ASSET: A Semi-supervised Approach for Entity Typing in Knowledge Graphs

Hamada M. Zahera
DICE Group
Department of Computer Science
Paderborn University
Paderborn, Germany
hamada.zahera@uni-paderborn.de

Stefan Heindorf
DICE Group
Department of Computer Science
Paderborn University
Paderborn, Germany
heindorf@uni-paderborn.de

Axel-Cyrille Ngonga Ngomo
DICE Group
Department of Computer Science
Paderborn University
Paderborn, Germany
axel.ngonga@uni-paderborn.de

ABSTRACT

Entity typing in knowledge graphs (KGs) aims to infer missing types of entities and might be considered one of the most significant tasks of knowledge graph construction since type information is highly relevant for querying, quality assurance, and KG applications. While supervised learning approaches for entity typing have been proposed, they require large amounts of (manually) labeled data, which can be expensive to obtain. In this paper, we propose a novel approach for KG entity typing that leverages semi-supervised learning from massive unlabeled data. Our approach follows a teacher-student paradigm that allows combining a small amount of labeled data with a large amount of unlabeled data to boost performance. We conduct several experiments on two benchmarking datasets (FB15k-ET and YAGO43k-ET). Our results demonstrate the effectiveness of our approach in improving entity typing in KGs. Given type information for only 1% of entities, our approach ASSET predicts missing types with a F_1-score of 0.47 and 0.64 on the datasets FB15k-ET and YAGO43k-ET, respectively, outperforming supervised baselines.

CCS CONCEPTS

• **Computing methodologies → Knowledge representation and reasoning**; **Semi-supervised learning settings**.

KEYWORDS

Entity Typing; Semi-supervised Learning; Knowledge Graph Completion; Knowledge Distillation

ACM Reference Format:
Hamada M. Zahera, Stefan Heindorf, and Axel-Cyrille Ngonga Ngomo. 2021. ASSET: A Semi-supervised Approach for Entity Typing in Knowledge Graphs. In *Proceedings of the 11th Knowledge Capture Conference (K-CAP '21), December 2–3, 2021, Virtual Event, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3460210.3493563

1 INTRODUCTION

In recent years, knowledge graphs (KGs) have had a major impact on many applications, such as web search, natural language processing, and biomedicine [6]. One of the most important features of KGs are the types of their entities such as *person*, *location*, or *organization*. This information is often required for querying, e.g., by means of SPARQL, for quality assurance, e.g., by means of SHACL, as well as for question answering [19]. Unfortunately, type information in existing knowledge graphs is often incomplete, noisy, and coarse-grained [5]. This pertains both to the construction of new knowledge graphs as well as to the maintenance of existing knowledge graphs, e.g. by introducing new types. For example, 36% of entities in DBpedia do not have types [8] and 10% of entities in Freebase labeled with *artist/music* are missing the type *people/person* [11].

While *supervised* approaches for entity type prediction have been proposed, e.g., based on hand-crafted features [10], knowledge graph embeddings [21], and language models [1], they all require a substantial amount of training data which is often not available. In contrast, *unsupervised* approaches based on clustering [3] do not require a priori labeled training data, but require labeling of clusters and do not reach the same predictive performance as supervised approaches. To overcome this challenge and to close the gap, we propose a *semi-supervised* approach for entity typing, dubbed ASSET, requiring only little labeled training data. Our approach leverages unlabeled data using the teacher-student framework [9, 13]: (i) we train a teacher model on labeled data, (ii) we use the teacher model to generate *pseudo-labels* on unlabeled data, (iii) we train a student model on the combination of labeled and pseudo-labeled data. We repeat the process by treating the student as a new teacher to re-label the unlabeled data and to train a new student. To the best of our knowledge, our approach is the first to adapt semi-supervised learning to the entity typing task and distills knowledge from unlabeled data to boost its performance. Our evaluation on the two benchmarking datasets FB15k-ET and YAGO43k-ET shows that given a small amount of training data, our approach significantly outperforms supervised baselines trained on the same labeled training data.

