Divide and Denoise: Learning from Noisy Labels in Fine-grained Entity Typing with Cluster-wise Loss Correction

Anonymous ACL submission

Abstract

Fine-grained Entity Typing (FET) has made great progress based on distant supervision but still suffers from label noise. Existing FET noise learning methods rely on prediction distributions in an instance-independent manner, which causes the problem of confirmation bias. In this work, we propose a clustering-based loss correction framework named Feature Cluster Loss Correction (FCLC), to address these two problems. FCLC first train a coarse backbone model as a feature extractor and noise estimator. Loss correction is then applied to each feature cluster, learning directly from the noisy labels. Experimental results on three public datasets show that FCLC achieves the best performance over existing competitive systems. Auxiliary experiments further demonstrate that FCLC is stable to hyperparameters and it does help mitigate confirmation bias. We also find that in the extreme case of no clean data, the FCLC framework still achieves competitive performance.

1 Introduction

Fine-grained entity typing (FET) is the task of classifying named entity mentions in a sentence over the given class set (typically a hierarchical class structure as shown in Fig. 1. FET serves as an important component in many down-stream NLP applications, e.g., relation extraction (Liu et al., 2014), entity linking (Raiman and Raiman, 2018) and question answering (Dong et al., 2015). FET task has a more wide range of entity types (usually over 100 classes) compared to entity typing, and hence neural-based FET systems require large-scale annotated training corpus.

Recent studies apply distant supervision to label the corpora automatically by linking mentions to knowledge base entities and using all entity types as the ground-truth labels. Although large-scale annotated data is provided, it brings about label noises in training. To overcome the problem of noisy label, some works directly pruned noisy instances (Gillick et al., 2014; Onoe and Durrett, 2019a). The others retain noisy training data but further improve by choosing (Ren et al., 2016a; Xu and Barbosa, 2018), weighting (Wu et al., 2019), and relabeling (Zhang et al., 2020) noisy labels using the prediction distribution.

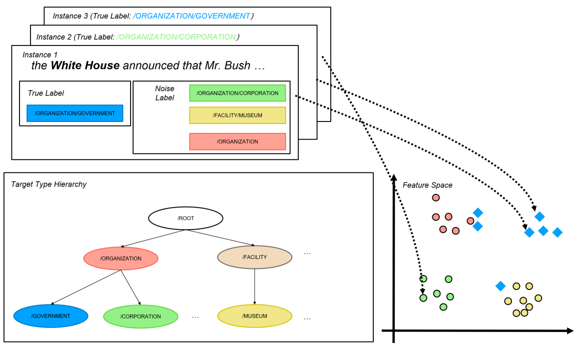

Figure 1: An Example of noisy labels and feature space illustration in FET task.

However, these noise combating methods have two major limitations. 1) They rely on the prediction distribution. As a result, they ought to cope with instance-agnostic noise better. The previous works expirically show (Zheng and Yang, 2021) that the prediction distribution is more likely to be affected by noisy instances and suffer from **confirmation bias**. This bias problem is also verified in our Sec. 3.5. The limitation leads to the intriguing question: ***Besides prediction distribution and entropy, what other information can we use to model label noise?***

2) They mostly aim to modify each instance isolatedly and only use instance-level information. Meanwhile, typical anti-noise machine learning (Patrini et al., 2017; Hendrycks et al., 2018) uses instance-agnostic global statistics. The latter is more robust to noise but might be too general. Local information is potentially more informative. For example, when the distant supervision introduces similar noise in some instances, these noises form

a locality in feature space. The noisy instances are near to each other and are separate from instances with the same but true labels. Our experiment result is similar to Fig. 1, even when the feature extractor is trained to fit noisy labels, they are still easily separable due to underlying semantic differences.

These two limitations are inter-related, causing noise-learning-based FET methods to still suffer from distantly supervised noise. To alleviate the label noise and avert these limitations, we propose a novel framework ***FCLC*** for noisy label learning inspired by weighted training and loss correction (Hendrycks et al., 2018) in machine learning. Our method utilizes feature representations from the model and learns global (local) information, i.e. a cluster-level label confusion matrix. Firstly, we use a backbone learner on noisy data. It serves as a feature extractor and a noise estimator. Secondly, all training data, including noisy data and a small portion of clean data are clustered. The clean data serve as anchors in the feature space to estimate label corruption and sample quality of each cluster. Finally, label corruption and sample quality are used for label correction.

Our main contributions are three-fold: (i) This study provides fresh insight into instance dependant label noise in FET. We pointed out a novel training method to further exploit feature space and global information. (ii) We designed a framework with feature clustering, estimating cluster-level confusion matrix, and loss correction. (iii) We experimented the proposed method on three datasets. Results show that we made significant improvements over previous state-of-the-art, thus proving the effectiveness of our model. Ablation studies further prove the robustness and wide applicability of our framework.

2 Framework

2.1 Definition

Given a finite set of types, $T = \{t_1, t_2, ..., t_{|T|}\}$, where $|T|$ denotes the number of candidate types. The task is to assign appropriate types to each mention under context. Formally, an instance is a triplet, $(m, c, \boldsymbol{y})$. $c = \{w_1, w_2, ..., w_n\}$ is the context of m, usually the original sentence. $m = \{w_{p_1}, ..., w_{p_l}\}$ is the mention. obviously, m is a continuous subsequence of c.

