Parasitic Neural Network for Zero-Shot Relation Extraction

Shengbin Jia*
Tongji University
Shanghai, China
shengbinjia@tongji.edu.cn

Shijia E
Tencent Ins
Shanghai, China
allene@tencent.com

Yang Xiang
Tongji University
Shanghai, China
shxiangyang@tongji.edu.cn

ABSTRACT

Conventional relation extraction methods can only identify limited relation classes and not recognize the unseen relation types that have no pre-labeled training data. In this paper, we explore the zero-shot relation extraction to overcome the challenge. The only requisite information about an unseen type is the label name. We propose a Parasitic Neural Network (PNN) where unseen types are parasitic on seen types to get automatic annotation and training. The model learns a mapping between the feature representations of text samples and the distributions of unseen types in a shared semantic space. Experiment results show that our model significantly outperforms others on the unseen relation extraction task and achieves effect improvement more than 20% when there are not any manual annotations or additional resources.

KEYWORDS

relation extraction, zero-shot, neural network, knowledge graph

ACM Reference format:
Shengbin Jia, Shijia E, and Yang Xiang. 2016. Parasitic Neural Network for Zero-Shot Relation Extraction. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 5 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Relation Extraction (RE) task aims to determine relational facts from the unstructured text and can populate knowledge bases or benefit downstream knowledge-driven applications. The conventional methods (including one/few -shot learning) [2, 5] cannot meet practical needs of the relation extraction. Generally, there are massive fine-grained types of relations in the real world. However, these methods are often to distinguish the limited relational taxonomy, where the relation types are seen and each type must have a certain number of pre-labeled samples. They are unable to generalize to new (unseen) relations (i.e., they will break down when predicting a type that has no training examples). Collecting sufficient labeled instances for training on all expected categories is almost impossible, in contrast with the limited number of relation types covered by existing datasets.

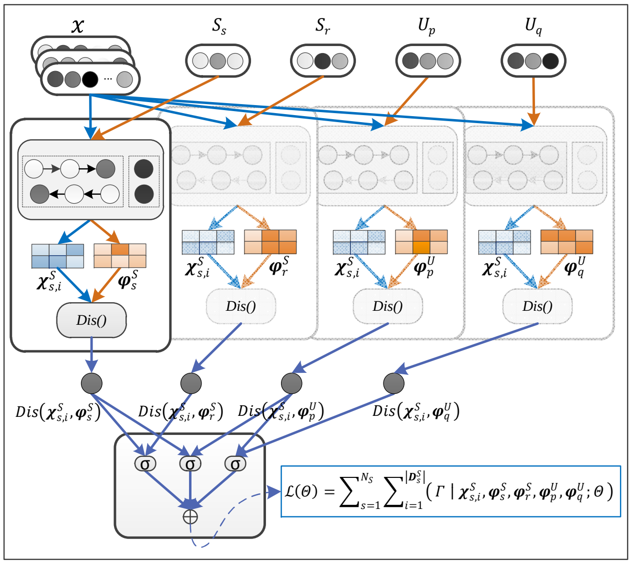
x
S_s
S_r
U_p
U_q
$\chi_{s,i}^S$
φ_s^S
φ_r^S
φ_p^U
φ_q^U
Dis()
Dis()
Dis()
Dis()
$Dis(\chi_{s,i}^S, \varphi_s^S)$
$Dis(\chi_{s,i}^S, \varphi_r^S)$
$Dis(\chi_{s,i}^S, \varphi_p^U)$
$Dis(\chi_{s,i}^S, \varphi_q^U)$
σ
σ
σ
⊕
$\mathcal{L}(\Theta) = \sum_{s=1}^{N_S} \sum_{i=1}^{|D_s^S|} (\Gamma \mid \chi_{s,i}^S, \varphi_s^S, \varphi_r^S, \varphi_p^U, \varphi_q^U; \Theta)$

Figure 1: The architecture of the parasitic neural network.

To address the challenge, we develop a Zero-Shot Relation Extraction (ZRE), which is under the restriction that the extractor should identify facts of new relation types after learning from limited labeled instances of seen types. The ZRE is a promising learning paradigm by reducing annotation costs and improving application efficiency. However, it is immature and has received limited attention. The existing popular methods address the ZRE task to develop specific transfer learning procedures by reading comprehension [6], textual entailment [10], and so on. We consider these methods to be **indirect-trick**. They need much unnatural descriptive information to improve the understandability of relation types. Annotation costs severely decrease their applicability to new types. In this paper, we are committed to the **direct-trick** method. It does not need any manual intervention to pre-describe relation types. Instead, it just uses the names of type labels that is a natural expression of relation semantics.

Furthermore, we raise a zero-shot learning framework that learns the mapping between the text feature representations and the relation type embeddings (prototypes) in a shared semantic space. To prevent over fitting seen types and successfully adapt to unseen types, the model requires to solve a principal problem: how to understand the distributions of unseen relation types in the shared space. Therefore, we propose the Parasitic Neural Network model.

