# Label Embedding for Zero-shot Fine-grained Named Entity Typing

**Yukun Ma**[1,2], **Erik Cambria**[1,2], **Sa Gao**[1,2]
[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore
[2]Rolls-Royce@NTU Corporate Lab, Nanyang Technological University, Singapore
mayu0010@e.ntu.edu.sg, cambria@ntu.edu.sg, gaos0011@e.ntu.edu.sg

## Abstract

Named entity typing is the task of detecting the types of a named entity in context. For instance, given "Eric is giving a presentation", our goal is to infer that 'Eric' is a speaker or a presenter and a person. Existing approaches to named entity typing cannot work with a growing type set and fails to recognize entity mentions of unseen types. In this paper, we present a label embedding method that incorporates prototypical and hierarchical information to learn pre-trained label embeddings. In addition, we adapt a zero-shot framework that can predict both seen and previously unseen entity types. We perform evaluation on three benchmark datasets with two settings: 1) few-shots recognition where all types are covered by the training set; and 2) zero-shot recognition where fine-grained types are assumed absent from training set. Results show that prior knowledge encoded using our label embedding methods can significantly boost the performance of classification for both cases.

## 1 Introduction

Named entity typing (NET) is the task of inferring types of named entity mentions in text. NET is a useful pre-processing step for many natural language processing (NLP) tasks, e.g., auto-categorization and sentiment analysis. Named entity linking, for instance, can use NET to refine entity candidates of a given mention (Ling and Weld, 2012). Besides, NET is capable of supporting applications based on a deeper understanding of natural language, e.g., knowledge completion (Dong et al., 2014) and question answering (Lin et al., 2012; Fader et al., 2014). Standard NET approaches or sometime known as named entity recognition (Chinchor and Robinson, 1997; Tjong Kim Sang and De Meulder, 2003; Doddington et al., 2004) are concerned with coarse-grained types (e.g, person, location, organization) that are flat in structure. In comparison, fine-grained named entity typing (FNET) (Ling and Weld, 2012), which has been studied as an extension of standard NET task, uses a tree-structured taxonomy including not only coarse-grained types but also fine-grained types of named entities. For instance, given "[*Intel*] *said that over the past decade*", standard NET only classifies *Intel* as *organization*, whereas FNET further classifies it as *organization/corporation*.

FNET is faced with two major challenges: growing type set and label noises. Since the type hierarchy of entities is typically built from knowledge bases such as DBpedia, which is regularly updated with new types (especially fine-grained types) and entities, it is natural to assume that the type hierarchy is growing rather than fixed over time. However, current FNET systems are impeded from handling a growing type set for that information learned from training set cannot be transferred to unseen types. Another problem with FNET is that the weakly supervised tagging process used for automatically generating labeled data inevitably introduces label noises. Current solutions rely on heuristic rules (Gillick et al., 2014) or embedding method (Ren et al., 2016) to remove noises prior to training the multi-label classifier. In order to address these two problems at the same time, we propose a simple yet effective method for learning prototype-driven label embeddings that works for both seen and unseen types and is robust to the label





*Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*,
pages 171–180, Osaka, Japan, December 11-17 2016.

noises. Another contribution of this work is that we combine prototypical and hierarchical information for learning label embeddings.

The remainder of this paper is organized as follows: Section 2 proposes a survey of prior works related to FNET; Section 3 introduces the embedding-based FNET method and its zero-shot extension; Section 4 describes our label embedding method; Section 5 illustrates experiments and analysis for both few-shot and zero-shot settings; finally, Section 6 concludes the paper and discusses future work.

## 2 Related Work

There is little related work specifically on zero-shot FNET but several research lines are considered related to this work: fine-grained named entity recognition, prototype-driven learning, and multi-label classification models based on embeddings. As FNET works with a much larger type set as compared with standard NET, it becomes difficult to have a sufficient training set for every type when relying on manual annotation. Instead, training data can be automatically generated from semi-structural data such as Wikipedia pages (Ling and Weld, 2012). Consequently, a single supervised classifier (Ling and Weld, 2012; Yogatama et al., 2015) or a series of classifiers (Yosef et al., 2012) are trained on this auto-annotated training set. This auto-annotating practice has been followed by later works on FNET (Yosef et al., 2012; Yogatama et al., 2015; Ren et al., 2016). However, since the automated tagging process is not accurate all the time, a number of noisy labels are then propagated to supervised training and affect the performance negatively.