$Y \subseteq T$ denotes appropriate types for (m, c). For convenience, denote Y's vector form $\boldsymbol{y} \in \{0, 1\}^{|T|}$, $y_j = 1$ means $t_j \in Y$.

When the instance is produced with crowdsourcing or distant supervision, annotated labels might contain so-called noise. We denote labels with noise $\tilde{\boldsymbol{y}}$. The instance is thus $(m, c, \tilde{\boldsymbol{y}})$. Denote the corpus with noisy instances $\tilde{\mathcal{D}}$, the corpus with trusted instances $\mathcal{D}_t$.[1] The two corpus form the whole training corpus $\mathcal{D}$.

The task is to predict the appropriate types for given (m, c).

2.2 Training Procedure

As shown in Fig. 2, the **FCLC** framework consists of the following steps :

Step 1. (Phase 1) Train the backbone model with noisy data $\tilde{\mathcal{D}}$ for e_1 epochs and get M_1. It serves as a feature extractor and a noise estimator. (Sec. 2.3)

Step 2. Cluster all training samples $\mathcal{D}$ with the feature extracted by E_1, and estimate confusion matrix for each cluster with predictions of M_1. (Sec. 2.4)

Step 3. (Phase 2) The calculated clustering-aware confusion matrix and FCLC loss are used to continue training the backbone model. (Sec. 2.5)

2.3 Backbone

For fair comparison, the backbone of our model has the same structure as NFETC (Xu and Barbosa, 2018).

For an instance $(m, c, \boldsymbol{y})$, for each word w_i in c, word embedding is $\boldsymbol{e_i^w} \in \mathbb{R}^{d_w}$ looked up in word embedding matrix $\mathbf{W} \in \mathbb{R}^{d_w \times |V|}$.

A position embedding $\boldsymbol{e_i^p} \in \mathbb{R}^{d_p}$ is used to model the context word position i and mention position (p_1, p_l) by looking up relative position in position embedding matrix $\mathbf{P} \in \mathbb{R}^{d_p \times 2N}$. The final embedding is the concatenation $\boldsymbol{e_i} = [\boldsymbol{e_i^w}, \boldsymbol{e_i^p}]$.

Context Representation A Bi-LSTM (Hochreiter and Schmidhuber, 1997) is used to model the context representation. Feeding the embedding of c i.e. $\{\boldsymbol{e_1}, \boldsymbol{e_2}, ..., \boldsymbol{e_n}\}$ into BiLSTM gets the two directional hidden states $\overrightarrow{\boldsymbol{h_i}}$ and $\overleftarrow{\boldsymbol{h_i}}$ for each word w_i. Word level attention following (Zhou et al., 2016) is applied on $\boldsymbol{h_i} = [\overrightarrow{\boldsymbol{h_i}} \oplus \overleftarrow{\boldsymbol{h_i}}]$, resulting in the final context representation $\boldsymbol{r_{c_i}}$.

Mention Representation The average encoder of a mention takes word embeddings of the mention $\{\boldsymbol{e_{p_1}}, \boldsymbol{e_{p_2}}, ..., \boldsymbol{e_{p_l}}\}$ and takes the average: $\boldsymbol{r_w} =$

[1]Normally $|\mathcal{D}_t| \ll |\tilde{\mathcal{D}}|$, as in all the datasets we reported in this paper.

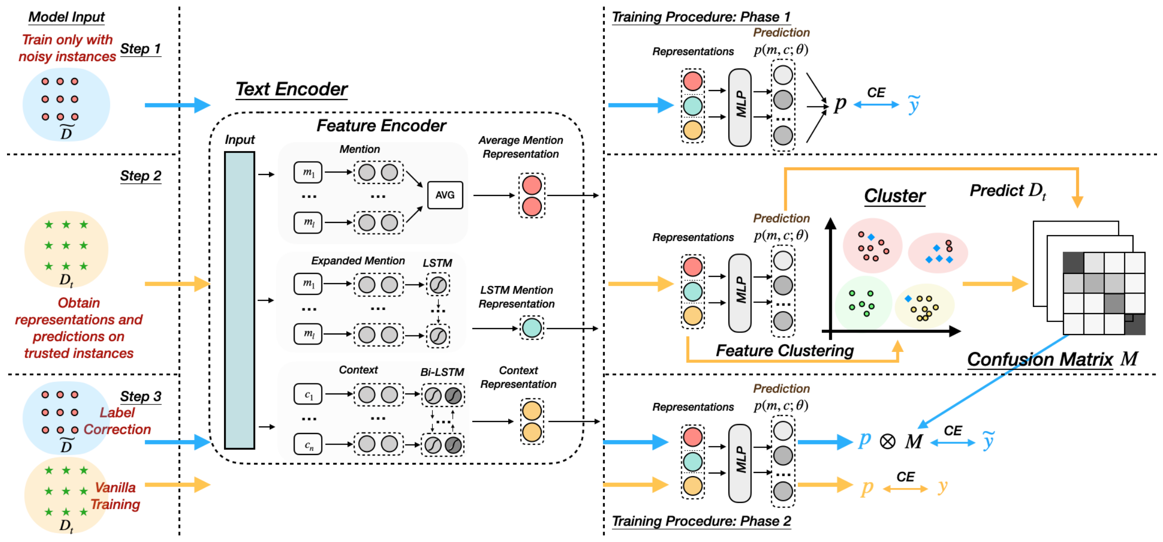

Figure 2: Model architecture.