2 RELATED WORK

Entity Typing in Knowledge Graphs. Approaches for entity typing in knowledge graphs can be distinguished by their features and models: For instance, Melo et al. [10] employ the incoming and outgoing relations of an entity to train a hierarchical multi-label classifier. Similarly, Xu et al. [17] predict the types of Chinese

Algorithm 1: Our Teacher-Student Framework (ASSET)

```
Require: labeled and unlabeled datasets: $\mathscr{D}_l$, $\mathscr{D}_u$.
Require: teacher and student models: $\mathcal{T}(\theta^t)$, $\mathcal{S}(\theta^s)$.
1  for num of epochs do
2      Sample batches $\beta_l$ from $\mathscr{D}_l = \{(x_i, y_i)\}, x_i \in \mathbb{R}^d$;
3      Train teacher model $\mathcal{T}_i(\theta^t)$ on $\beta_l$;
4      Calculate $\mathcal{L}_{\mathscr{D}_l}$ by Equation (1) on $\beta_l$;
5  Use $\mathcal{T}_i(\theta^t)$ to infer pseudo-labels $\tilde{y}_i$ for $\mathscr{D}_u$ and let $\tilde{\mathscr{D}}_u$ be the
   dataset $\mathscr{D}_u$ together with pseudo-labels;
6  for num of epochs do
7      Sample batches $\beta_{l+u}$ from $\mathscr{D}_l$ and $\tilde{\mathscr{D}}_u$;
8      Train student model $\mathcal{S}_i(\theta^s)$ on $\beta_{l+u}$;
9      Calculate $\mathcal{L}_{\mathscr{D}_u}$ by Equation (2) on $\beta_{l+u}$;
10 Replace teacher model with student model $\mathcal{T}_{i+1} = \mathcal{S}_i$;
11 Repeat from Step 1 until student model $\mathcal{S}_{i+1}$ has converged.;
```

entities by linking them to the English DBpedia and employing DBpedia's property and category information within a multi-label hierarchical classifier. Neelakantan and Chang [12] employ textual descriptions of entities as features in combination with a linear classifier. While there has been a lot of research on knowledge graph embeddings and link prediction [14], most of the approaches focus on predicting non-type links, and the employed benchmarking datasets, e.g., FB15k-237, WN18RR, and YAGO3-10 do not contain type information. Only Moon et al. [11] and Zhao et al. [21] proposed approaches for embedding entities according to their types. All of these approaches assume a large number of labeled training samples, which might not be available and expensive to obtain. In contrast, we employ a semi-supervised approach requiring only a little training data.

Semi-supervised Learning. Early semi-supervised algorithms employ *hard* pseudo-labeling techniques to make 0,1-predictions for unlabeled data and then train a machine learning model on a combination of *pseudo-labeled* and *labeled* data to improve its performance. Recently, the exploration of a teacher-student framework and *soft* pseudo-labels representing probabilities has produced impressive results for image classification [16]. However, few studies employ this paradigm in the context of natural language processing [4] and to the best of our knowledge, it has not been employed for knowledge graphs and the entity typing task.

3 SEMI-SUPERVISED TYPE PREDICTION

3.1 Problem Formulation

We formulate the task of entity typing as a *multi-label classification* problem, where each entity can have one or more types simultaneously. Formally, let $\mathcal{E} = \{e_1, e_2, \ldots, e_z\}$ be a set of z entities and $\Lambda = \{\lambda_1, \lambda_2, \ldots, \lambda_k\}$ be a set of k types in a KG. Each $e_i \in \mathcal{E}$ can be associated with a subset of labels $\mathbf{y_i} \in 2^{\Lambda}$. We identify a set of relevant labels with a binary vector $\mathbf{y_i} = (y_i^{(1)}, y_i^{(2)}, \ldots, y_i^{(k)})$, where $y_i^{(j)} = 1$ means that e_i has type λ_j. Our goal is to learn a multi-label model $\mathbf{H} : \mathcal{E} \to 2^{\Lambda}$ that maps each entity $e_i \in \mathcal{E}$ to a set of relevant types $\mathrm{y_i} \in 2^{\Lambda}$. We assume a semi-supervised setting in which there are two datasets: (i) *labeled* data $\mathscr{D}_l = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathbb{R}^d$ is a d-dimensional vector of entity e_i—in our case, pre-trained ConnectE [21] embeddings—and y_i denotes the vector of the corresponding labels; (ii) *unlabeled* data $\mathscr{D}_u = \{(x_j)\}_{j=1}^M$. In the following section, we describe the details of our teacher-student framework that leverages knowledge distillation [7] and generate pseudo-labels for unlabeled data $\mathscr{D}_u$.