The starting point of this work is the embedding method, WSABIE (Weston et al., 2011), adapted by (Yogatama et al., 2015) to FNET. WSABIE maps input features and labels to a joint space, where information is shared among correlated labels. However, the joint embedding method still suffers from label noises which have negative impacts on the learning of joint embeddings. In addition, since the labeled training set is the only source used for learning label embeddings, WSABIE cannot learn label embeddings for unseen types. DeViSE (Frome et al., 2013) is proposed for annotating image with words or phrases. As in such case, labels are natural words, e.g., fruit, that can be found in textual data, Skip-gram word embeddings (Mikolov et al., 2013) learned from a large text corpus are directly used for representing labels. In addition to label itself, prior works have also tried to learn label embeddings from side information such as attributes (Akata et al., 2013), manually-written descriptions (Larochelle et al., 2008), taxonomy of types (Weinberger and Chapelle, 2009; Akata et al., 2013; Akata et al., 2015), and so on.

Another related line of research is prototype-driven learning. (Haghighi and Klein, 2006) presented a sequence labeling model using prototypes as features and has tested the model on NLP tasks such as part-of-speech (POS) tagging. Prototype-based features (Guo et al., 2014) are then adapted for coarse-grained named entity recognition task. Even though we select prototypes in the same way as (Guo et al., 2014), we use prototypes in a very different manner: we consider prototypes as the basis for representing labels, whereas prototypes are mainly used as additional features in prior works (Haghighi and Klein, 2006; Guo et al., 2014). In other words, prototypes are previously used on the input side, while we use them on the label side.

## 3 Embedding Methods for FNET

In this section, we introduce the embedding method for FNET proposed by (Yogatama et al., 2015) and its extension to zero-shot entity typing.

### 3.1 Joint Embedding Model

Each entity mention $m$ is represented as a feature vector $x \in \mathbb{R}^V$; and each label $y \in Y$ is a one-hot vector, where $Y$ is the set of true labels associated with $x$. $\bar{Y}$ denotes the set of false labels of the given entity mention. The bi-linear scoring function for a given pair of $x$ and $y$ is defined as follows:

$$f(x, y, W) = x'Wy,$$

where $W \in \mathbb{R}^{M \times N}$ matrix with $M$ the dimension of feature vector and $N$ the number of types.

Instead of using a single compatibility matrix, WSABIE (Weston et al., 2011; Yogatama et al., 2015) considers an alternate low-rank decomposition of $W$, i.e., $W = A^\top B$, in order to reduce the number of parameters. WSABIE rewrites the scoring function as

$$f(x, y, A, B) = \phi(x, A) \cdot \theta(y, B) = x' A^\top B y,$$

which maps feature vector $x$ and label vector $y$ to a joint space. Note that it actually defines feature embeddings and label embeddings as

$$\phi(x, A) : x \rightarrow Ax,$$
$$\theta(y, B) : y \rightarrow By,$$

where $A \in \mathbb{R}^{D \times M}$ and $B \in \mathbb{R}^{D \times N}$ are matrices corresponding to lookup tables of feature embeddings and label embeddings, respectively. The embedding matrices $A$ and $B$ are the only parameters to be learned from supervised training process. In (Weston et al., 2011), the learning is formulated as a learning-to-rank problem using weighted approximate-rank pairwise (WARP) loss,

$$\sum_{y \in Y} \sum_{y' \in \bar{Y}} L(rank(x, y)) \max(1 - f(x, y, A, B) + f(x, y', A, B), 0),$$

where the ranking function $rank(x, y) = \sum_{y' \in \bar{Y}} \mathbb{I}(1 + f(x, y', A, B) > f(x, y, A, B))$, and $L(k) = \sum_{i=1}^{k} \frac{1}{i}$ which maps the ranking to a floating-point weight.

