Description-Based Zero-shot Fine-Grained Entity Typing

Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi and Prasad Tadepalli
School of Electrical Engineering and Computer Science, Oregon State University
{obeidatr, Xfern, shahbazh, tadepall}@eecs.oregonstate.edu

Abstract

Fine-grained Entity typing (FGET) is the task of assigning a fine-grained type from a hierarchy to entity mentions in the text. As the taxonomy of types evolves continuously, it is desirable for an entity typing system to be able to recognize novel types without additional training. This work proposes a *zero-shot* entity typing approach that utilizes the type description available from Wikipedia to build a distributed semantic representation of the types. During training, our system learns to align the entity mentions and their corresponding type representations on the known types. At test time, any new type can be incorporated into the system given its Wikipedia descriptions. We evaluate our approach on FIGER, a public benchmark entity tying dataset. Because the existing test set of FIGER covers only a small portion of the fine-grained types, we create a new test set by manually annotating a portion of the noisy training data. Our experiments demonstrate the effectiveness of the proposed method in recognizing novel types that are not present in the training data.

1 Introduction

Entity Typing assigns a semantic type (e.g., *person*, *location*, *organization*) to an entity mention in text based on the local context. It is useful for enhancing a variety of Natural Language Processing(NLP) tasks such as question answering (Han et al., 2017; Das et al., 2017), relation extraction (Liu et al., 2014; Yaghoobzadeh et al., 2016), and entity linking (Stern et al., 2012). Traditional Named Entity Typing systems consider a small set of coarse types (e.g., *person*, *location*, *organization*) (Tjong Kim Sang and De Meulder, 2003; Krishnan and Manning, 2006; Chieu and Ng, 2002). Recent studies address larger sets of fine-grained types organized in type hierarchies (e.g., *person/artist*, *person/author*) (Ling and Weld, 2012; Corro et al., 2015; Xu and Barbosa, 2018; Murty et al., 2018). Fine-Grained Entity Typing (FGET) is usually approached as a multi-label classification task where an entity mention can be assigned multiple types that usually constitute a path in the hierarchy (Ren et al., 2016).

In real-world scenarios, there is a need to deal with ever-growing type taxonomies. New types emerge, and existing types are refined into finer sub-categories. Traditional methods for entity typing assume that the training data contains all possible types, thus require new annotation effort for each new type that emerges. *Zero-shot learning (ZSL)*, a special kind of transfer learning, allows for new types to be incorporated at the prediction stage without the need for additional annotation and retraining. The main idea behind ZSL is to learn a shared semantic space for representing both the seen and unseen types, which allows the knowledge about how examples link to the seen types to be transferred to unseen types.

For fine-grained entity types, we observe that their associated Wikipedia pages often provide a rich description of the types. To capture this, we propose a *Description-based* Zero-shot Entity Typing (DZET) approach that utilizes the Wikipedia description of each type (e.g., see `https://en.wikipedia.org/wiki/Artist` for description of the type *person/artist*) to generate a representation of that type. We learn to project the entity-mention representations and the type representations into a shared semantic space, such that the mention is closer to the correct type(s) than the incorrect types. The mid-level type representation derived from the Wikipedia page along with the learned projection function allows the system to recognize new types requiring zero training examples.

We investigate different approaches for constructing the type representation based on

Proceedings of NAACL-HLT 2019, pages 807–814
Minneapolis, Minnesota, June 2 - June 7, 2019.

Wikipedia descriptions. Note that the descriptions can be quite long, often containing many different parts that are useful for recognizing different entity mentions. This motivates us to generate a bag of representations for each type and apply average pooling to aggregate the results.

We evaluate the performance of our methods on FIGER, a benchmark dataset for the FNET task, in which types are organized in 2-levels hierarchy. In this work, We focus on testing our method's capability in recognizing unseen fine-grained types (Level-2 types in this dataset). As the current test set of FIGER contains examples from only a few level-2 types, we created a new test data that covers most of the level-2 types by manually annotating a portion of the noisy training data. Below we summarize our main contributions.