3.2 The Teacher-Student Framework

Our semi-supervised approach is based on self-training [13] in the teacher-student framework. We formally describe the overall procedure in Algorithm 1. The inputs of our approach are labeled and unlabeled datasets $\mathscr{D}_l$ and $\mathscr{D}_u$, respectively, from which we sample batches. First, we train a teacher model $\mathcal{T}(\theta_t)$ only on labeled data $\mathscr{D}_l$ and compute the supervised loss $\mathcal{L}_{\mathscr{D}_l}$ using the cross-entropy function in Equation (1). Then, we employ the teacher model to generate *pseudo-labels* $\tilde{\mathbf{y}}_i$ for the unlabeled data $\mathscr{D}_u$ and we refer to the dataset along with pseudo-labels as $\tilde{\mathscr{D}}_u$. The pseudo-labels $\tilde{\mathbf{y}}_i^{(j)}$ are soft labels representing the probability of type λ_j being assigned to an entity e_i. Second, we train the student model $\mathcal{S}(\theta^s)$ on the combined dataset of *labeled* and *unlabeled* data to minimize the combined cross-entropy loss as illustrated in Equation (2). Finally, we iterate this paradigm by replacing the teacher model $\mathcal{T}_i$ with the student model $\mathcal{S}_i$ to generate new pseudo-labels and train a new student $\mathcal{S}_{i+1}$.

Teacher Model. We use a neural network with one fully-connected layer with 128 units and *ReLU* activations, and an output layer with sigmoid activation. We train the teacher model for at most 100 epochs using the ADAM optimizer. The model's hyperparameters are fine-tuned with grid search (see Section 4.1). To avoid over-fitting, we employ early-stopping [20]. The loss function is

$$\mathcal{L}_{\mathscr{D}_l} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{k} \left[y_i^{(j)} \log\left(\hat{y}_i^{(j)}\right) + \left(1 - y_i^{(j)}\right) \log\left(1 - \hat{y}_i^{(j)}\right) \right] \quad (1)$$

where y_i are the ground-truth types of e_i, $\hat{y}_i$ are the predicted types and N is the size of dataset $\mathscr{D}_l$.

Student Model. We employ a network similar to the teacher model with an additional dropout layer with rate 0.20; we optimize the hyperparameters with grid search. We train the student model for 100 epochs on batches of labeled and pseudo-labeled data. Each batch has 128 samples, and we employ the ADAM optimizer with early-stopping to avoid over-fitting. Our loss function is

$$\mathcal{L}_{\mathscr{D}_u} = \mathcal{L}_{\mathscr{D}_l} - \frac{1}{M} \sum_{i=1}^{M} \sum_{j=1}^{k} \left[\tilde{\mathbf{y}}_i^{(j)} \log\left(\hat{y}_i^{(j)}\right) + \left(1 - \tilde{\mathbf{y}}_i^{(j)}\right) \log\left(1 - \hat{y}_i^{(j)}\right) \right] \quad (2)$$

where $\tilde{\mathbf{y}}_i$ denote the pseudo-labels generated by the teacher model, $\hat{y}_i$ the predicted labels by the student model, and M the size of dataset $\mathscr{D}_u$.

4 EXPERIMENTS

We conduct a set of experiments to answer the following research questions:

RQ.1. How effective is our semi-supervised approach compared to state-of-the-art baselines employing the same number of labeled samples?

RQ.2. How does the effectiveness depend on the number of training samples?

Table 1: Statistics of datasets: FB15k-ET and YAGO43k-ET.

	FB15k-ET			YAGO43k-ET		
	Top 3	Top 5	Top 10	Top 3	Top 5	Top 10
Relations	1,345	1,345	1,345	37	37	37
Total Type Triples	22,849	26,184	29,058	29,528	32,193	35,225
Train Type Triples	12,748	13,726	14,376	24,078	25,792	27,201
Valid Type Triples	5,038	6,200	7,321	2,689	3,170	3,997
Test Type Triples	5,063	6,258	7,361	2,761	3,231	4,027

