction method in detail.

3.1 Problem Definition

Entity typing datasets consist of a collection of (sentence, mention) tuples: $\mathcal{D} = \{(x_1, m_1), (x_2, m_2), .., (x_n, m_n)\}$. Given a sentence x and a marked entity mention m, the fine-grained entity typing task aims to predict a set of types $y \subset \mathcal{T}$ of m, where $\mathcal{T}$ is a set of pre-defined fine-grained semantic types (*e.g., actor, company, building*).

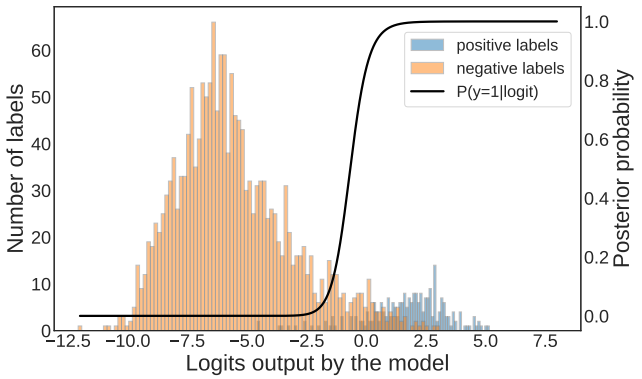


Figure 2: The logits output by the model on type "organization" when achieving early stopping on the Ultra-Fine dataset. Positive and negative labels are colored for illustrative purposes. This figure also shows the estimated posterior probability of a label being positive.

3.2 Entity Typing Model

In FET tasks, a widely-used paradigm is to fine-tune the pre-trianed language models (PLMs) for entity typing. In this paper we adopt the prompt learning model [Ding *et al.*, 2021] to tune PLMs for entity typing, as it is proved to be effective for fine-tuning PLMs to specific tasks. Specifically, we reformulate the original input (sentence, mention) into the prompt input via a crafted template as follows:

$$T(x, m) = x[\mathrm{P}_1]m[\mathrm{P}_2][\mathrm{P}_3][\mathrm{MASK}], \tag{1}$$

where $[\mathrm{P}_1], [\mathrm{P}_2], [\mathrm{P}_3]$ are additional trainable special tokens. Their embeddings are randomly initialized. Conventionally, prompt-based tuning models need to define a set of label words V_y for each $y \in \mathcal{T}$, and then the model can predict entity types by filling the [MASK]:

$$p(y \in \mathcal{T}|x, m) = p([\mathrm{MASK}] = w \in V_y|T(x, m)). \tag{2}$$

In this way, FET is converted to mask prediction, one of the pre-training tasks in Masked Language Models. However, it's difficult to choose a proper label word set for each type in FET. In practice, we directly modify the decoder in the Masked Language Modeling Head (MLM Head) and let it output the probability of samples belonging to each candidate type:

$$p(y \in \mathcal{T}|x, m) = \sigma(\mathbf{w}_y \mathrm{F}(\mathbf{h}_{[\mathrm{MASK}]}) + \mathbf{b}_y), \tag{3}$$

where $\mathbf{h}_{[\mathrm{MASK}]}$ is [MASK]'s contextualized representation produced by PLMs; F represent the Masked Language Modeling Head without decoder; $\mathbf{w}_y$ and $\mathbf{b}_y$ are trainable parameters in modified decoder. If type y is in PLMs' vocabulary, we initialize $\mathbf{w}_y$ and $\mathbf{b}_y$ according to y's weight and bias in MLM Head's decoder, respectively. If y consists of multi tokens, we take the average of their weight and bias in MLM Head's decoder to initialize $\mathbf{w}_y$ and $\mathbf{b}_y$.

3.3 Candidate Noisy Labels Selection

The basic idea of our method is to exclude potentially noisy labels from the supervision signals for robust training, then generate a cleaner dataset by relabeling the training set. In this section, we detail how to select potentially noisy labels.