- We proposed a description-based zero-shot fine-grained entity typing framework that uses Wikipedia descriptions to represent and detect novel types unseen in training.
- We created a new test set for fine-grained entity typing that provides much better coverage of the level-2 (fine-grained) types compared to the original FIGER test data.
- We provided experimental evidence of the effectiveness of our approach in comparison with established baselines.

2 Related Work

Existing work on FGET focuses on performing context-sensitive typing (Gillick et al., 2014; Corro et al., 2015), learning from noisy training data (Abhishek et al., 2017; Ren et al., 2016; Xu and Barbosa, 2018), and exploiting the type hierarchies to improve the learning and inference (Yogatama et al., 2015; Murty et al., 2018). More recent studies support even finer granularity (Choi et al., 2018; Murty et al., 2018). However, all the methods above have the limitation that they assume all types are present during training.

Zero-Shot Learning has been extensively studied in Computer Vision (CV) (Wang et al., 2019) for tasks such as image classification (Lampert et al., 2014; Zhang and Saligrama, 2015; Socher et al., 2013), object localization (Li et al., 2014, 2017) and image retrieval (Xu et al., 2017; Zhang et al., 2018). A common approach for zero-shot learning in CV is to represent each class (e.g., Zebra) by a set of semantic attributes such as its shape and color. The semantic attributes serve as the intermediate level that connects the visual features with the classes. The model is trained to learn an alignment between the semantic attributes and the visual features where a new class can be recognized using its semantic attributes without the need for any training examples. In contrast, this type of approach tends not to work well for NLP applications as the semantic concepts/classes in NLP are often more complex and cannot be easily described by a set of pre-defined attributes. This explains why the few studies of ZSL for NLP use very different methods to create the transferable intermediate representations.

Zero-Shot Learning has been studied for a number of NLP tasks including event extraction (Huang et al., 2018; Lee and Jha, 2018; Srivastava et al., 2018), relation extraction(Liu et al., 2014), Conversational Language Understanding (Lee and Jha, 2018). Specifically, Zero shot entity typing has also been explored, where most of the prior methods adopt the idea of learning a shared semantic space for representing the entities as well as the types, but differ in how they construct the type embeddings. In OTyper (Yuan and Downey, 2018), each type is represented by averaging the embedding of the words constitutes the type label. On the other hand, ProtoLE (Ma et al., 2016) represents each type by a prototype that consists of manually selected entity mentions, where the type embedding is obtained by averaging the prototype mentions' word embeddings.In contrast, our work differs from OTyper and ProtoLE by constructing the type representations based on the Wikipedia descriptions of the types, which not only carry more information about the type but also can be easily adapted to other tasks such as event typing and text classification.

3 Proposed Approaches

Following prior work on fine-grained entity typing, we formulate it as a multi-class multi-label classification problem. Given an entity mention m along with its left textual context c_l and right context c_r, We learn a classifier that predicts a binary label vector $y \in \{0,1\}^{|L|}$, where L denotes the set of all types, which forms a hierarchy Ψ. Here $y^{(t)} = 1$ if the mention m is of type t, and 0 otherwise. In the case of zero-shot FGET, new types can be introduced and added to L during testing.

3.1 The Typing Function

We will begin by introducing our typing function that is used to compute a score between a given mention and type pair, given their corresponding vector representations. We will discuss how to construct the representations in later sections.

Formally, the input to this typing function consists of the representation of the mention, denoted by $x \in \mathbf{R}^d$; and the representation of a candidate type t, denoted by $y_t \in \mathbf{R}^{\acute{d}}$. It computes a bi-linear score for the (x, y_t) pair as follows:

$$f(x, y_t, W) = x^T W y_t$$

where $W \in \mathbf{R}^{d \times \acute{d}}$ is a compatibility matrix. Following (Yogatama et al., 2015; Ma et al., 2016), we factorize W as a product of two low-rank matrices to reduce the number of parameters. That is $W = A^T B$, where $A \in \mathbf{R}^{h \times d}$ and $B \in \mathbf{R}^{h \times \acute{d}}$ (We use $h = 20$). The scoring function f can be rewritten as:

$$f(x, y_t, A, B) = \theta(x, A) \cdot \phi(y_t, B) = (Ax)^T B y_t$$

where $\theta(x, A) : x \rightarrow Ax$ and $\phi(y_t, B) : y_t \rightarrow By_t$ serve as the projection functions that map x and y_t into a shared semantic space.