### 3.2 Zero-shot FNET Extension

A zero-shot extension of above WSABIE method can be done by introducing pre-trained label embeddings into the framework. The pre-trained label embeddings are learned from additional resources, e.g., text corpora, to encode semantic relation and dependency between labels. Similar to (Akata et al., 2013), we use two different methods for incorporating pre-trained label embeddings. The first one is to fully trust pre-trained label embeddings. Namely, we fix $B$ as the pre-trained $\tilde{B}$ and only learn $A$ in an iterative process. The second method is to use pre-trained label embedding as prior knowledge while adjusting both $A$ and $B$ according to the labeled data, i.e., adding a regularizer to the WARP loss function,

$$\sum_{y \in Y} \sum_{y' \in \bar{Y}} L(rank(x, y)) \max(1 - f(x, y, A, B) + f(x, y', A, B), 0) + \lambda ||B - \tilde{B}||_F^2,$$

where $|| \cdot ||_F$ is the Frobenius norm, and $\lambda$ is the trade-off parameter.

## 4 Methods

### 4.1 Prototype-driven Label Embedding

Joint embedding methods such as WSABIE learn label embeddings from the whole training set including noisy labeled instances resulting from weak supervision. It is inevitable that the resulting label embeddings are affected by noisy labels and fail to accurately capture the semantic correlation between types. Another issue is that zero-shot frameworks such as DeViSE are not directly applicable to FNET as conceptually complex types, e.g, *GPE* (Geo-political Entity) cannot be simply mapped to a single natural word or phrase.

To address this issue, we propose a simple yet effective solution which is referred to as prototype-driven label embedding (ProtoLE), and henceforth we use $\tilde{B}^P$ to denote the label embedding matrix learned by ProtoLE. The first step is to learn a set of prototypes for each type in the type set. ProtoLE does not fully rely on training data to generate label embeddings. Instead, it selects a subset of entity mentions as the prototypes of each type. These prototypes are less ambiguous and noisy compared to the rest of the full set.

Even though it is already far less labor-intensive to manually select prototypes than annotating entity mentions one by one, we consider an alternative automated process using Normalized Point-wise Mutual Information (NPMI) as the particular criterion for prototype selection. The NPMI between a label and an entity mention is computed as:

$$\texttt{NPMI}(y, m) = \frac{\texttt{PMI}(y, m)}{-\ln p(y, m)},$$

where $\texttt{NPMI}(\cdot, \cdot)$ is the point-wise mutual information computed as follows:

$$\texttt{PMI}(y, m) = \log \frac{p(y, m)}{p(y)p(m)},$$

where $p(y)$, $p(m)$ and $p(y, m)$ are the probability of entity mention $m$, label $y$ and their joint probability. For each label, NPMI is computed for all the entity mentions and only a list of top $k$ mentions are selected as prototypes. Note that NPMI is not applicable to unseen labels. In such case, it is necessary to combine manual selection and NPMI.

Word embeddings methods such as Skip-gram model (Mikolov et al., 2013) are shown capable of learning distributional semantics of words from unlabeled text corpora. To further avoid affected by label noises, we use pre-trained word embeddings as the source to compute prototype-driven label embeddings. For each label $y_i$, we compute its label embedding as the average of pre-trained word embeddings of the head words of prototypes, i.e.,

$$\tilde{B}_i^P = \frac{1}{k} \sum_{j=1}^{k} v_{m_{ik}},$$

where $v_{m_{ik}}$ denotes the word embedding of $k$th word in the prototype list of label $y_i$. In the case of using phrase embeddings, the full strings of multi-word prototypes could be used directly.

### 4.2 Hierarchical Label Embedding

Another side information that is available for generating label embeddings is the label hierarchy. We adapt the Hierarchical Label Embeddings (HLE) (Akata et al., 2013) to FNET task. Unlike (Akata et al., 2013), which uses the WordNet hierarchy, FNET systems typically have direct access to predefined tree hierarchy of type set. We denote the label embedding matrix resulting from label hierarchy as $\tilde{B}^H$. Each row in $\tilde{B}^H$ corresponds to a binary label embedding and has a dimension equal to the size of label set. For each label, the sets $\tilde{B}_{ij}^H$ to 1 when $y_j$ is the parent of $y_i$ or $i = j$, and 0 to the remainder,

$$\tilde{B}_{ij}^H = \begin{cases} 1 & \text{if } i = j \text{ or } y_j \in Parent(y_i) \\ 0 & \text{otherwise} \end{cases}.$$

HLE explicitly encodes the hierarchical dependency between labels by scoring a type $y_i$ given $m$ using not only $y_i$ but also its parent type $Parent(y_i)$. The underlying intuition is that recognition of a child type should be also based on the recognition of its parent.

