Improving Distantly-supervised Entity Typing with Compact Latent Space Clustering

Bo Chen[1], **Xiaotao Gu**[2], **Yufeng Hu**[1], **Siliang Tang**[1*], **Guoping Hu**[3], **Yueting Zhuang**[1] **& Xiang Ren**[4]
[1]Zhejiang University, [2]University of Illinois at Urbana Champaign
[3]iFLYTEK Research, [4]University of Southern California,
{chenbo123, xiaofeem, siliang, yzhuang}@zju.edu.cn,
xiaotao2@illinois.edu, gphu@iflytek.com, xiangren@usc.edu

Abstract

Recently, distant supervision has gained great success on Fine-grained Entity Typing (FET). Despite its efficiency in reducing manual labeling efforts, it also brings the challenge of dealing with false entity type labels, as distant supervision assigns labels in a context-agnostic manner. Existing works alleviated this issue with partial-label loss, but usually suffer from confirmation bias, which means the classifier fit a pseudo data distribution given by itself. In this work, we propose to regularize distantly supervised models with Compact Latent Space Clustering (CLSC) to bypass this problem and effectively utilize noisy data yet. Our proposed method first dynamically constructs a similarity graph of different entity mentions; infer the labels of noisy instances via label propagation. Based on the inferred labels, mention embeddings are updated accordingly to encourage entity mentions with close semantics to form a compact cluster in the embedding space, thus leading to better classification performance. Extensive experiments on standard benchmarks show that our CLSC model consistently outperforms state-of-the-art distantly supervised entity typing systems by a significant margin.

1 Introduction

Recent years have seen a surge of interests in fine-grained entity typing **(FET)** as it serves as an important cornerstone of several nature language processing tasks including relation extraction (Mintz et al., 2009), entity linking (Raiman and Raiman, 2018), and knowledge base completion (Dong et al., 2014). To reduce manual efforts in labelling training data, distant supervision (Mintz et al., 2009) has been widely adopted by recent FET systems. With the help of an external knowledge base (KB), an entity mention is first linked to an existing entity in KB, and then labeled with all possible types of the KB entity as supervision. However, despite its efficiency, distant supervision also brings the challenge of **out-of-context noise**, as it assigns labels in a context agnostic manner. Early works usually ignore such noise in supervision (Ling and Weld, 2012; Shimaoka et al., 2016), which dampens the performance of distantly supervised models.

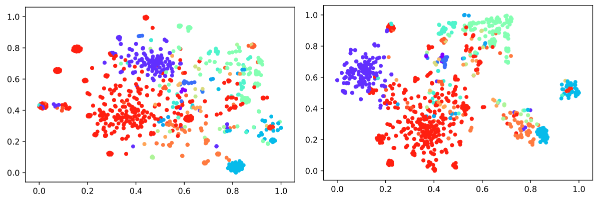


Figure 1: T-SNE visualization of the mention embeddings generated by NFETC (left) and CLSC (right) on the BBN dataset. Our model (CLSC) clearly groups mentions of the same type into a compact cluster.

Towards overcoming out-of-context noise, two lines of work have been proposed to distantly supervised FET. The first kind of work try to filter out noisy labels using heuristic rules (Gillick et al., 2014). However, such heuristic pruning significantly reduces the amount of training data, and thus cannot make full use of distantly annotated data. In contrast, the other thread of works try to incorporate such imperfect annotation by partial-label loss (**PLL**). The basic assumption is that, *for a noisy mention, the maximum score associated with its candidate types should be greater than the scores associated with any other non-candidate types* (Ren et al., 2016a; Abhishek et al., 2017; Xu and Barbosa, 2018). Despite their success, **PLL**-based models still suffer from ***Confirmation Bias*** by taking its own prediction as optimization objective in the next step. Specifically, given an entity mention, if the typing system selected a wrong

*Corresponding Author.