3.2 Entity Mention Representation

To obtain the representation for entity mentions, we adopt the same neural approach proposed by Shimaoka et al. (2017). Given an entity mention with its context, we compute a vector v_m to present the mention m itself, and another vector v_c to represent its left and right contexts c^l and c^r. v_m is computed by simply averaging the embedding of the individual words in m.

To compute the context embedding v_c, we first encode c^l and c^r using a bidirectional-LSTM. Let $c^l_1, ..., c^l_s$ and $c^r_1, ..., c^r_s$ be the word embedding of the left and the right context respectively, where s is the window size (we use $s = 10$), the output layer of the bi-LSTM is denoted as: $\overrightarrow{h^l_1}, \overleftarrow{h^l_1} ..., \overrightarrow{h^l_s}, \overleftarrow{h^l_s}$ and $\overrightarrow{h^r_1}, \overleftarrow{h^r_1} ..., \overrightarrow{h^r_s}, \overleftarrow{h^r_s}$. We then compute a scalar attention for each context word using a 2-level feedforward neural network:

$$e^j_i = tanh(W_e \begin{bmatrix} \overrightarrow{h^j_i} \\ \overleftarrow{h^j_i} \end{bmatrix}); \tilde{a}^j_i = exp(W_a e^j_i)$$

Where $W_e \in \mathbf{R}^{d_h \times 2 \times d_a}$, $W_a \in \mathbf{R}^{1 \times d_a}$, d_h is the dimension of LSTM, d_a is the attention dimension, $j \in \{l, r\}$. Next, we normalize a^j_is such that they sum up to 1. i.e., $a^j_i = \frac{\tilde{a}^j_i}{\sum_{i=1}^{s}(\tilde{a}^l_i + \tilde{a}^r_i)}$. Finally the context representation is computed as $v_c = \sum_{i=1}^{s}(a^l_i \begin{bmatrix} \overrightarrow{h^l_i} \\ \overleftarrow{h^l_i} \end{bmatrix} + a^r_i \begin{bmatrix} \overrightarrow{h^r_i} \\ \overleftarrow{h^r_i} \end{bmatrix})$. The final representation of the entity mention $x \in \mathbf{R}^d$ is a concatenation of v_m and v_c.

3.3 Type Representation

Let P_t be the Wikipedia page that is used to build a representation for type t. Some types do not have a Wikipedia page with a title the same as the type label. In such cases, we manually look for a Wikipedia page of a similar concept. For example, we represent the type *living-thing* by the Wikipedia page *organism*.

To get a type representation, We started by the simplest possible method which is averaging the embedding of words in the Wikipedia page (we call this **Avg encoder**). Since some words in the Wikipedia page carry more of the type semantic than the other words we also consider a (tf-idf)-weighted version of the Avg encoder.

Learning multiple representations. Wikipedia descriptions are often long and contain multiple parts, where different parts may capture different aspects of the type and relate to different mentions. Moreover, sequence models such as LSTM cannot be applied to such long sequences. This motivates us to consider the approach of constructing a bag of multiple representations for each type based on its Wikipedia description. To obtain a bag of representations for type t, we first use a fixed-length window to incrementally break P_t into multiple parts, one paragraph at a time. If a paragraph fits in the current Window, it is added. Otherwise, a new window is initiated. Each window of text r_{ti} is then used to generate one representation. To construct an embedding for r_{ti}, we adopt the same Bi-directional LSTM and attention mechanism that are used to embed the mention context.