### 4.3 Prototype-driven Hierarchical Label Embedding

One shortcoming of HLE is that it is too sparse. A natural solution is combining HLE with ProtoLE, which is denoted as Proto-HLE. Since $\tilde{B}^H \in \mathbb{R}^{N \times N}$ and $\tilde{B}^P \in \mathbb{R}^{D \times N}$, the combined embedding matrix $\tilde{B}^{HP}$ can be obtained by simply multiplying $\tilde{B}^H$ by $\tilde{B}^P$, i.e.,

$$\tilde{B}^{HP} = \tilde{B}^P \tilde{B}^{H\top}.$$

Note that $\tilde{B}^{HP}$ has the same shape as $\tilde{B}^P$, and it is actually representing the child label as a linear combination of the ProtoLE vectors of its parent and itself.

### 4.4 Type Inference

Having computed the scoring function for each label given a feature vector of the mention, we conduct type inference to refine the top $k$ type candidates. In the setting of few-shots FNET, $k$ is typically set to the maximum depth of type hierarchy, while different values for $k$ may be used for a better prediction of unseen labels in zero-shot typing. For top $k$ type candidates, we greedily remove the labels that conflict with others. However, unlike (Yogatama et al., 2015), we use a relative threshold $t$ to decide whether the selected type should remain in the final results, which is more consistent with the margin-infused objective function than a global threshold. Namely, a type candidate will be passed to type inference only if the difference of score from the 1-best is less than a threshold.

## 5 Experiments

### 5.1 Experiment Setup

Our method uses feature templates similar to what have been used by state-of-the-art FNET methods (Ling and Weld, 2012; Gillick et al., 2014; Yogatama et al., 2015; Xiang Ren, 2015). Table 1 illustrates the full set of feature templates used in this work. We evaluate the performance of our methods on three benchmark datasets that have been used for the FNET task: BBN dataset (Weischedel and Brunstein, 2005), OntoNotes dataset (Weischedel et al., 2011) and Wikipedia dataset (Ling and Weld, 2012). (Xiang Ren, 2015) has pre-processed the training sets of BBN and OntoNotes using DBpedia Spotlight[1]. Entity mentions in the training set are automatically linked to a named entity in Freebase and assigned with the Freebase types of induced named entity. As shown in Table 2, BBN dataset contains 2.3K news articles of Wall Street Journal, which includes 109K entity mentions belonging to 47 types. OntoNotes contains 13.1K news articles and 223.3K entity mentions belonging to 89 entity types. The size of Wikipedia dataset is much larger than the other two with 2.69M entity mentions of 113 types extracted from 780.5K Wikipedia articles. Each data set has a test set that is manually annotated for purpose of evaluation. To tune parameters such as the type inference threshold $t$ and trade-off parameter $\lambda$, we randomly sample 10% instances from each testing set as the development sets and use the rest as evaluation sets.

| Feature | Description | Example |
|---|---|---|
| Tokens | Unigram words in the mentions | "White", "House" |
| Head | Head word of the mention | "House" |
| Cluster | Brown Cluster IDs of the head word | "4_1111", .. ,"8_11111101" |
| POS Tag | POS tag of the mention | "NNP" |
| Character | Lower-cased character trigrams in the head word | "hou","ous","use" |
| Word Shape | The word shape of words in the mention | "Aa","Aa" |
| Context | Unigram/bigram words in context of the mention | "Bennett","the", "Bennett_the" |
| Dependency | Dependency relations involving the head word | "gov_nn_director" |