$\frac{1}{l}\sum_{k=1}^{l} \boldsymbol{e_{p_k}}$. The LSTM encoder of a mention takes an extended mention with one more token before and after the original mention and produces hidden state features $\{\boldsymbol{h_{p_1-1}}, ..., \boldsymbol{h_{p_l+1}}\}$. Take the last output $\boldsymbol{h_{p_l+1}}$ as $\boldsymbol{r_l}$. The final representation of the mention is $\boldsymbol{r_m} = [\boldsymbol{r_a}, \boldsymbol{r_l}]$

Classification Softmax classifier and cross-entropy are used based on the feature $\boldsymbol{r_{m,c}} = [\boldsymbol{r_c}, \boldsymbol{r_m}]$ of x:

$$\begin{aligned} \boldsymbol{s}(x) &= \mathbf{W}\boldsymbol{r_{m,c}} + \mathbf{b} && (1)\\ \hat{\boldsymbol{p}}(\boldsymbol{y}|x) &= \text{softmax}(\boldsymbol{s}(x)) && (2)\\ \ell(x, \boldsymbol{y}; \theta) &= -log(\hat{p}(\boldsymbol{y}|x)) && (3) \end{aligned}$$

With a given dataset D, the model is trained with all samples $(x, \boldsymbol{y_l})$ in D. For baseline, $D = \mathcal{D}$. For **FCLC** step 1, $D = \tilde{\mathcal{D}}$:

$$\mathcal{L}_{base}(\theta) = \frac{1}{|D|} \sum_{(x,\boldsymbol{y}) \in D} \ell(x, \boldsymbol{y}; \theta) \qquad (4)$$

2.4 Feature Clustering

We make the assumption that the noise $(\boldsymbol{y}, \tilde{\boldsymbol{y}})$forms locality in the feature space, especially when the feature is calculated from the original mention and context(m, c), (m, c) determines $\boldsymbol{y}$, and the feature is trained with $\tilde{\boldsymbol{y}}$.

We adopt clustering to utilize local statistics as smaller-grained feature information. To be specific, we perform k-means with $\boldsymbol{r_{m,c}}$ on the whole training set D, and separate D into K clusters. Denote the k-th cluster $\bar{\mathcal{C}}_k$, $\mathcal{C}_{t-k} = \bar{\mathcal{C}}_k \cap \mathcal{D}_t$, $\tilde{\mathcal{C}}_k = \bar{\mathcal{C}}_k \cap \tilde{\mathcal{D}}$.

We mainly utilize the two following statistics:

$$\tau_k = \frac{|\mathcal{C}_k|}{\mathcal{D}} \qquad (5)$$

τ_k estimates the quality of the cluster k it act as a soft cluster sieving.

$$\widehat{C}_{ijk} = \frac{1}{|A_{ik}|} \sum_{(x,y) \in A_{ik}} \hat{p}(\boldsymbol{y}_j = 1|x) \qquad (6)$$

where $A_{ik} = \{(x, \boldsymbol{y})|(x, \boldsymbol{y}) \in \mathcal{C}_{t-k} \text{ and } \boldsymbol{y}_i = 1\}$, estimates the probability in cluster k to annotate noise j for true label i.

2.5 Loss Correction

The idea of forward loss correction is proposed by Patrini et al. (2017). The basic idea is to modify the loss with the noise transition matrix T. Such that the minimizer under the new loss with noisy labels is the same as the minimizer of the original loss under clean labels. The modification relies on the assumption that the label noise is independent from instances, i.e. $\tilde{\boldsymbol{y}} \perp x \mid \boldsymbol{y}$. Hendrycks et al. (2018) proposed to estimate $\mathbf{T}$ with a small set of clean labels, under the assumption that $\tilde{\boldsymbol{y}} \perp \boldsymbol{y} \mid x$. While these assumptions do not hold globally for distantly supervised FET, they hold better in clusters. We introduce the cluster-wise loss correction in the following sections.

Transition Matrix Estimation Assuming the backbone model is well trained, i.e. $\hat{p}(\tilde{\boldsymbol{y}}_j = 1|x)$ is close enough to $p(\tilde{\boldsymbol{y}}_j = 1|x)$. We use the predicted probability on trusted instances in cluster-k

to estimate the transition probability.

$$\begin{aligned} C_{ijk} &= p(\tilde{\boldsymbol{y}}_j = 1 \mid \boldsymbol{y}_i = 1, x \in \tilde{\mathcal{C}_k}) \\ &\approx p(\tilde{\boldsymbol{y}}_j = 1 \mid \boldsymbol{y}_i = 1, x \in \mathcal{C}_{t-k}) \\ &\approx \frac{1}{|A_{ik}|} \sum_{(x,y)\in A_{ik}} \hat{p}(\tilde{\boldsymbol{y}}_j = 1|x) \\ &= \widehat{C}_{ijk} \end{aligned} \tag{7}$$

Forward Loss Correction Cross-entropy is composite (Reid and Williamson, 2010),denote it as ℓ_ψ, its inverse link function ψ^{-1} is softmax.