To compute the score for type t given its multiple representations, we compute the score with each individual representation and average them to produce the final score. This is equivalent to applying average pooling to the multiple representations to obtain a single representation due to the bi-linear typing function.

3.4 Training and inference

Given the training data, we jointly train the representation and the scoring function by minimizing a ranking score. Let $\mathcal{Y}^{(i)}$ and $\overline{\mathcal{Y}}^{(i)}$ denote the set of correct and incorrect types assigned to the example $x^{(i)}$ respectively, we learn to score types in $\mathcal{Y}^{(i)}$ higher than types in $\overline{\mathcal{Y}}^{(i)}$ with a multi-label max-margin ranking objective as follows:

$$\sum_{y \in \mathcal{Y}} \sum_{\acute{y} \in \overline{\mathcal{Y}}} max(0, 1 - f(x, y, A, B) + f(x, \acute{y}, A, B))$$

At testing, both seen and unseen types are mapped to their learned representations, which are then scored for a given input. Given the scores, we conduct a top-down search following the type hierarchy Ψ. Starting from the root we recursively find the type with the highest score among the children. Since we focus on the fine-grained types, we stop the search when a leaf type is reached and predict that the mention is positive for all types along the path leading the to leaf type.

4 Experiments

Datasets. Our experiments use FIGER, a publicly available fine-grained entity typing benchmark dataset in which types are organized into a 2-level hierarchy. The training data consists of sentences sampled from Wikipedia articles and automatically annotated via distant supervision (Ling and Weld, 2012). The test data consisting of manually annotated sentences sampled from news reports.

Setting. To evaluate our capability to recognize fine-grained types in zero-shot setting, we assume all second-level types are unseen during training, i.e., we remove all level-2 types from the train and dev data but keep them in the test data. We observe that the FIGER test set covers only a small number of second-level types. This renders it insufficient for testing under the evaluation setting we adopt. Moreover, the training data is noisy since it is automatically annotated by distant supervision. As a result, we cannot just use part of it for testing.

	Original dataset			New dataset		
	train	dev	test	train	dev	test
# of mentions	2000k	10k	563	1999k	10k	917
# of types	111	111	47	46	46	66
# of level-2 types	65	65	26	0	0	40

Table 1: Statistics of FIGER dataset.

To overcome this limitation, We manually annotated a new test set from the noisy training data. We first divide the train set into clean and noisy as suggested in (Ren et al., 2016). Clean examples are those whose types fall on a single path (not necessarily ending with a leaf) in Ψ. For instance, the mention with labels *person*, *person/author*, and *person/doctor* is considered as noisy example because the labels form two paths. We then manually verify the correctness of up to 20 examples from the clean training data for every level-2 type. These examples are removed from training and added to the test set. We ignore the types with no clean examples. The statistics of the new and original datasets are reported in Table 1.

Baselines. We consider two baselines that employ the same neural architecture but use different type representations. The **Label embd** baseline use the average of the embedding of the words in the type label as the type representation. **ProtoLE** baseline uses the prototypes-based label embedding learned by Ma et al. (2016), where each type is represented by the set of the most representative entity mentions. The type embedding is the average of all mentions in the corresponding prototype.

Evaluation metrics. Following prior works in FGET, we report *Accuracy (Strict-*$F1$*)*, *loose Macro-averaged* $F1$ ($F1_{ma}$) and *loose Micro-averaged* $F1$ ($F1_{mi}$) (Ling and Weld, 2012). The training and hyperparameter tuning details are described in the Appendices.

Results and discussions. Table 2 presents the results on FIGER, evaluated on all types (Overall), the seen types (Level-1) and the unseen types (Level-2) respectively. From the results, we can see that our description based methods have a particularly strong advantage over baselines on level-2 types. This is consistent with our expectation because Wikipedia descriptions tend to be highly informative for fine-grained types, but less so for coarser types.