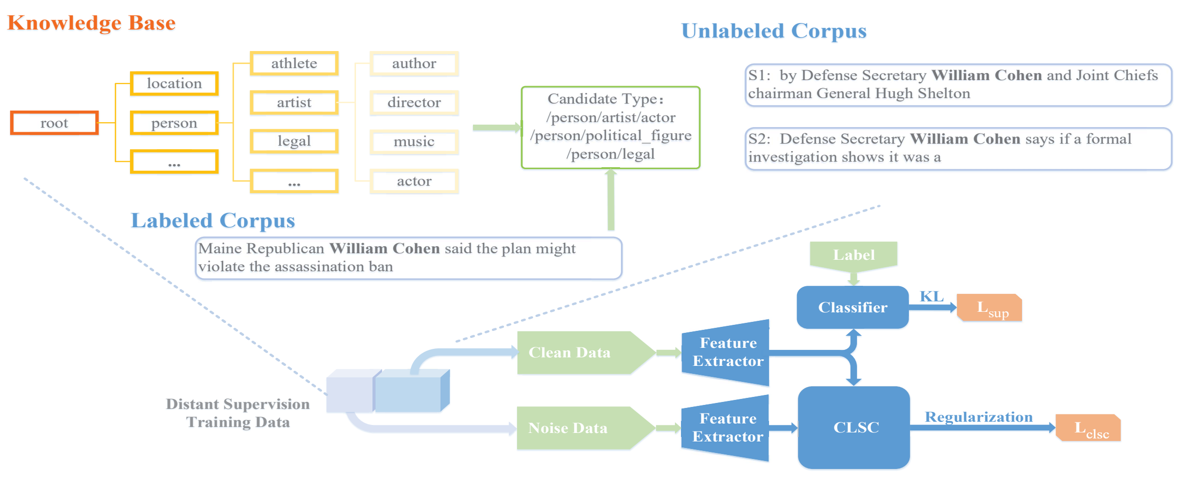


Figure 2: The overall framework of CLSC. We calculate classification loss only on clean data, while regularize the feature extractor with CLSC using both clean and noisy data.

type with the maximum score among all candidates, it will try to further maximize the score of the wrong type in following optimization epoches (in order to minimize **PLL**), thus amplifying the confirmation bias. Such bias starts from the early stage of training, when the typing model is still very suboptimal, and can accumulate in training process. Related discussion can be also found in the setting of semi-supervised learning (Lee et al., 2006; Laine and Aila, 2017; Tarvainen and Valpola, 2017).

In this paper, we propose a new method for distantly supervised fine-grained entity typing. Enlightened by (Kamnitsas et al., 2018), we propose to effectively utilize imperfect annotation as model regularization via **C**ompact **L**atent **S**pace **C**lustering **(CLSC)**. More specifically, our model encourages the feature extractor to group mentions of the same type as a compact cluster (dense region) in the representation space, which leads to better classification performance. For training data with noisy labels, instead of generating pseudo supervision by the typing model itself, we dynamically construct a similarity-weighted graph between clean and noisy mentions, and apply label propagation on the graph to help the formation of compact clusters. Figure 1 demonstrates the effectiveness of our method in clustering mentions of different types into dense regions. In contrast to **PLL**-based models, we do not force the model to fit pseudo supervision generated by itself, but only use noisy data as part of regularization for our feature extractor layer, thus avoiding bias accumulation.

Extensive experiments on standard benchmarks show that our method consistently outperforms state-of-the-art models. Further study reveals that, the advantage of our model over the competitors gets even more significant as the portion of noisy data rises.

2 Problem Definition

Fine-grained entity typing takes a corpus and an external knowledge base (KB) with a type hierarchy $\mathcal{Y}$ as input. Given an entity mention (i.e., a sequence of token spans representing an entity) in the corpus, our task is to uncover its corresponding type-path in $\mathcal{Y}$ based on the context.

By applying distant supervision, each mention is first linked to an existing entity in KB, and then labeled with all its possible types. Formally, a labeled corpus can be represented as triples $\mathcal{D} = \{(m_i, c_i, \mathcal{Y}_i)\}_{i=1}^{n}$, where m_i is the i-th mention, c_i is the context of m_i, $\mathcal{Y}_i$ is the set of candidate types of m_i. Note that types in $\mathcal{Y}_i$ can form one or more type paths. In addition, we denote all terminal (leaf) types of each type path in $\mathcal{Y}_i$ as the target type set $\mathcal{Y}_i^t$ (*e.g.*, for $\mathcal{Y}_i = \{artist, teacher, person\}$, $\mathcal{Y}_i^t = \{artist, teacher\}$). This setting is also adopted by (Xu and Barbosa, 2018).

As each entity in KB can have several type paths, *out-of-context* noise may exist when $\mathcal{Y}_i$ contains type paths that are irrelevant to m_i in context c_i. In this work, we argue triples where $\mathcal{Y}_i$ contains only one type path (i.e., $|\mathcal{Y}_i^t| = 1$) as **clean data**. Other triples are treated as **noisy data**, where $\mathcal{Y}_i$ contains both the true type path and irrel-

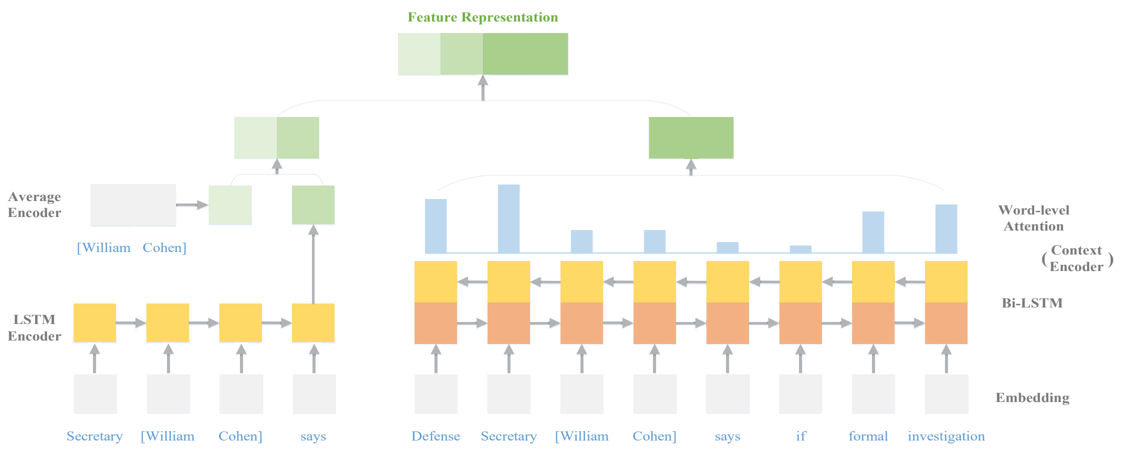


Figure 3: The architecture of feature extractor $z((m_i, c_i); \theta_z)$

evant type paths. Noisy data usually takes a considerable portion of the entire dataset. The major challenge for distantly supervised typing systems is to incorporate both clean and noisy data to train high-quality type classifiers.