Notice C_{ijk} can bridge the loss with noisy label $\tilde{\boldsymbol{y}}, (x \in \tilde{\mathcal{C}_k}, \tilde{\boldsymbol{y}}_i = 1)$, to predictions for the true label:

$$-log(\hat{p}(\tilde{\boldsymbol{y}}|x)) \approx -\log \sum_{j=1}^{c} C_{jik}\hat{p}(\boldsymbol{y} = \boldsymbol{e}^j \mid x) \tag{8}$$

Let $T_k = C_{**k}$, define the forward loss as:

$$\boldsymbol{\ell}_\psi^{\rightarrow}(\boldsymbol{s}(\boldsymbol{x})) = \boldsymbol{\ell}_\psi(T_k^\top \boldsymbol{s}(x)) \tag{9}$$

The property holds on each cluster similar as in (Patrini et al., 2017), with all $x \in \tilde{\mathcal{C}_k}$, training with noisy label $\tilde{\boldsymbol{y}}$ on $\boldsymbol{\ell}_\psi^{\rightarrow}$ is the same as with true label $\boldsymbol{y}$ on the original loss $\boldsymbol{\ell}_\psi$:

$$\underset{\boldsymbol{h}}{\text{argmin}} \mathbb{E}_{\boldsymbol{x},\tilde{\boldsymbol{y}}} \boldsymbol{\ell}_\psi^{\rightarrow}(\boldsymbol{s}(\boldsymbol{x})) = \underset{\boldsymbol{s}}{\text{argmin}} \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}} \boldsymbol{\ell}_\psi(\boldsymbol{s}(\boldsymbol{x})) \tag{10}$$

Different from global forward loss correction, the parameters that minimize the loss in each cluster are not the same. We balance the clusters with τ_k. The trusted samples $(x, y) \in \mathcal{D}$ are also used. The loss of the full model is:

$$\begin{aligned} \mathcal{L}_{\textbf{FCLC}} &= \textstyle\sum_{(x,y)\in\mathcal{D}_t} \ell_\psi(\boldsymbol{h}(x)) \\ &+\beta \textstyle\sum_{k=1}^{K} \tau_k \sum_{(x,\tilde{y})\in\tilde{\mathcal{C}_k}} \ell_\psi^{\rightarrow}(\boldsymbol{h}(x))) \\ &+(1-\beta) \textstyle\sum_{k=1}^{K} \tau_k \sum_{(x,\tilde{y})\in\tilde{\mathcal{C}_k}} \ell_\psi(\boldsymbol{h}(x))) \end{aligned} \tag{11}$$

Where β is the hyperparameter to balance FCLC loss and the original loss.

Our introduced framework has several advantages: 1) **Lightweight**. This method does not include extra trainable parameters to the backbone model. 2) **Stable**. The framework involves two hyperparameters, β and phase-1 train epochs e_1 and we empirically find them stable. 3) **Flexibility**. Our improvement is orthogonal to the backbone model. It only requires that the backbone model is sufficiently expressive and uses an appropriate composite loss (Reid and Williamson, 2010). Thus, it is pluggable to a large number of FET models.

3 Experiments

We evaluate the proposed model on three different FET datasets and compare it to several state-of-the-art models. In addition, to support our claims we also conduct several subsidiary experiments to analyze the impacts of our proposed module in detail.

	Wiki	**OntoNotes**	**BBN**
types	113	89	47
hierarchy depth	2	3	2
mentions-train	2009898	253241	86078
⊢mentions-train-trusted	9999	2202	642
⊢mentions-train-noisy	1999899	251039	85436
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 1: Fine-Grained Entity Typing datasets Statistics.

3.1 Datasets

The datasets are described below, we use exactly the same train/dev/test split with previous works (Ren et al., 2016a; Chen et al., 2019). Detailed statistics of the three datasets are also shown in Table 1. **BBN** It contains sentences extracted from the Wall Street Journal and distantly labeled by DBpedia Spotlight (Weischedel and Brunstein, 2005). **OntoNotes** It was constructed using sentences in the OntoNotes corpus and distantly supervised by DBpedia Spotlight (Weischedel et al., 2013). **Wiki/FIGER** It was derived from Wikipedia articles and news reports, entities of the training samples are distantly annotated using Freebase (Ling and Weld, 2012).

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
Phase 1 Epochs (e_1)	5	14	20
#Clusters (k)	116	104	42
LC Loss Weight (β)	0.25	0.35	0.95

Table 2: Hyper-parameters chosen for the three datasets.

3.2 Evaluation Metrics

We follow prior work and use the strict accuracy (Acc), Macro F1 (Ma-F1), and Micro F1 (Mi-F1) scores. During the experiment, all these metrics are calculated by running the model five times and computing the mean and standard deviation values.