According to the memorizatioin effect [Arpit *et al.*,], deep neural networks usually memorize simple patterns before overfitting noisy labels, which is also observed in our experiments. Figure 2 shows our model's output logits on type "organization" when achieving early stopping on the Ultra-Fine dataset. Some samples with different annotations have similar logits, and this phenomenon is universal across types. According to our observation, those *"ambiguous"* samples mainly consists of mislabeled samples. This is because the mislabeled samples belong to the same pattern as some correctly labeled samples in datasets. Due to the memorization effect, the model tends to output similar logits on samples with the same pattern before overfitting. As a result, the predictions of noisy labels may be "ambiguous" or even opposite to the annotation during the early training phase.

Therefore, we propose to pre-train the model with the original annotation until early stopping, then select potentially noisy labels by independently identifying those "ambiguous labels" for each type. Specifically, we first estimate the posterior probability of a label being positive or negative according to the logits: $p(y = 1|l)$. Then we can identify "ambiguous labels" through the agreement between $p(y = 1|l)$ and original annotations. During pre-training, we use the following standard binary cross-entropy (BCE) loss:

$$\mathcal{L}_1 = \frac{1}{|B|} \sum_{y_{s,t} \in \hat{Y}} \left[y_{s,t} \log p_{s,t} + (1 - y_{s,t}) \log(1 - p_{s,t}) \right], \tag{4}$$

where $y_{s,t} \in \{0, 1\}$ is the original annotation in datasets; $p_{s,t}$ is the model prediction, denotes the probability of sample s has type t; $|B|$ is the batch size. After achieving early stopping, we gather the logit output by the model and estimate the posterior probability $p(y = 1|l)$:

$$p(y = 1|l) = \frac{p(l|y = 1)p(y = 1)}{p(l|y = 1)p(y = 1) + p(l|y = 0)p(y = 0)}, \tag{5}$$

where y is the ground truth label and 0 means negative label, 1 means positive label; l is the logit output by the model. The main challenge is estimating the prior probability: $p(y = 0), p(y = 1)$ and the likelihood function: $p(l|y = 1), p(l|y = 0)$. We note the logits output by the model on positive (negative) labels form unimodal distribution (as shown in Figure 2). It's a natural idea to fit it using a one-dimensional Gaussian distribution:

$$G(l|\mu, \delta) = \frac{1}{\sqrt{2\pi}\delta} \exp^{-\frac{(l-\mu)^2}{2\delta^2}}. \tag{6}$$

However, original datasets only provide the noisy annotation $\hat{y}$ instead of the ground truth label y. Therefore, directly fit Gaussian distribution according to the original annotation only give $p(l|\hat{y} = 0)$ and $p(l|\hat{y} = 1)$. We recommend to remove those obvious noisy labels before estimating the likelihood function. Algorithm 1 provides a heuristic method to filter obvious noisy labels iteratively. This method first fits a Gaussian distribution on the model output logits then removes the outliers. This process will be repeated until no outliers can

Algorithm 1 Filter obvious noisy labels

Input: The logits output by the model: l; original noisy annotation in the dataset: $\hat{Y}$.
Parameter: Outlier threshold α.
Output: Relative clean positive (negative) label set Y_1 (Y_0)

Let $Y_1 = \hat{Y}_1 = \{y = 1 | y \in \hat{Y}\}$
Let $Y_0 = \hat{Y}_0 = \{y = 0 | y \in \hat{Y}\}$
repeat
  Let $\hat{Y}_1 = Y_1$ and $\hat{Y}_0 = Y_0$
  Calculate the average $\hat{\mu}_1$ ($\hat{\mu}_0$) and standard deviation $\hat{\delta}_1$ ($\hat{\delta}_0$) of the logits output by the model corresponding to the labels in $\hat{Y}_1$ ($\hat{Y}_0$)
  Let $Y_1 = \{l_y > \hat{\mu}_1 - \alpha\hat{\delta}_1 | y \in \hat{Y}_1\}$
  Let $Y_0 = \{l_y < \hat{\mu}_0 + \alpha\hat{\delta}_0 | y \in \hat{Y}_0\}$
until $|Y_1| == |\hat{Y}_1|$ and $|Y_0| == |\hat{Y}_0|$
return Y_1, Y_0