3 The Proposed Approach

Overview. The basic assumptions of our idea are: (1) all mentions belong to the same type should be close to each other in the representation space because they should have similar context, (2) similar contexts lead to the same type. For clean data, we compact the representation space of the same type to comply (1). For noisy data, given assumption (2), we infer the their type distributions via label propagation and candidate types constrain.
Figure 2 shows the overall framework of the proposed method. Clean data is used to train classifier and feature extractor end-to-endly, while noisy data is only used in CLSC regularization. Formally, given a batch of samples $\{(m_i, c_i, \mathcal{Y}_i^t)\}_{i=1}^B$, we first convert each sample (m_i, c_i) into a real-valued vector z_i via a feature extractor $z((m_i, c_i); \theta_z)$ parameterized by θ_z. Then a type classifier $g(z_i; \theta_g)$ parameterized by θ_g gives the posterior $P(y|z_i; \theta_g)$. By incorporating CLSC regularization in the objective function, we encourage the feature extractor z to group mentions of the same type into a compact cluster, which facilitates classification as is shown in Figure 1. Noisy data enhances the formation of compact clusters with the help of label propagation.

3.1 Feature Extractor

Figure 3 illustrates our feature extractor. For fair comparison, we adopt the same feature extraction pipeline as used in (Xu and Barbosa, 2018). The feature extractor is composed of an embedding layer and two encoders which encode mentions and contexts respectively.
Embedding Layer: The output of this layer is

a concatenation of word embedding and word position embedding. We use the popular 300-dimensional word embedding supplied by (Pennington et al., 2014) to capture the semantic information and random initialized position embedding (Zeng et al., 2014) to acquire information about the relation between words and the mentions.

Formally, Given a word embedding matrix W_{word} of shape $d_w \times |V|$, where V is the vocabulary and d_w is the size of word embedding, each column of W_{word} represents a specific word w in V. We map each word w_j in (m_i, c_i) to a word embedding $\mathbf{w_j^d} \in R^{d_w}$. Analogously, we get the word position embedding $\mathbf{w}_j^p \in R^{d_p}$ of each word according to the relative distance between the word and the mention, we only use a fixed length context here. The final embedding of the j-th word is $\mathbf{w_j^E} = [\mathbf{w_j^d}, \mathbf{w_j^P}]$.

Mention Encoder: To capture lexical level information of mentions, an averaging mention encoder and a LSTM mention encoder (Hochreiter and Schmidhuber, 1997) is applied to encode mentions. Given $m_i = (w_s, w_{s+1}, \cdots, w_e)$, the averaging mention representation $r_{a_i} \in R^{d_w}$ is :

$$r_{a_i} = \frac{1}{e-s+1} \sum_{j=s}^{e} \mathbf{w_j^d} \quad (1)$$

By applying a LSTM over an extended mention $(w_{s-1}, w_s, w_{s+1}, \cdots, w_e, w_{e+1})$, we get a sequence $(h_{s-1}, h_s, h_{s+1}, \cdots, h_e, h_{e+1})$. We use h_{e+1} as LSTM mention representation $r_{l_i} \in R^{d_l}$. The final mention representation is $r_{m_i} = [r_{a_i}, r_{l_i}] \in R^{d_w+d_l}$.

Context Encoder: A bidirectional LSTM with d_l hidden units is employed to encode embedding sequence $(\mathbf{w^E_{s-W}}, \mathbf{w^E_{s-W+1}}, \cdots, \mathbf{w^E_{e+W}})$:

$$\begin{aligned}\overrightarrow{h_j} &= LSTM(\overrightarrow{h_{j-1}}, \mathbf{w^E_{j-1}}) \\ \overleftarrow{h_j} &= LSTM(\overleftarrow{h_{j-1}}, \mathbf{w^E_{j-1}}) \\ h_j &= [\overrightarrow{h_j} \oplus \overleftarrow{h_j}]\end{aligned} \quad (2)$$

where $\oplus$ denotes element-wise plus. Then, the word-level attention mechanism computes a score $\beta_{i,j}$ over different word j in the context c_i to get the final context representation r_{c_i}:

$$\begin{aligned}\alpha_j &= w^T tanh(h_j) \\ \beta_{i,j} &= \frac{exp(\alpha_j)}{\sum_k exp(\alpha_k)} \\ r_{c_i} &= \sum_j \beta_{i,j} h_{i,j}\end{aligned} \quad (3)$$

We use $r_i = [r_{m_i}, r_{c_i}] \in R^{d_z} = R^{d_w+d_l+d_l}$ as the feature representation of (m_i, c_i) and use a Neural Networks q over r_i to get the feature vector z_i. q has n layers with h_n hidden units and use ReLu activation.