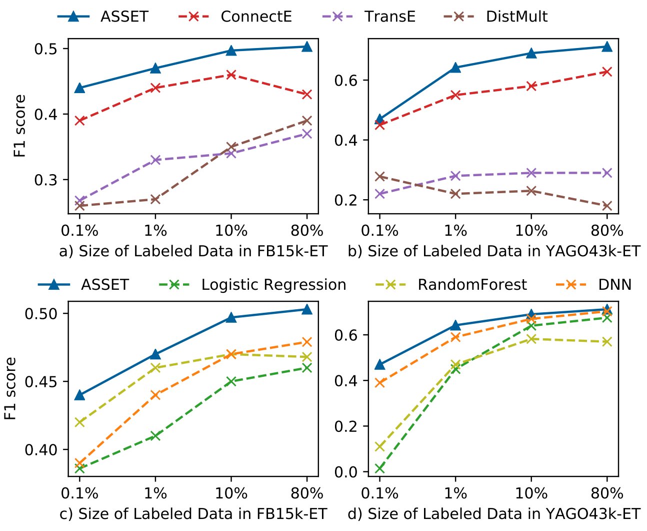

Figure 1: Plots (a) and (b) compare the performance of our approach ASSET with embedding baselines, plots (c) and (d) with supervised baselines. All models predict the top 10 types on FB15k-ET and YAGO43k-ET.

4.1 Experimental Setup and Implementation

Datasets. Table 1 gives an overview of our benchmarking datasets. FB15k-ET [21] consists of the benchmarking dataset FB15k-237 along with type triples of the form (*entity, entity type*). We use the same dataset split (*train-valid-test*) as Zhao et al. [21] to ensure the same evaluation setting. In particular, we use three subsets: train (136,618 triples), valid (15,749 triples) and test (15,780 triples). Similarly, YAGO43k-ET [21] enriches the benchmarking dataset YAGO43k with type information, and we use the same data split: train (375,853 triples), valid (42,750 triples), test (45,182 triples). For both datasets, we determine the $k \in \{3, 5, 10\}$ most frequent types and employ the subset of entities and triples induced by them for our experiments.

Embeddings and Supervised Baselines. We employ three KG embedding models as baselines: TransE-ET [2], DistMult-ET [18], and ConnectE [21]. To predict the types of an entity, we infer its closest entity in the embedding space according to Euclidean distance and use its types. On top of the best-performing embeddings (ConnectE), we employ three supervised classifiers as baselines: Logistic Regression, RandomForest, and a Deep Neural Network (DNN).

Evaluation Metrics. We consider the following metrics for multi-label classification [15]: the *hamming loss* ($\mathcal{H}_{loss}$) denotes the fraction of wrongly predicted labels per instance (lower is better) and the F_1-score denotes the harmonic mean of precision and recall of correctly predicted labels per instance (higher is better).

Hyperparameters. We perform a grid search for the hyperparameters of our semi-supervised approach ASSET within the ranges `batch-size` = {128, 256, 512}, `learning-rate` = {0.001, 0.01}, `dropout` = {0.10, 0.20, 0.25, 0.30}, and gradient `optimizer` = {ADAM, RMSPROP, ADAGRAD}. We find the following values to yield the best performance in terms of F_1-score: `batch-size`=128, `learning-rate`=0.001, `dropout`=0.20 and `optimizer`=ADAM. For ConnectE, we set the hyperparameters as described in its paper [21]. For TransE-ET and DistMult-ET, we employ a dimension size of 200 and the default parameters of the PyKeen[1] library. For Logistic Regression and RandomForest, we set their hyperparameters to the default values of the Scikit-learn library. For DNN, we employ the same hyperparameters as for ASSET.

Reproducibility. We provide our source code and datasets on GitHub.[2] ASSET and DNN are based on TensorFlow version 2.0. The implementation of ConnectE was obtained from its repository[3] and TransE-ET and DistMult-ET were obtained from PyKeen version 1.5. For Logistic Regression and RandomForest, we employ the implementation from Scikit-learn version 0.24.

4.2 Discussion and Results

Performance Comparison (RQ.1). Table 2 compares our approach both with embedding and supervised baselines on 1% of the two datasets FB15k-ET and YAGO43k-ET with varying numbers of entity types (Top 3, Top 5, and Top 10). We observe that—given such little training data—our semi-supervised approach ASSET significantly outperforms each of the baselines in terms of both $\mathcal{H}_{loss}$ and F_1-score with $p < 0.03$.[4] We attribute this to our teacher-student paradigm that augments the original training dataset with *pseudo-labeled* data from the unlabeled dataset, boosting overall performance (as discussed in Algorithm 1).