Model	Wiki			OntoNotes			BBN		
	Strict Acc	Macro F1	Micro F1	Strict Acc	Macro F1	Micro F1	Strict Acc	Macro F1	Micro F1
AFET(2016a)	53.3	69.3	66.4	55.3	71.2	64.6	68.3	74.4	74.7
Attentive(2016)	59.7	80.0	75.4	51.7	71.0	64.91	48.4	73.2	72.4
NFETC(2018)	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
w/ *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
CLSC(2019)	-	-	-	59.6±0.3	75.5±0.4	69.3±0.4	74.7±0.3	80.7±0.2	80.5±0.2
w/ *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(2020)	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
w/ *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
NFETC-VAT(2020)	-	-	-	63.8	78.7	73.0	76.7	80.7	80.9
CLSC-VAT(2020)	-	-	-	63.9	78.6	73.1	76.9	81.2	81.4
ML-L2R(2020)	69.1	82.6	80.8	58.7	73.0	68.1	75.2	79.7	80.5
Box(2021)	-	81.6	77.0	-	77.3	70.9	-	78.7	78.0
FCLC	58.0±1.7	77.8±0.8	76.2±0.8	62.7±1.1	77.5±0.7	71.4±0.7	**82.0±0.8**	**86.2±0.7**	**86.7±0.7**
FCLC$_{hier}$	**71.3±1.1**	82.2±0.7	**81.1±0.6**	**65.3±0.2**	**79.6±0.3**	**74.0±0.3**	79.0±0.5	84.2±0.5	84.8±0.5
w/o τ_k	70.9±1.6	81.8±1.0	80.7±1.1	64.6±0.2	78.8±0.2	73.1±0.3	81.6±0.4	85.9±0.4	86.5±0.4
w/o loss correction	70.4±1.4	81.6±1.0	80.5±0.9	64.2±0.3	78.4±0.3	72.6±0.5	76.5±0.5	81.0±0.4	81.2±0.4
w/o cluster	71.3±0.4	82.0±0.6	80.9±0.5	64.6±0.3	79.2±0.3	73.4±0.2	79.2±0.6	83.2±0.5	83.7±0.6
w/ reinit	69.7±2.4	81.2±1.2	80.1±1.3	62.4±0.3	77.8±0.7	71.7±0.7	79.9±0.9	84.2±0.9	84.6±0.6

Table 3: Performance results on three benchmark datasets.

3.3 Baselines

We consider the following competitive FET systems as our baselines: (1) AFET (Ren et al., 2016a); (2) Attentive (Shimaoka et al., 2016); (3) NFETC/NFETC$_{hier}$ (Xu and Barbosa, 2018); (4) CLSC/CLSC$_{hier}$ (Chen et al., 2019); (5) NFETC-AR/NFETC-AR$_{hier}$ (Zhang et al., 2020); (6) NFETC-VAT/CLSC-VAT (Shi et al., 2020); (7) Multi Level Learning to Rank (ML-L2R) (Chen et al., 2020); (8) Box (Onoe et al., 2021).

These baselines are compared with several variants of our proposed model: (1) **FCLC**: proposed model without the hierarchical loss; (2) **FCLC**$_{hier}$ proposed model with the hierarchical loss; (3) **FCLC**(without τ_k) our proposed model trained without cluster quality estimation, i.e. $\tau = 1$ for all clusters; (4) **FCLC**(without loss correction) our proposed model without loss correction, only cluster quality estimation working; (5)**FCLC**(without cluster) our proposed model without clustering, i.e. calculated a globally-uniform confusion matrix; (6) **FCLC**(with reinit): our proposed model with fresh parameters before the start of step 3 as suggested by Patrini et al. (2017).

3.4 Implementation Details

To make an equal comparison, following (Xu and Barbosa, 2018; Chen et al., 2019; Zhang et al., 2020), we use exactly the same pre-trained 300-dimensional GloVe word embeddings (Pennington et al., 2014) and fix the embedding vectors during training. The model parameters are optimized using the Adam (Kingma and Ba, 2014) optimizer. All of our models are implemented in Tensorflow. [2] As NFETC and NFETC$_{hier}$ are our backbone models, we follow the hyper-parameters of the backbone except for our introduced hyper-parameters β and e_1. The detailed hyper-parameter settings on the three datasets are shown in Table 2, we also report hyper-parameter impact curves in Fig. 3.

3.5 Results and Analysis

Main Result Table 3 shows the results of our proposed approach (**FCLC**) and several competitive FET systems. We highlight the statistically significant best scores of each metric in bold. According to the experimental results, we make two main observations:

(1) The performances of our proposed model surpass the backbone NFETC model by a remarkable large margin (improving Micro F1 by 2.1%, 3.8%, and 7.8% separately), demonstrating the benefits of the proposed two-phase FCLC module. The relative performance improvements are consistent with or without the hierarchy loss (compared **FCLC** and **FCLChier** to the corresponding baselines).

(2) Compared to other noisy learning methods such as CLSC, NFETC-AR, and VAT, our model still achieves considerable improvements under most metrics when using the same backbone and very similar hyper-parameter settings. For example, compared to NFETC-AR, our model improves Micro-F1 by 1.25% to 6.38% on three datasets. It indicates that, by utilizing both the feature space representations and the global and local statistical

[2]The implementation of our model will be released publicly for further study.

information, the model can reduce the impact of noisy labels more effectively.

Ablation Study To study the detail of our models, we explore the performances of three main model variants, shown in the last several rows of Table 3. We find that the cluster quality τ_k, the loss correction module and the feature cluster process are all critical to model performances in some situations. Specifically, as shown in **FCLC** (without cluster), feature clustering has minor impacts on Wiki and Ontonotes. This is probably because the noisy distribution on these two datasets is relatively simple and the global confusion matrix is sufficient. Moreover, we observe that the re-initialization before Step 3 has a great impact on all metrics. Staring Step 3 with a fresh re-initialized FET model degrades the accuracy by 3.2% on Ontonotes. It denotes that the learner trained in the first phase is beneficial for the noisy robust learning process, by providing optimal parameters initialization.