3.2 Compact Latent Space Clustering for Distant Supervision

The overview of CLSC regularization is exhibited in Figure 4, which includes three steps: dynamic graph construction (Figure 4c), label propagation (Figure 4d, e) and Markov chains (Figure 4g). The idea of compact clustering for semi-supervised learning is first proposed by (Kamnitsas et al., 2018). The basic idea is to encourage mentions of the same type to be clustered into a dense region in the embedding space. We introduce more details of CLSC for distantly supervised FET in following sections.

Dynamic Graph Construction: We start by creating a fully connected graph G over the batch of samples $\mathbf{Z} = \{z_i\}_{i=1}^{B}$, as shown in Figure 4c[1]. Each node of G is a feature representation z_i, while the distance between nodes is defined by a scaled dot-product distance function (Vaswani et al., 2017):

$$\begin{aligned}A_{ij} &= exp(\frac{z_i^T z_j}{\sqrt{d_z}}), \forall z_i, z_j \in \mathbf{Z} \\ A &= exp(\frac{Z^T Z}{\sqrt{d_z}})\end{aligned} \quad (4)$$

Each entry A_{ij} measures the similarity between z_i and z_j, $A \in R^{B \times B}$ can be viewed as the weighted adjacency matrix of G.

Label Propagation: The end goal of CLSC is to cluster mentions of the same type to a dense region. For mentions which have more than one labeled types, we apply label propagation (**LP**) on G to estimate their type distribution. Formally, we denote $\mathbf{\Phi} \in R^{B \times K}$ as the label propagation posterior of a training batch.

The original label propagation proposed by (Zhu and Ghahramani, 2002) uses a transition matrix H to model the probability of a node i propagating its type posterior $\phi_i = P(y_i|x_i) \in R^K$ to the other nodes. Each entry of the transition matrix $H \in R^{B \times B}$ is defined as:

$$H_{ij} = A_{ij} / \sum_b A_{ib} \quad (5)$$

The original label propagation algorithm is defined as:

1. Propagate the label by transition matrix H, $\mathbf{\Phi}^{(t+1)} = H\mathbf{\Phi}^{(t)}$
2. Clamp the labeled data to their true labels. Repeat from step 1 until $\mathbf{\Phi}$ converges

In this work $\Phi^{(0)}$ is randomly initialized[2]. Unlike unlabeled data in semi-supervised learning, distantly labeled mentions in FET have a limited set of candidate types. Based on this observation, We assume that (m_i, c_i) can only transmit and receive probability of types in $\mathcal{Y}_i^t$ no matter it is noisy data or clean data. Formally, define a $B \times K$ indicator matrix $M \in R^{B \times K}$, where $M_{ij} = 1$ if type j in $\mathcal{Y}_i^t$ otherwise 0, where B is the batch size and K

[1]$\mathbf{Z} = \{z_i\}_{i=1}^{B}$ is a small subsample of the entire data, we didn't observe significant performance gain when the batch size increases.

[2]We also explored other initialization (e.g. uniform initialization), but found no essential performance difference between different initialization setups.

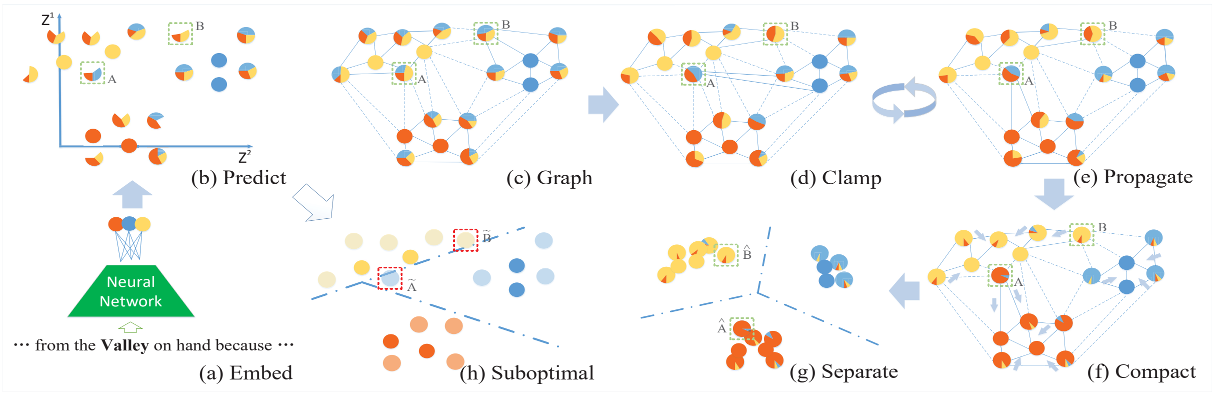


Figure 4: A demonstration of the CLSC process. (a) represents the feature extraction step; (b)→(h) shows the traditional type classification process (each color represents one candidate type), where suboptimal classifiers make predictions for each mention and misclassifies A into the Blue type; (c)→(d)→(e)→(f)→(g) demonstrates the process of CLSC as described in Section 3. Through label propagation and compact clustering, our model is able to group mentions of the same type into a dense region and leaves clear separation boundaries in sparse regions.

is the number of types. Our clamping step relies on M as is shown in Figure 4d:

$$\Phi_{ij}^{(t+1)} \leftarrow \Phi_{ij}^{(t+1)} M_{ij} / \sum_k \Phi_{ik}^{(t+1)} M_{ik} \quad (6)$$

For convenience, we iterate through these two steps S_{lp} times, S_{lp} is a hyperparameter.