Table 1: Features extracted for context "*William Bennet, the [White House] drug-policy director....*"

| Dataset | | Types | Documents | Sentences | Mentions |
|---|---|---|---|---|---|
| BBN | train | 47 | 2.3K | 48.8K | 109K |
| | test | | 459 | 6.4K | 13.8K |
| OntoNotes | train | 89 | 13.1K | 147.7K | 223.3K |
| | test | | 76 | 1.3K | 9.6K |
| Wikipedia | train | 113 | 780.5K | 1.15M | 2.69M |
| | test | | - | 434 | 563 |

Table 2: Statistics of datasets

[1] http://github.com/dbpedia-spotlight/dbpedia-spotlight

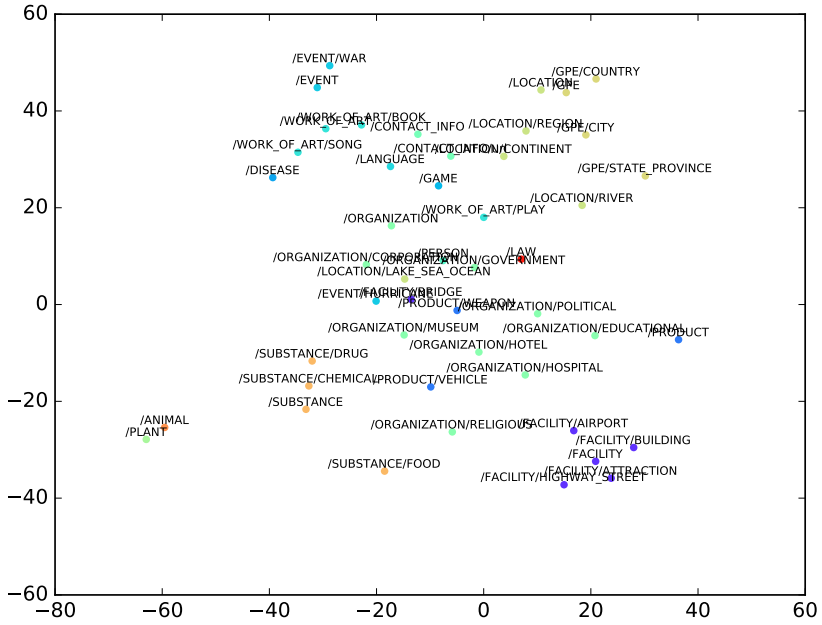



Figure 1: t-SNE visualization of the prototype-driven label embeddings for BBN dataset

Following prior works (Ling and Weld, 2012), we evaluate our methods and baseline systems using both loose and strict metrics, i.e., Macro-F1, Micro-F1, and strict Accuracy (Acc.). Given the evaluation set $D$, we denote $Y_m$ as the ground truth types for entity mention $m \in D$ and $\widehat{Y}_m$ as the predicted labels. Strict accuracy (Acc) can be computed as:Acc= $\frac{1}{D}\sum_{m\in D}\sigma(Y_m = \widehat{Y}_m)$, where $\sigma(\cdot)$ is an indicator function. Macro-F1 is based on Macro-Precision (Ma-P) and Micro-Recall (Ma-R), where Ma-P = $\frac{1}{|D|}\sum_{m\in D}\frac{|Y_m\cap\widehat{Y}_m|}{Y_m}$, and Ma-R= $\frac{1}{|D|}\sum_{m\in D}\frac{|Y_m\cap\widehat{Y}_m|}{Y_m}$. And Micro-F1 is based on Micro-Precision (Mi-P) and Micro-Recall (Mi-R), where Mi-P= $\frac{\sum_{m\in D}|Y_m\cap\widehat{Y}_m|}{\sum_{m\in D}\widehat{Y}_m}$, and Mi-R= $\frac{\sum_{m\in D}|Y_m\cap\widehat{Y}_m|}{\sum_{m\in D}Y_m}$.