Among the embedding approaches, ConnectE significantly outperforms the other KG embeddings TransE-ET and DistMult-ET in terms of both $\mathcal{H}_{loss}$ and F_1-score. For example, differences in terms of $\mathcal{H}_{loss}$ range from -0.05 (TransE-ET, YAGO43k-ET, Top 10) to -0.26 (DistMult-ET, YAGO43K-ET, Top 3) and differences of F_1-scores range from +0.10 (TransE-ET, FB15k-ET, Top 10) to +0.39 (DistMult-ET, YAGO43k-ET, Top 3). Our results corroborate previous findings [21] that ConnectE outperforms other embedding models for the entity typing task, and we employ ConnectE embeddings as feature representation to train our supervised baselines (Logistic Regression, RandomForest, and DNN). Among the supervised approaches, the differences are less pronounced and the DNN achieves comparable performance to Logistic Regression and RandomForest. For example, the DNN is at least as good as Logistic

[1]https://github.com/pykeen/pykeen
[2]https://github.com/dice-group/ASSET
[3]https://github.com/Adam1679/ConnectE
[4]For each pair of approaches, we employ a two-sided Wilcoxon signed-rank test between the $\mathcal{H}_{loss}/F_1$-scores on Top 3, Top 5, and Top 10 of FB15k-ET and YAGO43k-ET datasets of two approaches. Our null hypothesis is that the two approaches produce $\mathcal{H}_{loss}/F_1$-scores from the same distribution.

Table 2: **Evaluation results on FB15k-ET and YAGO43k-ET datasets with 1% labeled training data. A lower value of $\mathcal{H}_{loss}$ indicates better performance, for the other metrics higher values are better. Best results are in bold.**

		FB15k-ET						YAGO43k-ET					
		Top 3		Top 5		Top 10		Top 3		Top 5		Top 10	
		$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1	$\mathcal{H}_{loss}$	F_1
Embeddings	TransE-ET	0.34	0.60	0.42	0.40	**0.26**	0.34	0.35	0.48	0.28	0.34	0.17	0.28
	DistMult-ET	0.40	0.54	0.37	0.42	0.29	0.31	0.40	0.41	0.29	0.30	0.19	0.22
	ConnectE	0.26	0.73	0.37	0.55	0.32	0.44	0.14	0.80	0.14	0.69	0.12	0.55
Supervised	Logistic Regression	0.25	0.72	0.35	0.56	0.35	0.41	0.06	0.90	0.10	0.77	0.12	0.62
	RandomForest	0.26	0.72	0.34	0.57	**0.26**	0.46	0.09	0.76	0.11	0.60	**0.08**	0.46
	DNN	0.26	0.73	0.34	0.56	0.29	0.44	0.09	0.80	0.11	0.66	**0.08**	0.61
Semi-supervised	ASSET (*Teacher-Student*)	**0.24**	**0.74**	**0.33**	**0.59**	0.28	**0.47**	**0.04**	**0.93**	**0.09**	**0.80**	0.11	**0.64**

Regression in 6 out of 12 measurements and at least as good as RandomForest in 9 out of 12 measurements.

Training Size Effect on Performance (RQ.2). We conduct a set of experiments with different *labeled-unlabeled* ratios to assess the effect of training size on the performance. In particular, we use four ratios of labeled data (0.1%, 1%, 10%, and 80% of train split) of the FB15k-ET and YAGO43k-ET datasets. In Figure 1, plots (a) and (b), we compare the performance of our approach with the embedding models TransE-ET, DistMult-ET, and ConnectE. As we can see, ASSET outperforms all embedding models at all ratios of labeled data. In Figure 1, plots (c) and (d) show F_1-scores of our approach against supervised methods on both datasets. With a small labeled dataset (0.1% and 1%), our approach outperforms all baseline methods with large margins. When the size of labeled data increases (e.g., 10%, 80%), the supervised methods increasingly achieve competitive performance to our approach—particularly, on the YAGO43k-ET dataset. The slight decrease in performance of RandomForest appears to be due to overfitting.

5 CONCLUSION

We propose a novel approach for knowledge graph entity typing leveraging semi-supervised learning. Our approach alleviates the problem of few training samples by employing a teacher-student paradigm to learn from *labeled* and *unlabeled* data. The task of a teacher model is to generate pseudo-labels for unlabeled data. Then, the student model is trained on the combination of both labeled and pseudo-labeled data. We conduct several experiments on two benchmarking datasets. Our results demonstrate the effectiveness of our approach ASSET compared with state-of-the-art baselines. In future work, we plan to employ our approach on Wikidata and help its community to predict newly introduced types with little training data.