be found. After getting the relative clean positive (negative) label set $Y_1(Y_0)$, the prior probability and likelihood function can be estimated in the following way:

$$p(y = k) = \frac{|Y_k|}{|Y_0| + |Y_1|}, \tag{7}$$

$$p(l|y = k) = G(l|\mu_k, \delta_k), \tag{8}$$

$$\mu_k = \frac{1}{|Y_k|} \sum_{y \in Y_k} l_y; \quad \delta_k^2 = \frac{1}{|Y_k|} \sum_{y \in Y_k} (l_y - \mu_k)^2, \tag{9}$$

where $k = 1(0)$ denotes the positive (negative) labels. Finally, a label is selected as a potentially noisy label if it satisfies one of the following rules:

$$\begin{cases} \hat{y} = 1 \quad and \quad p(y = 1|l_{\hat{y}}) < (0.5 + \epsilon) \\ \hat{y} = 0 \quad and \quad p(y = 1|l_{\hat{y}}) > (0.5 - \epsilon) \end{cases}, \tag{10}$$

where $\epsilon > 0$ is a hyperparameter, and the larger value means more labels are selected as potentially noisy labels.

3.4 Noisy Label Correction

To correct the noisy labels in the dataset as many as possible, we propose to fine-tune the model using clean labels. Intuitively, training on clean samples helps the model to make correct predictions on those similar but mislabeled samples. For the selected potentially noisy labels, we treat them as unlabeled data and perform entropy regularization [Grandvalet *et al.*,] to reduce class overlap. Formally, the loss on a training batch during noisy label correction is as follows:

$$\begin{aligned} \mathcal{L}_2 = & \frac{1}{|B|} \sum_{y_{s,t} \in Y_c} [y_{s,t} \log p_{s,t} + (1 - y_{s,t}) \log(1 - p_{s,t})] \\ & - \frac{\beta}{|B|} \sum_{y_{s,t} \in Y_n} [p_{s,t} \log p_{s,t} + (1 - p_{s,t}) \log(1 - p_{s,t})], \end{aligned} \tag{11}$$

Algorithm 2 Noisy labels correction for FET

Input: Noisy labeled training set $\hat{\mathcal{D}}_{train}$; validation set $\mathcal{D}_{val}$.
Parameter: Threshold ϵ; weight of entropy regularization β; fine-tuning step k on clean samples.
Output: Cleaner training set $\mathcal{D}_{train}$.

Fine-tune model $\mathcal{M}$ on $\hat{\mathcal{D}}_{train}$ until the performance drops on $\mathcal{D}_{val}$
Filter obvious noisy labels by Algorithm 1
Select potentially noisy labels by Equation (10)
Conduct k step fine-tuning on model $\mathcal{M}$ using loss function in Equation (11)
Relabel $\hat{\mathcal{D}}_{train}$ according to Equation (12) to generate $\mathcal{D}_{train}$
return $\mathcal{D}_{train}$

where Y_c and Y_n stand for clean and potentially noisy label set respectively; β is a hyperparameter that balances the BCE loss and the entropy regularization. Finally, we relabel potentially noisy labels to generate cleaner datasets:

$$\begin{cases} y'_{s,t} = 1, & if\ p_{s,t} > 0.5 \\ y'_{s,t} = 0, & if\ p_{s,t} < 0.5 \end{cases} \quad for\ y_{s,t} \in Y_n, \tag{12}$$

where $y'_{s,t}$ is the new label after noisy label correction. Algorithm 2 summarizes the whole process of our method.

4 Experiments

4.1 Datasets and Experiment Setup

Datasets. We conduct experiments on two standard fine-grained entity typing datasets: Ultra-fine and OntoNotes. Followed [Liu *et al.*,] we use the 2k/2k/2k train/dev/test splits for Ultra-fine which contains 6K manually annotated samples, 2519 categories. Original OntoNotes dataset contains 25k/2k/9k train/dev/test data, 89 categories. [Choi *et al.*,] offers an augmented training set containing 0.8M samples. We conduct experiments on both versions. In terms of evaluation metrics, we follow the widely used setting proposed in [Ling and Weld, 2012]. On Ultra-Fine, we report macro-averaged precision, recall, and F1-score. On OntoNotes, we report accuracy, macro-averaged F1-score, and micro-averaged F1-score.