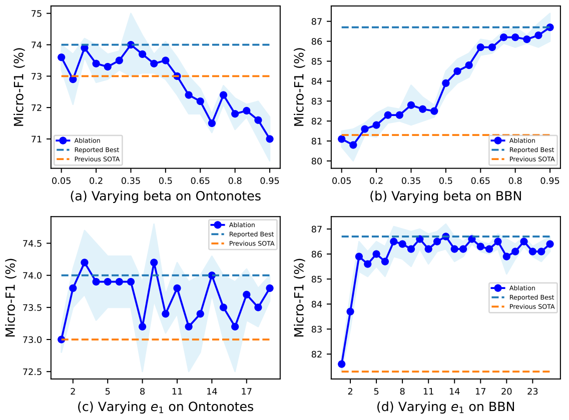

Figure 3: Performance change with respect to β and e_1 on the Ontonotes (sub-figure a, c) and BBN (sub-figure b, d) dataset. The horizontal lines hereinafter denotes for previous SOTA performances and our reported performances.

Sensitivity of the introduced hyper-parameters Using the same setting for model training, Fig. 3 analyses the sensitivity of **FCLC** to the introduced hyper-parameters: the **FCLC** objective weight β, the Step-1 training epochs e_1 . Fig. 3(a, b) shows the performance trend on the Ontonotes and BBN datasets when changing β. While selecting a proper ratio between loss-correction loss and the original loss is important, the performance near optimum β is stable and steadily outperforms the baseline. Fig. 3(c, d) analyses the sensitivity with respect to e_1. the Micro-F1 improves as e_1 increases but stops improving and become unstable when e_1 is large enough, since the model starts to overfit noise. It is also reasonable that the optimal range of β and e_1 in BBN and Ontonotes are different as they have different training set sizes and different distance supervision noise distribution.

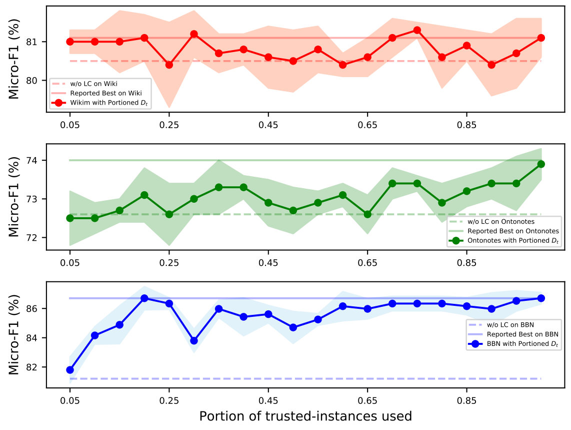

Figure 4: Performance curves with different trusted instance set $\mathcal{D}_t$ sizes on three datasets.

Will cluster number affect performance? We investigate how much the **FCLC** model benefits from different values of feature cluster number k. Fig. 5 demonstrates that under a reasonable feature cluster range (near $|T|$), the model can achieve competitive and similar performances.

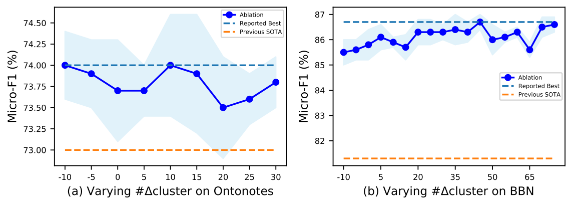

Figure 5: Performance curves under different feature-cluster numbers k on the Ontonotes (a) and BBN (b), #Δcluster represents $k - |T|$.

How many trusted instances does the model need? We examine the robustness of the model to the amount of clean data by comparing the performances with 5% to 100% trusted instances. Refer to Fig. 4, we observe that due to the differences of the training set, our model achieves comparable accuracy with 30%, 40%, and 70% $\mathcal{D}_t$ samples on Wiki, Ontonotes, and BBN separately. With only a very small size of trusted instances, e.g. 20% BBN trusted set, or 128 samples, the model begins to improve significantly.

What if we did not have any trusted instances? Although a small number of clean samples is always practical to obtain or relabel with an expert, we push the limit to no trusted instances at all. What performance can our model achieve in such a situation? We performed the "no clean training set" experiment to test the robustness of our model. In Table 4, **FCLC** (w/o $\mathcal{D}_t$) indicates for the variant that the trusted instances are not used for phase 2 training but only in feature clustering and confusion matrix calculation. In that situation, our approach still has similar performances with previous SotA models on most metrics[3].

FCLC (w/ pl) variant means that, during the clustering process, instead of using the trusted instance set $\mathcal{D}_t$ split from the training set, we introduce a simple and classic pseudo labeling method (Lee et al., 2013) to generate the labels needed by clustering and training. We find that compared to the baseline method, **FCLC** with pseudo labeling still achieves much better performances.

It is proved by results in Table 4 that **FCLC** does not rely on a clean training subset, thus having a wide range of applications.

Models	Wiki			Ontonotes		
	Acc	Ma-F1	Mi-F1	Acc	Ma-F1	Mi-F1
Backbone	68.9	81.9	79.0	60.2	76.4	70.2
NFETC-AR	70.1	**83.2**	80.1	64.0	78.8	73.0
FCLC	**71.3**	82.2	**81.1**	**65.3**	**79.6**	**74.0**
w/o $\mathcal{D}_t$ in phase 2	70.0	81.3	80.2	64.6	79.0	73.3
w/o $\mathcal{D}_t$ & w/ pl	**71.3**	82.1	**81.0**	64.2	78.7	72.9

Table 4: The model performances with no trusted instances on phase 2 (w/o $\mathcal{D}_t$) or on the whole training process (w/ pl).

Visualization of the representations We analyze the role of **FCLC** module by visualizing the feature vectors.