ACKNOWLEDGEMENT

This work has been supported by the German Federal Ministry of Education and Research (BMBF) within the project DAIKIRI under the grant no 01IS19085B.

REFERENCES

[1] Russa Biswas, Radina Sofronova, Mehwish Alam, Nicolas Heist, Heiko Paulheim, and Harald Sack. 2021. Do Judge an Entity by Its Name! Entity Typing Using Language Models. In *ESWC (Satellite Events)*, Vol. 12739. 65–70.

[2] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*. 2787–2795.

[3] Bo Chen, Xiaotao Gu, Yufeng Hu, Siliang Tang, Guoping Hu, Yueting Zhuang, and Xiang Ren. 2019. Improving Distantly-supervised Entity Typing with Compact Latent Space Clustering. In *NAACL-HLT*. 2862–2872.

[4] Yun Chen, Yang Liu, Yong Cheng, and Victor O. K. Li. 2017. A Teacher-Student Framework for Zero-Resource Neural Machine Translation. In *ACL*. 1925–1935.

[5] Zhe Chen, Yuehan Wang, Bin Zhao, Jing Cheng, Xin Zhao, and Zongtao Duan. 2020. Knowledge Graph Completion: A Review. *IEEE Access* 8 (2020), 192435–192456.

[6] Yuanfei Dai, Shiping Wang, Neal N Xiong, and Wenzhong Guo. 2020. A survey on knowledge graph embedding: Approaches, applications and benchmarks. *Electronics* 9, 5 (2020).

[7] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the Knowledge in a Neural Network. *CoRR* abs/1503.02531 (2015).

[8] Hailong Jin, Lei Hou, Juanzi Li, and Tiansi Dong. 2019. Fine-Grained Entity Typing via Hierarchical Multi Graph Convolutional Networks. In *EMNLP/IJCNLP*. 4969–4978.

[9] Dong-Hyun Lee et al. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML2013*, Vol. 3. 896.

[10] André Melo, Johanna Völker, and Heiko Paulheim. 2017. Type Prediction in Noisy RDF Knowledge Bases Using Hierarchical Multilabel Classification with Graph and Latent Features. *Int. J. Artif. Intell. Tools* 26, 2 (2017), 1760011:1–1760011:32.

[11] Changsung Moon, Paul Jones, and Nagiza F. Samatova. 2017. Learning Entity Type Embeddings for Knowledge Graph Completion. In *CIKM*. 2215–2218.

[12] Arvind Neelakantan and Ming-Wei Chang. 2015. Inferring Missing Entity Type Instances for Knowledge Base Completion: New Dataset and Methods. *CoRR* abs/1504.06658 (2015).

[13] Jesper E. van Engelen and Holger H. Hoos. 2020. A Survey on Semi-Supervised Learning Techniques. *Mach. Learn.* 109, 2 (2020), 373–440.

[14] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743.

[15] Xi-Zhu Wu and Zhi-Hua Zhou. 2017. A Unified View of Multi-Label Performance Measures. In *ICML (Proceedings of Machine Learning Research, Vol. 70)*. 3780–3788.

[16] Qizhe Xie, Minh-Thang Luong, Eduard H. Hovy, and Quoc V. Le. 2020. Self-Training With Noisy Student Improves ImageNet Classification. In *CVPR*. 10684–10695.

[17] Bo Xu, Yi Zhang, Jiaqing Liang, Yanghua Xiao, Seung-won Hwang, and Wei Wang. 2016. Cross-Lingual Type Inference. In *DASFAA*, Vol. 9642. 447–462.

[18] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *ICLR (Poster)*.

[19] Xuchen Yao and Benjamin Van Durme. 2014. Information Extraction over Structured Data: Question Answering with Freebase. In *ACL*. 956–966.

[20] Yuan Yao, Lorenzo Rosasco, and Andrea Caponnetto. 2007. On Early Stopping in Gradient Descent Learning. *Constructive Approximation* 26, 2 (2007), 289–315.

[21] Yu Zhao, Anxiang Zhang, Ruobing Xie, Kang Liu, and Xiaojie Wang. 2020. Connecting Embeddings for Knowledge Graph Entity Typing. In *ACL*. 6419–6428.