Baselines. For ultra-fine entity typing task, we compare with the following approaches:

- **UFET** [Choi *et al.*,] performs multi-label classification on features encode by GloVe+LSTM and a character level CNN.
- **LabelGCN** [Xiong *et al.*,] models the label dependency using a graph propagation layer.
- **LDET** [Onoe and Durrett, c] performs relabeling and sample filtering to denoise the datasets before training FET models.
- **MLMET** [Dai *et al.*,] automatically generates ultra-fine entity typing labels to train the entity typing model by combining hypernym extraction patterns with a masked language model.

Method	P	R	F1
UFET	47.1	24.2	32.0
LabelGCN	50.3	29.2	36.9
LDET	51.5	33.0	40.2
Box	52.8	38.8	44.8
LRN	54.5	38.9	45.4
MLMET	53.6	45.3	49.1
Ours (original)	58.2±0.8	40.8±0.1	48.0±0.2
Ours (denoised)	55.6±0.4	44.7±0.3	**49.5±0.1**
w/o filter	**59.6±0.3**	40.8±0.3	48.5±0.3
w/o ER	53.0±0.2	**46.2±0.3**	49.3±0.1

Table 1: Macro-averaged Precision, Recall, and F1 of different approaches on Ultra-Fine test set.

- **Box** [Onoe *et al.*, 2021] proposes to use box embeddings instead of vector embeddings to capture latent type hierarchies.
- **LRN** [Liu *et al.*,] learns and reasons label dependencies in both sequence-to-set and ene-to-end manner.

For the OntoNotes dataset, in addition to the baselines mentioned above, we also compare with several approaches focusing on noisy labeling problems introduced in related work [Chen *et al.*, ; Zhang *et al.*, ; Wu *et al.*,].

Implementation. We use BERT-base-cased as backbone structures, AdamW optimizer with the learning rate of 2e-5 on Ultra-fine and 2e-6 on OntoNotes. The training batch size is 16 for all datasets. Other hyper-parameters are tuned by grid search and the optima configurations are $\{\epsilon = 0.1, \alpha = 2.0, \beta = 0.5, k = 2000\}$ on Ultra-Fine; $\{\epsilon = 0.4, \alpha = 3.0, \beta = 2.0, k = 5000\}$ on augmented OntoNotes; $\{\epsilon = 0.3, \alpha = 2.0, \beta = 1.0, k = 5000\}$ on original OntoNotes.

4.2 Overall Results

Table 1 and Table 2 show the overall results on Ultra-Fine and OntoNotes test sets. We report the performance of our model (described in Section 3.2) training on the original training set (original) and the denoised training set (denoised) respectively. We can see that: i) Our noisy label correction method is effective across datasets. After using the denoised training set generated by our method, the model performance significantly improved and outperform other advanced baselines on almost all metrics; ii) Generate cleaner training data is critical to FET. Both our method and MLMET generate cleaner data to train the FET model. We can see they significantly outperform other baselines. iii) Prompt-learning is an effective approach to leverage PLMs in FET. Compared with methods using vanilla fine-tuning (*e.g.,* Box, LRN), prompt-leaning brings significant performance gains. Training on the original noisy training data, our prompt-learning model still achieves competitive performance against other baselines.

Compared with previous works focusing on noisy labeling problems in FET, our method is more universal. NFETC-CLSC$_{hier}$, NFETC-AR$_{hier}$ and FBTree require pre-defined hierarchical type structures to detect potentially noisy labels. They are difficult to apply to datasets like Ultra-Fine whose type set is not being explicitly hierarchical. LDET requires a small set of manually labeled training data, which is not provided by datasets like OntoNotes. In contrast, our method corrects noisy labels without auxiliary resources, which can be applied to all FET datasets. It's also worth noting that our approach is orthogonal to most existing FET models. In practice, our method can combine with other advanced FET models to yield better performance.