Among the average encoders, we found that weighting the word embedding by the word tf-idf produces better results than treating the words equivalently. As expected, using LSTM based multi-representation adds a noticeable benefit to our system as it produces the best performance among all tested methods, achieving the best performance for level-2 types and outperforming oth-

Approach	Overall			Level-1		Level-2	
	Acc	$F1_{ma}$	$F1_{mi}$	$F1_{ma}$	$F1_{mir}$	$F1_{ma}$	$F1_{mir}$
Label embd	0.2846	0.5510	0.5603	**0.8165**	**0.8163**	0.2854	0.2954
ProtoLE	0.2541	0.4982	0.5093	0.7424	0.7422	0.2541	0.2657
DZET + Avg encoder	0.3141	0.5522	0.5614	0.7903	0.7902	0.3141	0.3247
DZET + Weighted Avg encoder	0.3261	0.5500	0.5607	0.7740	0.7738	0.3261	0.3390
DZET + Multi-rep	**0.3806**	**0.5953**	**0.6045**	0.8100	0.8098	**0.3806**	**0.3926**

Table 2: Level-1 , Level-2 and overall performance of the models on FIGER dataset.

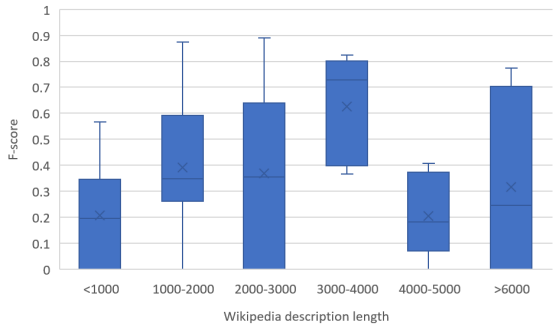

Figure 1: The relationship between the length of the Wikipedia description (word count) of level-2 types and the F-score obtained by *DZET+Multi-rep* method.

ers by a large margin while maintaining a highly competitive performance for level-1 types.

The effect of description quality. Figure 1 analyzes the relationship between the length of the Wikipedia description as one criterion of the description quality and the performance of *Multi-rep* method. In particular, we group the types based on the length of their Wikipedia descriptions and provide the five-number summary box plot of the F-scores for each group. It can be readily observed that the performance is low when the description of the type's Wikipedia page is too short ($<$ 1000 words) or too long ($>$ 4000 words). Short descriptions are less informative and carry less shared semantics with the type's mentions. On the other hand, overly long descriptions could also be confusing as it might share a significant number of common words with the descriptions of other types. A closer look into the results unveils some exceptions. For example, the F-score on the type '/education/educational-degree' is 0.7742 even it has a long description (6845 words). The description of this type is indeed very informative and includes a comprehensive list of the educational degrees awarded all around the world.

The length of the description is not the only factor that affects the performance of DZET methods. One factor is the performance on the Level-1 types. Since the inference is performed by following the type hierarchy, if an incorrect type is inferred at level-1, there is no hope to get the correct level-2 type. Another factor is the amount of overlapping between the descriptions of the related types. For instance, *Multi-rep* produces zero F-score on the types'/event/protest' and '/location/province' because they share a lot of common words with the types '/event/attack' and '/location/county' respectively, which negatively affects the ability of *Multi-rep* to distinguish between the related types. Both '/event/protest' and '/location/province' have a description length between 2000 and 3000 words.

To mitigate the effect of the contents overlapping between the highly related type, We plan to apply mention-sensitive attention mechanisms for future work to aggregate the scores in Multi-rep instead of max-pooling.

5 Conclusions

In this paper, we propose a novel zero-shot entity typing approach that uses Wikipedia descriptions to construct type embeddings. Our architecture relies on the type embeddings to make predictions for unseen types. Experimental results demonstrate the effectiveness of the proposed methods.

Acknowledgments

We thank Jordan University of Science and Technology for Ph.D. fellowship (to R. O.).

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.

Hai Leong Chieu and Hwee Tou Ng. 2002. Named entity recognition: a maximum entropy approach using global information. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *In Annual Meeting of the Association for Computational Linguistics*.