### 5.2 Generating ProtoLE

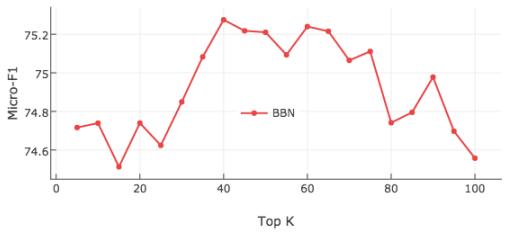
Our ProtoLE embeddings use Continuous-Bag-of-Words (CBOW) word embedding model (Mikolov et al., 2013) trained on Wikipedia dump using a window of 2 words to both directions. We use 300 dimensions for all embedding methods except HLE. Table 3 illustrates examples of prototypes learned for types in BBN dataset. It can be observed that most of the top ranked mentions are correctly linked to types, even though there are still some noises, e.g., north_american for `/LOCATION/CONTINENT`. It also shows that prototypes of related types such as `/LOCATION` and `/GPE` are also semantically related. Figure1 visualizes the prototype-driven label embeddings for BBN dataset using -Distributed Stochastic Neighbor Embedding (t-SNE)(Maaten and Hinton, 2008). It can be easily observed that semantic related types are close to each other in the new space, which proves that prototype-driven label embeddings can capture the semantic correlation between labels.

Figure 2 shows the Micro-F1 score of FNET with regard to the number of PMI prototypes used by ProtoLE. It shows that the Micro-F1 score does not change significantly on BBN and Wikipedia dataset, whereas using fewer prototypes per type ($\leq 40$) results in a drop of Micro-F1. Since the definitions of several types, especially the coarse-grained types, are actually very general, it may introduce bias into the label embeddings if using too few prototypes. We use $K = 60$ for all our experiments for that it achieves decent performance on all three datasets. Our pre-trained label embeddings and manually-selected prototypes (zero-shot typing) are available for download[2].

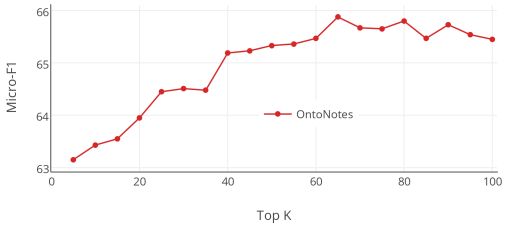
[2] `http://github.com/fnet-coling/ner-zero/tree/master/label_embedding`

| Type | Prototypes |
|---|---|
| /LOCATION | areas connaught earth lane brooklyn |
| /LOCATION/CONTINENT | north_america europe africa north_american asia |
| /LOCATION/LAKE_SEA_OCEAN | big_bear lake_erie champ lake_geneva fujisawa |
| /LOCATION/RIVER | hudson thompson mississippi_river james_river tana |
| /GPE | soviet edisto canada china france |
| /GPE/STATE_PROVINCE | california texas ohio arizona jersey |

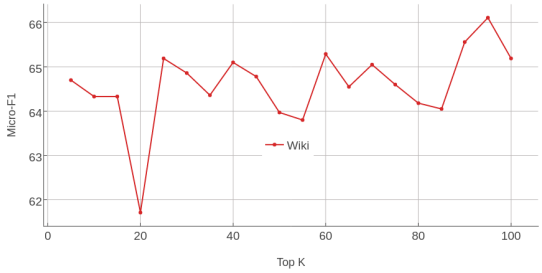
Table 3: Example prototypes learned by PMI for types in BBN dataset




(a) BNN Dataset




(b) OntoNotes Dataset




(c) OntoNotes Dataset

Figure 2: Performance changes on the development set with regard to the sizes of prototype list

### 5.3 Few-shots Fine-grained Entity Typing

In this section, we compare performances of FNET methods in the setting of few-shots FNET where the training set covers all types. Methods compared in this section are trained using the entire type set. We use evaluation metrics for our experiments: macro-F1, micro-F1 and accuracy. As in section 3.2, we train our label embeddings in two different ways: 1) non-adaptive training where label embeddings are fixed during training; and 2) adaptive training where label embeddings are also updated. Table 4 shows the comparison with state-of-the-art FNET methods: FIGER(Ling and Weld, 2012), HYENA(Yosef et al., 2012) and WSABIE (Yogatama et al., 2015). We make several findings from the results.

Firstly, embedding methods with WARP loss function consistently outperform non-embedding methods (i.e., FIGER and HYENA) on all three datasets. The performance gaps are huge for BBN and OntoNotes, where the best embedding method achieves 10%-20% absolute improvement over the best non-embedding method (FIGER). However, the gap is much smaller on Wikipedia dataset whose size is significantly larger than the other two.