Method	Acc	Macro F1	Micro F1
with augmentation			
UFET	59.5	76.8	71.8
LabelGCN	59.6	77.8	72.2
LDET	64.9	84.5	79.2
LRN	64.5	84.5	79.3
MLMET	67.4	85.4	80.4
Ours (original)	63.7±0.3	84.8±0.2	79.6±0.4
Ours (denoised)	**67.8±0.1**	**87.1±0.2**	**81.5±0.1**
w/o filter	67.3±0.4	87.0±0.1	81.3±0.1
w/o ER	65.3±0.3	86.2±0.3	81.0±0.2
without augmentation			
LRN	56.6	77.6	71.8
NFETC-CLSC$_{hier}$	62.8±0.3	77.8±0.4	72.0±0.4
NFETC-AR$_{hier}$	**64.0±0.3**	78.8±0.3	73.0±0.3
FBTree	**64.0±0.1**	78.4±0.2	72.5±0.2
Ours (original)	52.6±2.2	77.1±1.6	71.1±1.2
Ours (denoised)	59.2±0.2	**81.3±0.3**	**75.3±0.4**
w/o filter	59.6±0.6	79.8±0.3	74.6±0.3
w/o ER	56.9±0.3	80.3±0.3	73.8±0.2

Table 2: Strict accuracy, macro-averaged F1, and micro-averaged F1 of different approaches on OntoNotes test set.

4.3 Ablation Study

Filter obvious noisy labels (filter) can estimate the posterior probability $p(y = 1|l)$ more accurately, which is critical to potentially noisy label selection. Entropy regularization (ER) encourages decision boundaries to fall in low-density regions, which is helpful for noisy label correction. To better understand their effect on the model, we conduct the ablation study on Ultra-Fine and OntoNotes datasets. The results are reported in Table 1 and Table 2.

Effect of Filter Obvious Noisy Labels. This strategy brings improvements to all datasets, especially in the Ultra-Fine dataset. Without filtering obvious noisy labels, our method only brings 0.5% improvement on Macro-averaged F1 compared to training on the original training set. This is mainly due to the long tail problem in Ultra-Fine: many long-tail types only have several positive samples. So the outliers (obvious noisy labels) can seriously affect the estimation of likelihood function $p(l|y = 1)$, lead to a poor estimation of $p(y = 1|l)$. In practice, we recommend enabling this strategy in datasets with long-tail problems.

Effect of Entropy Regularization. Entropy regularization brings improvements to all datasets. This indicates that treat-

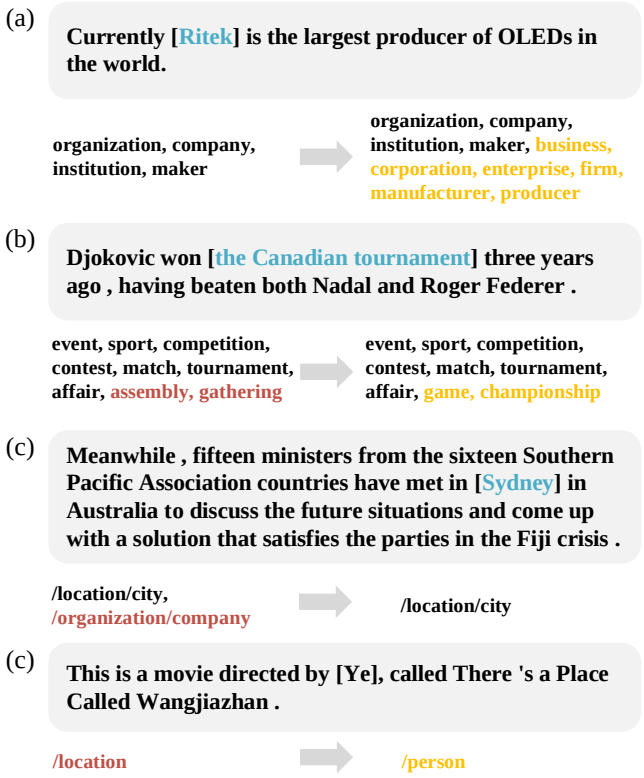


Figure 3: Example of noisy labels (left) and the denoised labels (right). Example (a), (b) are manually-annotated samples taken from Ultra-Fine; (c), (d) are distantly-labeled samples taken from OntoNotes. False positive labels are colored in red, false negative labels are colored in yellow.

ing potentially noisy labels as unlabeled data and training the model in a semi-supervised manner helps with noisy label correction.