Luciano del Corro, Abdalghani Abujabal, Rainer Gemulla, and Gerhard Weikum. 2015. Finet: Context-aware fine-grained named entity typing. Assoc. for Computational Linguistics.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017. Question answering on knowledge bases and text using universal schema and memory networks. *arXiv preprint arXiv:1704.08384*.

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

Sangdo Han, Soonchoul Kwon, Hwanjo Yu, and Gary Geunbae Lee. 2017. Answer ranking based on named entity types for question answering. In *Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication*, page 71. ACM.

Lifu Huang, Heng Ji, Kyunghyun Cho, and Clare R Voss. 2018. Zero-shot transfer learning for event extractions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1.

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

Vijay Krishnan and Christopher D Manning. 2006. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1121–1128. Association for Computational Linguistics.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2014. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465.

Sungjin Lee and Rahul Jha. 2018. Zero-shot adaptive transfer for conversational language understanding. *arXiv preprint arXiv:1808.10059*.

Zhenyang Li, Efstratios Gavves, Thomas Mensink, and Cees GM Snoek. 2014. Attributes make sense on segmented objects. In *European Conference on Computer Vision*, pages 350–365. Springer.

Zhenyang Li, Ran Tao, Efstratios Gavves, Cees GM Snoek, and Arnold WM Smeulders. 2017. Tracking by natural language specification. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7350–7358. IEEE.

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

Yang Liu, Kang Liu, Liheng Xu, and Jun Zhao. 2014. Exploring fine-grained entity type constraints for distantly supervised relation extraction. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2107–2116.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 97–109.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016. 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.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pages 935–943.

Shashank Srivastava, Igor Labutov, and Tom Mitchell. 2018. Zero-shot learning of classifiers from natural language quantification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 306–316.

Rosa Stern, Benoît Sagot, and Frédéric Béchet. 2012. A joint named entity recognition and entity linking system. In *Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data*, pages 52–60. Association for Computational Linguistics.

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-Volume 4*, pages 142–147. Association for Computational Linguistics.

Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):13.

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.

Yahui Xu, Yang Yang, Fumin Shen, Xing Xu, Yuxuan Zhou, and Heng Tao Shen. 2017. Attribute hashing for zero-shot image retrieval. In *Multimedia and Expo (ICME), 2017 IEEE International Conference on*, pages 133–138. IEEE.

Yadollah Yaghoobzadeh, Heike Adel, and Hinrich Schütze. 2016. Noise mitigation for neural entity typing and relation extraction. *arXiv preprint arXiv:1612.07495*.

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.

Zheng Yuan and Doug Downey. 2018. Otyper: A neural architecture for open named entity typing. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Haofeng Zhang, Yang Long, and Ling Shao. 2018. Zero-shot hashing with orthogonal projection for image retrieval. *Pattern Recognition Letters*.

Ziming Zhang and Venkatesh Saligrama. 2015. Zero-shot learning via semantic similarity embedding. In *Proceedings of the IEEE international conference on computer vision*, pages 4166–4174.

A Appendices

A.1 Training and Hyperparamters

For every model we trained, we tune all of the hyper-parameters using a dev set. We use the version of FIGER provided by (Shimaoka et al., 2017) which already withhold a portion of the train set as a dev set. For each experiment, we report that testing results of the model that has the best *accuracy* on the dev set. We adopt *glove* 300-dimensional word embedding (Pennington et al., 2014) throughout this work except for *prototype* baselines; we use word2vec (Mikolov et al., 2013) as it is used to compute the prototypes embedding in the original works (Ma et al., 2016). The hyper-parameters used in the feature representation component are the same as in (Shimaoka et al., 2017). we set both of the hidden-size of the LSTM was set and the hidden-layer size of the attention module to 100. We use Adam optimizer (Kingma and Ba, 2014) with the learning rate .001. The model is trained for five epochs. We use Window of size 200 to build a bag of representations for each type from its Wikipedia description.