Secondly, non-adaptive embedding methods always outperform their adaptive versions except HLE on Wikipedia dataset. Performance of adaptive label embeddings are all close to WSABIE, which suggests that adaptive label embeddings might suffer from same label noise problem as WSABIE does.

Thirdly, our ProtoLE and its combination with HLE consistently outperform both non-embedding and embedding baselines. Using the prototype information and non-adaptive framework results in absolute 3%-5% improvement with both loose and strict evaluation metrics. Non-adaptive HLE performs poorer than other embedding methods, which is most likely due to its sparsity in representing labels. However, Proto-HLE performs very close to ProtoLE on BBN and Wiki, while it improves all three measures by another absolute ≈2.5% on OntoNotes .

| Method | Adapt | BBN | | | OntoNotes | | | Wiki | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Ma-F1 | Mi-F1 | Acc. | Ma-F1 | Mi-F1 | Acc. | Ma-F1 | Mi-F1 | Acc. |
| FIGER | NA | 67.28 | 60.70 | 46.92 | 58.77 | 52.37 | 38.01 | **68.28** | 64.71 | 47.37 |
| HYENA | NA | 51.38 | 52.85 | 45.01 | 47.65 | 43.97 | 26.56 | 45.51 | 43.80 | 30.67 |
| WSABIE | NA | 71.28 | 72.08 | 66.22 | 62.03 | 55.83 | 43.61 | 67.97 | 64.49 | 48.28 |
| HLE | Y | 70.84 | 71.61 | 65.74 | 61.54 | 49.16 | 43.25 | 67.09 | 65.65 | 47.01 |
| | N | 68.86 | 70.00 | 63.32 | 59.52 | 54.01 | 41.60 | 65.29 | 62.53 | 45.19 |
| ProtoLE | Y | 72.67 | 73.54 | 67.58 | 60.90 | 54.68 | 42.82 | 66.96 | 65.78 | 49.18 |
| | N | **75.78** | **76.50** | **70.43** | 65.91 | 59.08 | 46.94 | 68.06 | **66.53** | **53.54** |
| Proto-HLE | Y | 71.97 | 72.89 | 67.05 | 62.71 | 56.64 | 44.81 | 67.85 | 65.74 | 50.27 |
| | N | 74.54 | 74.38 | 69.46 | **68.23** | **61.27** | **49.30** | 66.61 | 65.29 | 50.45 |

Table 4: Performance of FNET in a few-shots learning on 3 benchmark datasets

### 5.4 Zero-shot Fine-grained Entity Typing

In this section, we evaluate our method's capability recognizing mentions of unseen fine-grained types. We assume that the training set contains only coarse-grained types (i.e., Level-1), and Level-2 types are unseen types to be removed from the training set. Table 5 shows the Micro-Precision for Level-1 and Level-2 types using top $k$ type candidates for type inference. NPMI is computed for Level-1 types. We manually build prototype lists for unseen types by choosing from a randomly sampled list of entity mentions. Level-3 types are ignored for OntoNotes as Level-3 types never show in top-10 list produced by all methods. As the prediction for coarse-grained types are the same with regard to $k$, we only list the results using $k = 3$.

One interesting finding on all three datasets is that combining hierarchical and prototypical information results in better classification of coarse-grained types. It suggests that embeddings of unseen fine-grained types contains information complementary to the embeddings of coarse-grained types. Since HLE actually produces random prediction on Level-2 types due to its sparse representation, HLE perform poorly on Level-2 types.

| Data Set | Method | Micro-Precision @k Level 1 | Micro-Precision @k Level 2 | | |
|---|---|---|---|---|---|
| | | 3 | 3 | 5 | 10 |
| BBN | ProtoLE | 76.71 | **42.95** | **36.61** | **42.34** |
| | HLE | 70.44 | 13.08 | 13.16 | 12.82 |
| | Proto-HLE | **76.89** | 42.35 | 35.18 | 30.16 |
| OntoNotes | ProtoLE | 73.26 | **21.01** | **13.72** | **12.22** |
| | HLE | 66.96 | 7.13 | 6.14 | 6.23 |
| | Proto-HLE | **76.33** | 7.09 | 11.43 | 9.91 |
| Wiki | ProtoLE | 65.52 | 12.50 | 21.28 | 17.91 |
| | HLE | 65.13 | 0.00 | 8.82 | 8.99 |
| | Proto-HLE | **67.41** | **20.01** | **31.25** | **24.24** |