Compact Clustering: The **LP** posterior $\mathbf{\Phi} = \mathbf{\Phi}^{(S_{lp}+1)}$ is used to judge the label agreement between samples. In the desired optimal state, transition probabilities between samples should be uniform inside the same class, while be zero between different classes. Based on this assumption, the desirable transition matrix $T \in R^{B \times B}$ is defined as:

$$T_{ij} = \sum_{k=1}^{K} \Phi_{ik} \frac{\Phi_{jk}}{m_k}, m_k = \sum_{b=1}^{B} \Phi_{bk} \quad (7)$$

m_k is a normalization term for class k. Transition matrix H derived from $z((m_i, c_i); \theta_z)$ should be in keeping with T. Thus we minimize the cross entropy between T and H:

$$\mathcal{L}_{1-step} = -\frac{1}{B^2} \sum_{i=1}^{B} \sum_{j=1}^{B} T_{ij} log(H_{ij}) \quad (8)$$

For instance, if T_{ij} is close to 1, H_{ij} needs to be bigger, which results in the growth of A_{ij} and finally optimize θ_z (Eq.4). The loss $\mathcal{L}_{1-step}$ has largely described the regularization we use in $z((m_i, c_i); \theta_z)$ for compression clustering.

In order to keep the structure of existing clusters, (Kamnitsas et al., 2018) proposed an extension of $\mathcal{L}_{1-step}$ to the case of **Markov chains** with multiple transitions between samples, which should remain within a single class. The extension maximizes probability of paths that only traverse among samples belong to one class. Define $E \in R_{B \times B}$ as:

$$E = \mathbf{\Phi}^T \mathbf{\Phi} \quad (9)$$

E_{ij} measures the label similarities between z_i and z_j, which is used to mask the transition between different clusters. The extension is given by:

$$\begin{aligned} H^{(1)} &= H \\ H^{(s)} &= (H \odot E)^{(s-1)} H \\ &= (H \odot E) H^{(s-1)}, \end{aligned} \quad (10)$$

where $\odot$ is Hadamard Product, and $H_{ij}^{(s)}$ is the probability of a Markov process to transit from node i to node j after $s-1$ steps within the same class. The extended loss function models paths of different length s between samples on the graph:

$$\mathcal{L}_{clsc} = -\frac{1}{S_m} \frac{1}{B^2} \sum_{s=1}^{S_m} \sum_{i=1}^{B} \sum_{j=1}^{B} T_{ij} log(H_{ij}^{(s)}). \quad (11)$$

For $S_m = 1$, $\mathcal{L}_{clsc} = \mathcal{L}_{1-step}$. By minimizing the cross entropy between T and $H^{(s)}$ (Eq.11), $\mathcal{L}_{clsc}$ compact paths of different length between samples within the same class. Here, S_m is a hyper-parameter to control the maximum length of Markov chain. $\mathcal{L}_{clsc}$ is added to the final objective function as regularization to encourage compact cluttering.

3.3 Overall Objective

Given the representation of a mention, the type posterior is given by a standard softmax classifier parameterized by θ_g:

$$P(\hat{y}_i|z_i;\theta_g) = softmax(W_c z_i + b_c), \qquad (12)$$

where $W_c \in R^{K \times d_z}$ is a parameter matrix, $b \in R^K$ is the bias vector, where K is the number of types. The predicted type is then given by $\hat{t_i} = argmax_{y_i} P(\hat{y}_i|z_i;\theta_g)$.

Our loss function consists of two parts. $\mathcal{L}_{sup}$ is supervision loss defined by KL divergence:

$$L_{sup} = -\frac{1}{B_c}\sum_{i=1}^{B_c}\sum_{k=1}^{K} y_{ik} log(P(y_i|z_i;\theta_g))_k \qquad (13)$$

Here B_c is the number of clean data in a training batch, K is the number of target types. The regularization term is given by $\mathcal{L}_{clsc}$. Hence, the overall loss function is:

$$\mathcal{L}_{final} = \mathcal{L}_{sup} + \lambda_{clsc} \times \mathcal{L}_{clsc} \qquad (14)$$

λ_{clsc} is a hyper parameter to control the influence of CLSC.