Fig. 6 illustrates samples in a cluster (circled in all 4 sub-figures). From Fig. 6(a), we observe that the backbone model fails to distinguish some samples of class A (/ORGANIZATION/GOVERNMENT, red) and class B (/GPE/COUNTRY, blue), due to noisy labels. Fig. 6(b) shows that our model learns to correct these instances. With **FCLC** the classifier is corrected to predict the right label. Meanwhile, in feature space, the boundary between these samples and the confusing class is also clearer, which means **FCLC** also helps to refine feature extraction with loss correction. Fig. 6(e) shows the row of '/GPE/COUNTRY'. Managing to notice the confusion from '/GPE/COUNTRY' to '/ORGANIZATION/GOVERNMENT' enables our model to perform the appropriate correction. Due to this, **FCLC** are resistant to the noisy labels.

[3]It is worth pointing out that it means our model is trained with fewer instances than previous SOTA, since $\mathcal{D}_t$ is not only a part but a precious trusted part from the training set they use.

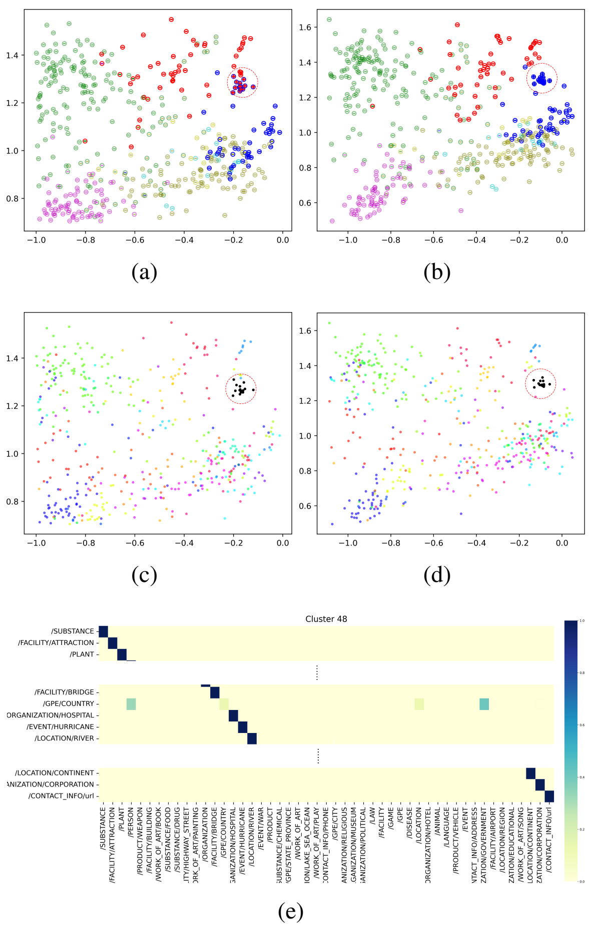

Figure 6: (a, b): the feature representations of backbone and **FCLC** model on BBN test set; (c, d): clusters denoted by colors according to samples in (a, b); (e): the row of '/GPE/COUNTRY' in the circled cluster's confusion matrix.

Quantitative Results of Confirmation Bias To further verify our claim that our model can alleviate the *confirmation bias* in the noisy FET task, we analyze the prediction confidence on test set samples, as shown in Fig. 7. The average confidence of correct and wrong test samples is calculated after each training epoch. The results show that, on the Wiki dataset, after phase one the wrong sample average confidence is 0.700 but the backbone model reached 0.833 at the end of the training (with early stopping). Also, after phase two **FCLC** improves the correct sample confidence from back-

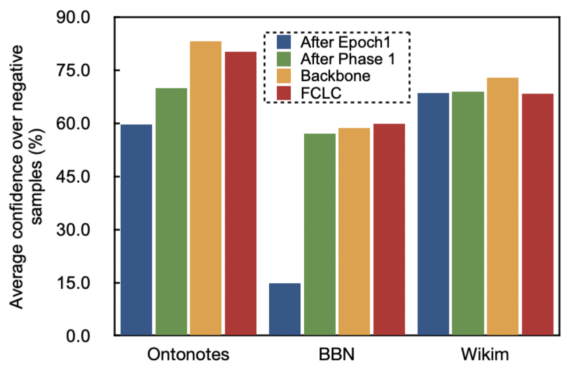

Figure 7: Average prediction confidence over negative predicted samples on three datasets.

bone's 0.939 to 0.950 on Wiki.

4 Related Work

4.1 Noisy Learning

The usage of datasets collected with distant supervision often results in so-called noisy labels. Several studies have investigated deep learning approaches with noise. Existing noisy learning methods include designing robust loss functions (Wang et al., 2019), designing robust architectures by adding noise adaptation layers (Chen and Gupta, 2015; Goldberger and Ben-Reuven, 2017), selecting samples (Onoe and Durrett, 2019b), and adding noise-robust regularization (Shi et al., 2020). Among them, Patrini et al. (2017) and Hendrycks et al. (2018) proposed forward loss correction. It avoided explicit relabeling and matrix inversion. These noisy learning methods are mostly restricted to the noise that is conditionally independent of the data features (Frénay and Verleysen, 2014). However, in real-world applications such as FET, noise distributions are more complex and instance-dependent, requiring more powerful noisy learning methods.