Table 5: Performance of zero-shot entity typing

ProtoLE outperforms HLE by 100%-300% in terms of Micro-Precision. However, again the combination of prototypes and hierarchy achieves similar or better results than ProtoLE on BBN and Wikipedia dataset. The drop of precision of Proto-HLE on OntoNotes is likely due to a different nature of annotation. It is more prevalent in test set of OntoNotes that one entity mention is annotated with multiple Level-1 types, and the presence of fine-grained types are less constrained by the label hierarchy. In such case, hierarchical constrains enforced by Proto-HLE might have negative impacts on type inference.

## 6 Conclusion

In this paper, we presented a prototype-driven label embedding method for fine-grained named entity typing (FNET). It shows that our method outperforms state-of-the-art embedding-based FNET methods in both few-shots and zero-shots settings. It also shows that combining prototype-driven label embeddings and type hierarchy can improve the prediction on coarse-grained types. In the near future, we plan to integrate our method with other types of side information such as definition sentences as well as label noise reduction framework (Ren et al., 2016) to further boost the robustness of FNET.

## Acknowledgements

This work was conducted within the Rolls-Royce@NTU Corp Lab with support from the National Research Foundation Singapore under the Corp Lab@University Scheme.

## References

Zeynep Akata, Florent Perronnin, Zad Harchaoui, and Cordelia Schmid. 2013. Label-embedding for attribute-based classification. In *Proceedings of CVPR*, pages 819–826.

Zeynep Akata, Scott Reed, Daniel Walter, Honglak Lee, and Bernt Schiele. 2015. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of CVPR*, pages 2927–2936.

Nancy Chinchor and Patricia Robinson. 1997. Muc-7 named entity task definition. In *Proceedings of the 7th MUC*, page 29.

George R Doddington, Alexis Mitchell, Mark A Przybocki, Lance A Ramshaw, Stephanie Strassel, and Ralph M Weischedel. 2004. The automatic content extraction (ace) program-tasks, data, and evaluation. In *LREC*, page 1.

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 SIGKDD*, pages 601–610.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open question answering over curated and extracted knowledge bases. In *Proceedings of SIGKDD*, pages 1156–1165.

Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Tomas Mikolov, et al. 2013. Devise: A deep visual-semantic embedding model. In *NIPS*, pages 2121–2129.

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

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Revisiting embedding features for simple semi-supervised learning. In *Proc. of EMNLP*, pages 110–120.

Aria Haghighi and Dan Klein. 2006. Prototype-driven learning for sequence models. In *Proceedings of HLT-NAACL*, pages 320–327. Association for Computational Linguistics.

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In *AAAI*, page 3.

Thomas Lin, Oren Etzioni, et al. 2012. No noun phrase left behind: detecting and typing unlinkable entities. In *Proceedings of EMNLP*, pages 893–903.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *In Proc. of the 26th AAAI Conference on Artificial Intelligence*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS'13*, pages 3111–3119.

Xiang Ren, Wenqi He, Meng Qu, Clare R Voss, Heng Ji, and Jiawei Han. 2016. Label noise reduction in entity typing by heterogeneous partial-label embedding. *arXiv preprint arXiv:1602.05307*.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*, pages 142–147.

Kilian Q Weinberger and Olivier Chapelle. 2009. Large margin taxonomy embedding for document categorization. In *Advances in Neural Information Processing Systems*, pages 1737–1744.

Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. Technical report, Linguistic Data Consortium, Philadelphia.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium*.

Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. Wsabie: Scaling up to large vocabulary image annotation. In *Proceedings of IJCAI'11*.

Chi Wang Xiang Ren, Ahmed El-Kishky. 2015. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *Proceedings of SIGKDD'15*.

Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of ACL'15*, pages 291–296.

Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. 2012. HYENA: Hierarchical type classification for entity names. In *Proceedings of COLING 2012: Posters*, pages 1361–1370.