4 Experiments

4.1 Dataset

We evaluate our method on two standard benchmarks: OntoNotes and BBN:

- **OntoNotes:** The OntoNotes dataset is composed of sentences from the Newswire part of OntoNotes corpus (Weischedel et al., 2013). (Gillick et al., 2014) annotated the training part with the aid of DBpedia spotlight (Daiber et al., 2013), while the test data is manually annotated.
- **BBN:** The BBN dataset is composed of sentences from Wall Street Journal articles and is manually annotated by (Weischedel and Brunstein, 2005). (Ren et al., 2016a) regenerated the training corpus via distant supervision.

In this work we use the preprocessed datasets provided by (Abhishek et al., 2017; Xu and Barbosa, 2018). Table 2 shows detailed statistics of the datasets.

4.2 Compared Methods

We compare the proposed method with several state-of-the-art FET systems[3]:

- **Attentive** (Shimaoka et al., 2016) uses an attention based feature extractor and doesn't distinguish clean from noisy data;
- **AFET** (Ren et al., 2016a) trains label embedding with partial label loss;
- **AAA** (Abhishek et al., 2017) learns joint representation of mentions and type labels;
- **PLE+HYENA/FIGER** (Ren et al., 2016b) proposes heterogeneous partial-label embedding for label noise reduction to boost typing systems. We compare two PLE models with HYENA (Yogatama et al., 2015) and FIGER (Ling and Weld, 2012) as the base typing system respectively;
- **NFETC** (Xu and Barbosa, 2018) trains neural fine-grained typing system with hierarchy-aware loss. We compare the performance of the NFETC model with two different loss functions: partial-label loss and **PLL**+hierarchical loss. We denote the two variants as $\textbf{NFETC}$ and $\textbf{NFETC}_{hier}$ respectively;
- **NFETC-CLSC** is the proposed model in this work. We use the NFETC model as our base model, based on which we apply Compact Latent Space Clustering Regularization as described in Section 3.2; Similarly, we report results produced by using both KL-divergense-based loss (**NFETC-CLSC**) and **KL**+hierarchical loss (**NFETC-CLSC**$_{hier}$).

4.3 Evaluation Settings

For evaluation metrics, we adopt strict accuracy, loose macro, and loose micro F-scores widely used in the FET task (Ling and Weld, 2012). To fine tuning the hyper-parameters, we randomly sampled 10% of the test set as a development set for both datasets. With the fine-tuned hyper-parameter as mentioned in 4.4, we run the model five times and report the average strict accuracy, macro F1 and micro F1 on the test set.

[3]The baselines result are reported on (Abhishek et al., 2017; Xu and Barbosa, 2018) in addition to performance of NFETC on BBN, we search the hyper parameters for it. (Xu and Barbosa, 2018) didn't report the results on BBN

Method		OntoNotes			BBN		
		Strict Acc.	Macro F1	Micro F1	Strict Acc.	Macro F1	Micro F1
AFET (Ren et al., 2016a)		55.3	71.2	64.6	68.3	74.4	74.7
AAA (Abhishek et al., 2017)		52.2	68.5	63.3	65.5	73.6	75.2
Attentive (Shimaoka et al., 2016)		51.7	71.0	64.91	48.4	73.2	72.4
PLE+HYENA (Ren et al., 2016b)		54.6	69.2	62.5	69.2	73.1	73.2
PLE+FIGER (Ren et al., 2016b)		57.2	71.5	66.1	68.5	77.7	75.0
NFETC	$clean$	54.4±0.3	71.5±0.4	64.9±0.3	71.2±0.2	77.1±0.3	76.9±0.3
	$+noisy$	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$_{hier}$	$clean$	59.6±0.2	76.1±0.2	69.7±0.2	70.3±0.3	76.8±0.3	76.6±0.2
	$+noisy$	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	$clean$	59.1±0.4	75.3±0.3	69.1±0.3	73.0±0.3	79.0±0.3	78.8±0.3
	$+noisy$	59.6±0.2	75.5±0.4	69.3±0.4	**74.7±0.3**	**80.7±0.2**	**80.5±0.2**
NFETC-CLSC$_{hier}$	$clean$	61.5±0.3	77.4±0.3	71.4±0.4	70.5±0.2	78.2±0.2	78.0±0.2
	$+noisy$	**62.8±0.3**	**77.8±0.4**	**72.0±0.4**	71.9±0.3	79.8±0.4	79.5±0.3

Table 1: Performance comparision of FET systems on the two datasets.

	OntoNotes	BBN
#types	89	47
Max hierarchy depth	3	2
#mentions-train	253241	86078
#mentions-test	8963	12845
%clean mentions-train	73.13	75.92
%clean mentions-test	94.00	100
Average $\|\mathcal{Y}_i^t\|$	1.40	1.26

Table 2: Detailed statistics of the two datasets.

4.4 Hyper Parameters

We search the hyper parameter of Ontonotes and BBN respectively via Hyperopt proposed by (Bergstra et al., 2013). Hyper parameters are shown in **Appendix A**. We optimize the model via Adam Optimizer. The full hyper parameters includes the learning rate lr, the dimension d_p of word position embedding, the dimension d_l of the mention encoder's output (equal to the dimension of the context encoder's ourput), the input dropout keep probability p_i and output dropout keep probability p_o for LSTM layers (in context encoder and LSTM mention encoder), the L2 regularization parameter λ, the factor of hierarchical loss normalization α ($\alpha > 0$ means use the normalization), BN (whether using Batch normalization), the max step S_{lp} of the label propagation, the max length S_m of Markov chain, the influence parameter λ_{clsc} of CLSC, the batch size B, the number n of hidden layers in q and the number h_n of hidden units of the hidden layers. We implement all models using Tensorflow[4].