4.2 Fine-Grained Entity Typing

FET is studied based on the distant supervision training data (Mintz et al., 2009; Ling and Weld, 2012). Various features (Yogatama et al., 2015; Xu and Barbosa, 2018), network structures (Dong et al., 2015; Shimaoka et al., 2016), and feature space (Ali et al., 2021; Onoe et al., 2021)are explored to refine the mention and type representation. Label inter-dependency (Lin and Ji, 2019) and type hierarchy (Chen et al., 2020) are often used, added by relations among instances and labels (Ali et al., 2020; Li et al., 2021; Liu et al., 2021). Label noise is the main problem brought by distance supervision. Besides common noisy learning methods discussed in Sec. 4.1 (Onoe and Durrett, 2019b; Shi et al., 2020; Wu et al., 2019), FET-specific noise combat methods are proposed. Ren et al. (2016a,b) utilized partial-label embedding. Xu and Barbosa (2018) modified hierarchical loss to cope with *overly-specific* noise. Zhang et al. (2020) automatically generated pseudo-truth label distribution for each sample. Additional resource also help to improve the performance. The resource include external knowledge base (Xin et al., 2018; Dai et al., 2019), and with BERT-like pipeline (Patel and Ferraro, 2020; Ding et al., 2021). Choi et al. (2018) proposed a way to utilize more distance supervision and crowd source, followed by Onoe and Durrett (2019b). Apart from the above, (Chen et al., 2019) and (Ali et al., 2020) are the closest to our proposed method. They both select some instances by feature distance to modify labels or refine mention representation for noisy instances. However, their refinement is still explicit and isolated to each instance. Thus the quality relies on the instances they retrieve for label propagation/mention reference. Different from these studies, we do not rely on any of these external resources and aim to impose label noise with only the original data without explicit sieving or label changing.

5 Conclusion

In this work, in order to tackle the instance-dependent label noise in fine-grained entity typing tasks, we present a neural FET noisy learning framework that utilizes the feature space information and global information jointly. Experimental results on three publicly available datasets demonstrate that our proposed model achieves the best performance compared with competitive existing FET systems. Furthermore, based on extensive auxiliary experiments, we study the impact of our proposed noisy learning framework in-depth with qualitative and quantitative analysis. In the future, the proposed approach can motivate the need for further understanding of the relationships between dataset noise distribution estimation and the instance features. More work can be done towards this direction. In addition, performances of the proposed framework under different backbone models can be dug to validate the flexibility of the framework.

References

Muhammad Asif Ali, Yifang Sun, Bing Li, and Wei Wang. 2020. Fine-grained named entity typing over distantly supervised data based on refined representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7391–7398.

Muhammad Asif Ali, Yifang Sun, Bing Li, and Wei Wang. 2021. Fine-grained named entity typing over distantly supervised data via refinement in hyperbolic space. *arXiv preprint arXiv:2101.11212*.

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

Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020. Hierarchical entity typing via multi-level learning to rank. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8465–8475. Association for Computational Linguistics.

Xinlei Chen and Abhinav Gupta. 2015. Webly supervised learning of convolutional networks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1431–1439. IEEE Computer Society.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *ACL*, pages 87–96.

Hongliang Dai, Donghong Du, Xin Li, and Yangqiu Song. 2019. Improving fine-grained entity typing with entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6211–6216.

Ning Ding, Yulin Chen, Xu Han, Guangwei Xu, Pengjun Xie, Hai-Tao Zheng, Zhiyuan Liu, Juanzi Li, and Hong-Gee Kim. 2021. Prompt-learning for fine-grained entity typing. *arXiv preprint arXiv:2108.10604*.

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

Benoît Frénay and Michel Verleysen. 2014. Classification in the presence of label noise: A survey. *IEEE Trans. Neural Networks Learn. Syst.*, 25(5):845–869.

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

Jacob Goldberger and Ehud Ben-Reuven. 2017. Training deep neural-networks using a noise adaptation layer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. 2018. Using trusted data to train deep networks on labels corrupted by severe noise. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 10477–10486.

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

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

Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896.

Jinqing Li, Xiaojun Chen, Dakui Wang, and Yuwei Li. 2021. Enhancing label representations with relational inductive bias constraint for fine-grained entity typing. In *IJCAI2021*.

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

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

Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. 2021. Fine-grained entity typing via label reasoning. *arXiv preprint arXiv:2109.05744*.

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

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

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics*.

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

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

Rajat Patel and Francis Ferraro. 2020. On the complementary nature of knowledge graph embedding, fine grain entity types, and language modeling. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 89–99.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952.

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

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *AAAI*, pages 5406–5413.

Mark D. Reid and Robert C. Williamson. 2010. Composite binary losses. *J. Mach. Learn. Res.*, 11:2387–2422.

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

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

Haochen Shi, Siliang Tang, Xiaotao Gu, Bo Chen, Zhigang Chen, Jian Shao, and Xiang Ren. 2020. Alleviate dataset shift problem in fine-grained entity typing with virtual adversarial training.

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

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. 2019. Symmetric cross entropy for robust learning with noisy labels. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 322–330. IEEE.

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

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

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

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

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

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

Haoyu Zhang, Dingkun Long, Guangwei Xu, Muhua Zhu, Pengjun Xie, Fei Huang, and Ji Wang. 2020. Learning with noise: Improving distantly-supervised fine-grained entity typing via automatic relabeling. In *IJCAI*.

Zhedong Zheng and Yi Yang. 2021. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *Int. J. Comput. Vis.*, 129(4):1106–1120.

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