In summary, our key contributions are presented as follows. (1) We develop a general zero-shot learning framework for unseen relation extraction by the direct-trick. It emphasizes to use no

[*] This author is the one who did all the really hard work.

Conference'17, Washington, DC, USA
© 2016 ACM. 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

manual annotations or external knowledge. (2) Based on the Parasitism, we propose the PNN that leverages the association of the relation types in a shared semantic space to learn the distributions of unseen types automatically. (3) Our experiment results achieve significant improvement than other direct-trick methods and most indirect-trick methods.

2 RELATED WORK

Most of the works on the zero-shot learning were focused on the area of computer vision [12, 17]. In the area of nature language processing, the applications of zero-shot learning have been emerging in recent years, such as entity typing [11], event extraction [3] and knowledge graph completion [1, 13].

As for zero-shot relation extraction, it is immature and has received limited attention. By analyzing linguistics, old-fashioned approaches developed unsupervised models (e.g., clustering) based on the combinations of manual features, patterns or corpus-level resources [4, 9, 18]. They tended to be inefficient and consumed much manpower. The recent methods were to transfer other tasks to produce relations. Levy et al. [6] formulated relation types as various parametrized natural-language questions, then used a reading comprehension model to process the questions to obtain relation facts. By considering the text and the relation description as the premise and hypothesis respectively, Obamuyide et al. [10] transformed the extraction task to determine the truthfulness of the hypothesis by a textual entailment model. They were expensive to manually formulate reading comprehension questions or entailment rules. In addition, transfer-based methods were constrained by the capability of indirect tasks whose errors or defects could be cascaded into the relation extraction.

In this paper, we take a universal and all-inclusive manner, which is to model the mapping between text instances and relation type prototypes [15, 16]. More importantly, we explore the ZRE via the direct-trick. In view of the extremely scarce type information, we set up parasitic learning.

3 METHODOLOGY

3.1 Parasitism Thought

Let $\boldsymbol{S} = \{S_s \mid s = 1, \cdots, N_S\}$ denotes a set of seen relation types and $\boldsymbol{U} = \{U_u \mid u = 1, \cdots, N_U\}$ unseen types, with $\boldsymbol{S} \cap \boldsymbol{U} = \emptyset$. Suppose that the dataset $\mathbb{D} = \mathbb{D}^S \cap \mathbb{D}^U$ is a collection of text instances. The $\mathbb{D}^S_s = \left\{x^S_{s,i} \mid y^S_s = S_s\right\}$ is as the set of labeled training instances belonging to seen types S_s. The $\mathbb{D}^U = \left\{x^U_j\right\}$ is as the set of testing instances, meanwhile, $y^U_j \in \boldsymbol{U}$ is to be predicted as the corresponding type labels for x^U_j. In semantic embedding space $\mathbb{R}^z$, the instance x will be embedded to $\boldsymbol{\chi}$ and it is assumed to belong to one category. The types will be vectorized as type prototypes $\boldsymbol{\varphi} = \left\{\boldsymbol{\varphi}^S, \boldsymbol{\varphi}^U\right\}$. Overall, the ZRE learning task is defined as: Given $\mathbb{D}^S$, the ZRE system learns the mapping $f(\cdot) : \boldsymbol{\chi} \longrightarrow \boldsymbol{\varphi}$, which can classify testing instances $\mathbb{D}^U$ (i.e., to predict y^U).

The instances with the same relation type will cluster around a single prototype in the shared semantic space, whereas they are far away from other type prototypes. Meanwhile, the more similar types are distributed closer in the space [15, 16]. Therefore, we determine the semantic distance $Dis(\cdot)$ between the feature representation $\boldsymbol{\chi}$ and the type prototype $\boldsymbol{\varphi}$. Here, the semantic distance is a quantification of the mapping $f(\cdot)$. The smaller the distance, the better the mapping fit.