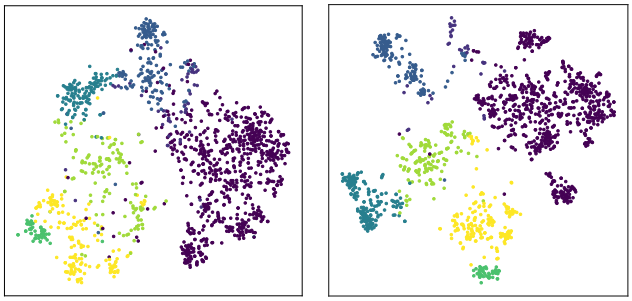
4.4 Case Study

Figure 3 shows examples of the original noisy labels and the denoised labels produced by our method. In example (a), the original human-annotated labels fail to cover all proper types. Our method generates a more comprehensive type set. In example (b), our method removes labels that are not highly correlated with this sample (assembly, gathering) and supplements two fine-grained labels (games, championship). In example (c), our method retains the correct label "/location/city" and removes the false positive label "/organization/company". In example (d), our method reassigns the correct label for this sample.

4.5 Visualization

To better illustrate how noisy labels affect model performance in FET, we train the model using the original and denoised training data separately and visualize mention embeddings in Figure 4[1].

[1]Some mentions may have multiple coarse-grained types, we only visualize mentions with one coarse-grained type.



(a) Original (b) Denoised

Figure 4: T-SNE virtualization of mention embeddings on the original (left) and denoised (right) training set of Ultra-Fine when achieving early stopping. Samples are shown in different colors according to their coarse-grained types.

In Figure 4(a), there is overlap between different type clusters. Intuitively, noisy labels in training sets prevent the model from learning a clear decision boundary for each type. [Pleiss *et al.*, 2020] offers an explanation for this. Take a mislabeled sample for example. The gradient updates from those similar but correctly labeled samples encourage the model to predict the underlying ground truth through generalization. However, the gradient update from this mislabeled sample itself forces the model to memorize the incorrectly assigned label. These two opposing gradient updates result in a poor margin between different types. Consequently, the model is easy to make incorrect predictions for samples near decision boundaries. From Figure 4(b) we can see the margin between different types increases and the decision boundaries are clear. This phenomenon shows correcting the noisy labels brings more discriminative sample representation in FET, and our method indeed generates cleaner training data.

5 Conclusion

In this paper, we focus on the noisy labeling problem in the fine-grained entity typing task and propose a novel approach to automatically correct the noisy labels. Different from previous denoise approaches, our method does not require auxiliary resources, which is more universal. Experiments on two benchmark datasets prove the effectiveness of our method.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No.61602197, Grant No.L1924068, Grant No.61772076, in part by CCF-AFSG Research Fund under Grant No.RF20210005, in part by the fund of Joint Laboratory of HUST and Pingan Property & Casualty Research (HPL), and in part by the National Research Foundation (NRF) of Singapore under its AI Singapore Programme (AISG Award No: AISG-GC-2019-003). Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not reflect the views of National Research Foundation, Singapore. The authors would also like to thank the anonymous reviewers for their comments on improving the quality of this paper.

References

[Arpit *et al.*,] Devansh Arpit, Stanislaw Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron C. Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *ICML*.

[Chen *et al.*,] 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*.

[Choi *et al.*,] Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. Ultra-fine entity typing. In *ACL*.

[Dai *et al.*,] Hongliang Dai, Yangqiu Song, and Haixun Wang. Ultra-fine entity typing with weak supervision from a masked language model. In *ACL*.