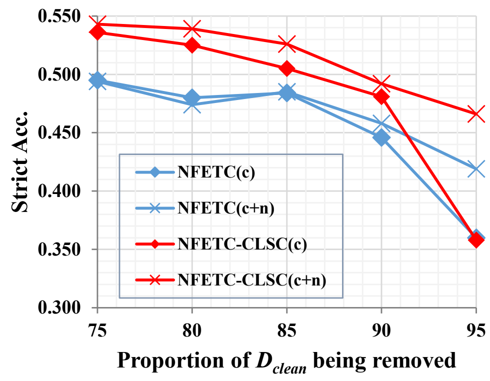
4.5 Performance comparison and analysis

Table 1 shows performance comparison between the proposed CLSC model and state-of-the-art FET systems. On both benchmarks, the CLSC model achieves the best performance in all three metrics. When focusing on the comparison between **NFETC** and CLSC, we have following observation:

- Compact Latent Space Clustering shows its effectiveness on both clean data and noisy data. By applying CLSC regularization on the basic **NFETC** model, we observe consistent and significant performance boost;
- Hierarchical-aware loss shows significant advantage on the OntoNotes dataset, while showing insignificant performance boost on the BBN dataset. This is due to different distribution of labels on the test set. The proportion of terminal types of the test set is 69% for the BBN dataset, while is only 33% on the OntoNotes dataset. Thus, applying hierarchical-aware loss on the BBN dataset brings little improvement;
- Both algorithms are able to utilize noisy data to improve performance, so we would like to further study their performance in different noisy scenarios in following discussions.

[4]The code for experiments is available at https://github.com/herbertchen1/NFETC-CLSC

4.6 How robust are the methods to the proportion of noisy data?



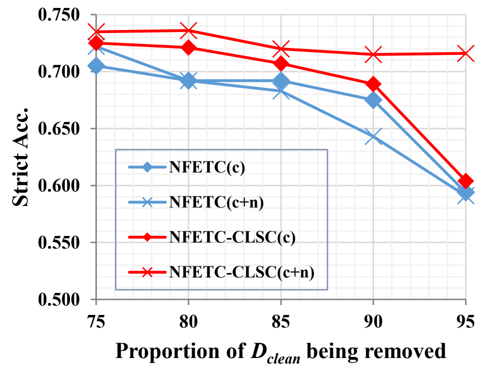


Figure 5: Performance comparison between NFETC-CLSC and NFETC by removing 75%-95% clean data.

By principle, with sufficient amount of clean training data, most typing systems can achieve satisfying performance. To further study the robustness of the methods to label noise, we compare their performance with the presence of $25\%, 20\%, 15\%, 10\%$ and 5% clean training data and all noisy training data. Figure 5 shows the performance curves as the proportion of clean data drops. As it reveals, the CLSC model consistently wins in the comparison. The advantage is especially clear on the BBN dataset, which offers less amount of training data. Note that, with only 27.9% of training data (when only leaving 5% clean data) on the BBN dataset, the CLSC model yield a comparable result with the **NFETC** model trained on full data. This comparison clearly shows the superiority of our approach in the effectiveness of utilizing noisy data.

4.7 Ablation: Do Markov Chains improve typing performance?

Table 3 shows the performance of CLSC with one-step transition ($\mathcal{L}_{1-step}$) and with Markov Chains ($\mathcal{L}_{clsc}$) as described in Section 3.2. Results show that the use of Markov Chains does bring improvement to the overall performance, which is consistent with the model intuition.

5 Related Work

Named entity Recognition (NER) has been excavated for a long time (Collins and Singer, 1999; Manning et al., 2014), which classifies coarse-grained types (e.g. person, location). Recently, (Nagesh and Surdeanu, 2018a,b) applied ladder network (Rasmus et al., 2015) to coarse-grained entity classification in a semi-supervised learning fashion. (Ling and Weld, 2012) proposed Fine-Grained Entity Recognition (FET). They used distant supervision to get training corpus for FET. Embedding techniques was applied to learn feature representations since (Yogatama et al., 2015; Dong et al., 2015). (Shimaoka et al., 2016) introduced attention mechanism for FET to capture informative words. (Xin et al., 2018a) used the TransE entity embeddings (Bordes et al., 2013) as the query vector of attention.
Early works ignore the out-of-context noise, (Gillick et al., 2014) proposed context dependent FET and use three heuristics to clean the noisy labels with the side effect of losing training data. To utilize noisy data, (Ren et al., 2016a) distinguished the loss function of noisy data from clean data via partial label loss (**PLL**). (Abhishek et al., 2017; Xu and Barbosa, 2018) proposed variants of **PLL**, which still suffer from confirmation bias. (Xu and Barbosa, 2018) proposed hierarchical loss to handle over-specific noise. On top of **AFET**, (Ren et al., 2016b) proposed a method **PLE** to reduce the label noise, which lead to a great success in FET. Because label noise reduction is separated from the learning of FET, there might be error propagation problem. Recently, (Xin et al., 2018b) proposed utilizing a pretrained language model measures the compatibility between context and type names, and use it to repel the interference of noisy labels. However, the compatibility got by language model may not be right and type information is defined by corpus and annotation guidelines rather than type names as is mentioned in (Azad et al., 2018). In addition, there are some work about entity-level typing which aim to figure out the types of entities in KB (Yaghoobzadeh and Schütze, 2015; Jin et al., 2018).