[Ding *et al.*, 2021] Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. Prompt-learning for fine-grained entity typing. *arXiv*, 2021.

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

[Grandvalet *et al.*,] Yves Grandvalet, Yoshua Bengio, et al. Semi-supervised learning by entropy minimization. *CAP*, 367.

[Han *et al.*, 2018] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018.

[Koch *et al.*,] Mitchell Koch, John Gilmer, Stephen Soderland, and Daniel S Weld. Type-aware distantly supervised relation extraction with linked arguments. In *EMNLP*.

[Li *et al.*, 2020] Junnan Li, Richard Socher, and Steven C. H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020.

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

[Liu *et al.*,] Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. Fine-grained entity typing via label reasoning. In *EMNLP*.

[Nguyen *et al.*, 2020] Tam Nguyen, C Mummadi, T Ngo, L Beggel, and Thomas Brox. Self: learning to filter noisy labels with self-ensembling. In *ICLR*, 2020.

[Onoe and Durrett, a] Yasumasa Onoe and Greg Durrett. Fine-grained entity typing for domain independent entity linking. In *AAAI*.

[Onoe and Durrett, b] Yasumasa Onoe and Greg Durrett. Interpretable entity representations through large-scale typing. In *EMNLP: Findings*.

[Onoe and Durrett, c] Yasumasa Onoe and Greg Durrett. Learning to denoise distantly-labeled data for entity typing. In *NAACL*.

[Onoe *et al.*, 2021] Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. Modeling fine-grained entity types with box embeddings. In *ACL*, 2021.

[Pleiss *et al.*, 2020] Geoff Pleiss, Tianyi Zhang, Ethan R. Elenberg, and Kilian Q. Weinberger. Identifying mislabeled data using the area under the margin ranking. In *NeurIPS*, 2020.

[Ren *et al.*, a] 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*.

[Ren *et al.*, b] 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 *SIGKDD*.

[Shang *et al.*,] Yuming Shang, He-Yan Huang, Xian-Ling Mao, Xin Sun, and Wei Wei. Are noisy sentences useless for distant supervised relation extraction? In *AAAI*.

[Shang *et al.*, 2022] Yuming Shang, Heyan Huang, Xin Sun, and Xianling Mao. Learning relation ties with a force-directed graph in distant supervised relation extraction. *TOIS*, 2022.

[Wei *et al.*, a] Wei Wei, Gao Cong, Xiaoli Li, See-Kiong Ng, and Guohui Li. Integrating community question and answer archives. In *AAAI*.

[Wei *et al.*, b] Wei Wei, Jiayi Liu, Xianling Mao, Guibing Guo, Feida Zhu, Pan Zhou, and Yuchong Hu. Emotion-aware chat machine: Automatic emotional response generation for human-like emotional interaction. In *CIKM*.

[Wei *et al.*, c] Wei Wei, Jiayi Liu, Xianling Mao, Guibing Guo, Feida Zhu, Pan Zhou, Yuchong Hu, and Shanshan Feng. Target-guided emotion-aware chat machine. *TOIS*, 39(4).

[Wei *et al.*, d] Wei Wei, ZhaoYan Ming, Liqiang Nie, Guohui Li, Jianjun Li, Feida Zhu, Tianfeng Shang, and Changyin Luo. Exploring heterogeneous features for query-focused summarization of categorized community answers. *Information Sciences*, 330.

[Wu *et al.*,] Junshuang Wu, Richong Zhang, Yongyi Mao, Masoumeh Soflaei Shahrbabak, and Jinpeng Huai. Hierarchical modeling of label dependency and label noise in fine-grained entity typing. In *IJCAI*.

[Xiong *et al.*,] Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. Imposing label-relational inductive bias for extremely fine-grained entity typing. In *NAACL*.

[Zhang *et al.*,] Haoyu Zhang, Dingkun Long, Guangwei Xu, Muhua Zhu, Pengjun Xie, Fei Huang, and Ji Wang. Learning with noise: improving distantly-supervised fine-grained entity typing via automatic relabeling. In *IJCAI*.

[Zhang *et al.*, 2017] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *ICLR*, 2017.