	Strict Acc.
CLSC(c)($\mathcal{L}_{1-step}$)	72.0±0.1
CLSC(c)($\mathcal{L}_{clsc}$)	73.0±0.3
CLSC(c+n)($\mathcal{L}_{1-step}$)	73.0±0.1
CLSC(c+n)($\mathcal{L}_{clsc}$)	74.7±0.3

Table 3: The comparison of $\mathcal{L}_{1-step}$ and $\mathcal{L}_{clsc}$ on BBN.

6 Conclusion

In this paper, we propose a new method for distantly supervised fine-grained entity typing, which leverages imperfect annotations as model regularization via Compact Latent Space Clustering (CLSC). Experiments on two standard benchmarks demonstrate that our method consistently outperforms state-of-the-art models. Further study reveals our method is more robust than the former state-of-the-art approach as the portion of noisy data rises. The proposed method is general for other tasks with imperfect annotation. As a part of future investigation, we plan to apply the approach to other distantly supervised tasks, such as relation extraction.

7 Acknowledgments

This work has been supported in part by NSFC (No.61751209, U1611461), Zhejiang University-iFLYTEK Joint Research Center, Chinese Knowledge Center of Engineering Science and Technology (CKCEST), Engineering Research Center of Digital Library, Ministry of Education. Xiang Ren's research has been supported in part by National Science Foundation SMA 18-29268.

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 797–807.

Amar Prakash Azad, Balaji Ganesan, Ashish Anand, Amit Awekar, et al. 2018. A unified labeling approach by pooling diverse datasets for entity typing. *arXiv preprint arXiv:1810.08782*.

James Bergstra, Dan Yamins, and David D Cox. 2013. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in Science Conference*, pages 13–20. Citeseer.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N Mendes. 2013. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, pages 121–124. ACM.

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.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM.

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

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

Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. 2018. Attributed and predictive entity embedding for fine-grained entity typing in knowledge bases. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 282–292.

Konstantinos Kamnitsas, Daniel Castro, Loic Le Folgoc, Ian Walker, Ryutaro Tanno, Daniel Rueckert, Ben Glocker, Antonio Criminisi, and Aditya Nori. 2018. Semi-supervised learning via compact latent space clustering. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2459–2468, Stockholmsmssan, Stockholm Sweden. PMLR.

Samuli Laine and Timo Aila. 2017. Temporal ensembling for semi-supervised learning. In *Proc. International Conference on Learning Representations (ICLR)*.

Changki Lee, Yi-Gyu Hwang, Hyo-Jung Oh, Soojong Lim, Jeong Heo, Chung-Hee Lee, Hyeon-Jin Kim, Ji-Hyun Wang, and Myung-Gil Jang. 2006. Fine-grained named entity recognition using conditional random fields for question answering. In *Asia Information Retrieval Symposium*, pages 581–587. Springer.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *AAAI*, volume 12, pages 94–100.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Ajay Nagesh and Mihai Surdeanu. 2018a. An exploration of three lightly-supervised representation learning approaches for named entity classification. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2312–2324.

Ajay Nagesh and Mihai Surdeanu. 2018b. Keep your bearings: Lightly-supervised information extraction with ladder networks that avoids semantic drift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 352–358.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5406–5413.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. 2015. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554.

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 *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 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 *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1825–1834. ACM.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 69–74.

Antti Tarvainen and Harri Valpola. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

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*.

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

Ji Xin, Hao Zhu, Xu Han, Zhiyuan Liu, and Maosong Sun. 2018b. Put it back: Entity typing with language model enhancement. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 993–998.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 16–25.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2015. Corpus-level fine-grained entity typing using contextual information. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 715–725.

Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 291–296.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

Xiaojin Zhu and Zoubin Ghahramani. 2002. Learning from labeled and unlabeled data with label propagation.

A Hyper parameters

	Ont.(C)	**BBN(C)**	**BBN(N)**
lr	0.0006	0.0007	0.0007
d_p	70	40	20
d_l	1000	1000	240
p_i	0.7	0.3	0.5
p_o	0.6	1.0	0.4
λ	0.0000	0.0000	0.0002
α	0.25/0.0	0.4/0.0	0.4/0.0
BN	FALSE	FALSE	TRUE
S_{lp}	200	200	-
S_m	8	12	-
λ_{clsc}	2.0	1.5	-
B	512	512	512
n	2	1	-
h_n	700	560	-

Table 4: Hyper parameters of our experiments: (C) denotes CLSC, (N) denotes the hyper parameter is used for NFETC.