Algorithm 1 Parasitic neural network training algorithm.

```
Require: $\boldsymbol{S}, \mathbb{D}^S, \boldsymbol{\varphi}^S, \boldsymbol{U}, \boldsymbol{\varphi}^U$.
1: Calculate the semantic distances of seen types $\boldsymbol{S}$ to unseen types $\boldsymbol{U}$, as,
       $\boldsymbol{D}(S_s, U_u) = Dis(\boldsymbol{\varphi}^S_s, \boldsymbol{\varphi}^U_u) \mid s = 1, ..., N_S,\ u = 1, ..., N_U$.
2: Obtain the array $\boldsymbol{R}$ by ranking the $\boldsymbol{D}(\boldsymbol{S}, \boldsymbol{U})$ (from small to large),
       $\forall\{ s = 1, ..., N_S,\ m = 1, ..., N_U{-}1\}$ s.t.
       $\boldsymbol{R}[s][m] \in \boldsymbol{U} \wedge \boldsymbol{D}(S_s, \boldsymbol{R}[s][m]) \leqslant \boldsymbol{D}(S_s, \boldsymbol{R}[s][m+1])$.
3: for $S_s$ (as Host) in $\boldsymbol{S}$ do
4:     for $x^S_{s,i}$ in $\mathbb{D}^S_s$ do
5:         Select $S_r$ from $\boldsymbol{S}$ randomly;
6:         Select any $U_p = \boldsymbol{R}[s][p]$ from $\boldsymbol{R}[s][1 : N_U{-}1]$ as Parasite;
7:         Select any $U_q = \boldsymbol{R}[s][q]$ from $\boldsymbol{R}[s][p : N_U]$ as Parasite.
8:         Construct four sets of inputs for PNN sub-networks, as
           $(x^S_{s,i},\ S_s), (x^S_{s,i},\ S_r), (x^S_{s,i},\ U_p), (x^S_{s,i},\ U_q)$.
9:         Run PNN to
10:        obtain the $\boldsymbol{\chi}^S_{s,i}$ of instance $x^S_{s,i}$;
11:        obtain the corresponding prototypes $\boldsymbol{\varphi}^S_s, \boldsymbol{\varphi}^S_r, \boldsymbol{\varphi}^U_p, \boldsymbol{\varphi}^U_q$;
12:        calculate $Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^S_s)$, $Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^S_r)$, $Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^U_p)$,
           $Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^U_q)$.
13:        Minimize the Joint energy function in Eq. 2.
14:    end for
15: end for
```

Furthermore, we can establish the following assumptions of the **premise**: (1) Given any relation type R_1 and a corresponding instance x, it should be sure that the semantic distance between x and R_1 is the smallest (or even 0), compared with the distance between x and any other types. (2) For arbitrarily given type R_2 and type R_3 ($R_1 \neq R_2 \neq R_3$), if the semantic distance between R_2 and R_1 is smaller than that between R_3 and R_1, the semantic distance between R_2 and x should be smaller than that between R_3 and x.

The above premises imply the association among the relation types in the shared semantic space. According to this correlation, we can create annotations for unseen types (*Parasite*) by considering the instances of seen types as *Host*, just like "Parasitism". Algorithm 1 (lines 1 to 8) shows the process of data creation. Then, we train the PNN model to learn the unseen types' distributions.

Joint Energy Function. As described in Algorithm 1, each sub-network produces a semantic distance metric. They interact with each other and then joint together. In detail, we establish a series of trunks, shaped like $\{Branch_1, Branch_2\}$, including $\{Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^S_s), Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^S_r)\}$, $\{Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^S_s), Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^U_p)\}$, and $\{Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^U_p), Dis(\boldsymbol{\chi}^S_{s,i}, \boldsymbol{\varphi}^U_q)\}$. According to the premises mentioned above, we are to compare the semantic distance between the two in each trunk. We are motivated by the triplet loss [12], where we set the σ function to ensure that the $Branch_1$ is smaller than the $Branch_2$ by at least a margin m, as,

$$\sigma(Branch_1, Branch_2, m) = max(Branch_1 - Branch_2 + m, 0). \quad (1)$$

Thus, the joint energy function is defined as,

$$\mathcal{L}(\Theta) = \sum_{s=1}^{N_S} \sum_{i=1}^{|\mathbb{D}_s^S|} (\Gamma \mid \boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_s^S,\ \boldsymbol{\varphi}_r^S,\ \boldsymbol{\varphi}_p^U,\ \boldsymbol{\varphi}_q^U; \Theta), \tag{2}$$

$$\Gamma = \begin{array}{c} \sigma(Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_s^S),\ Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_r^S),\ m_1) + \\ \beta\ \sigma(Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_s^S),\ Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_p^U),\ m_2) +, \\ \gamma\ \sigma(Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_p^U),\ Dis(\boldsymbol{\chi}_{s,i}^S, \boldsymbol{\varphi}_q^U),\ m_3) \end{array} \tag{3}$$

where we employ the cosine distance (within the range [0, 2]), β and γ are the trade-off parameters.

3.2 Network Architecture

As shown in Figure 1, the PNN consists of four sub-networks that accept distinct inputs but are then joined by a joint energy function.

The parameters between the sub-networks are tied, that is, each network computes the same metric on a shared workbench (shared by *Host* and *Parasite*, this seems to be a parasitic energy community). Tying guarantees that two inputs of an identical class cannot be mapped by their respective networks to very different locations in the semantic space, and each sub-network can also distinguish inputs of varied types.

Text Embedding. The sub-network takes as input one piece of text and a relation label, the text contains the head and tail entities of a candidate relation. We transform the text instance x into its distributed representation $\boldsymbol{x}$ by adopting triple embeddings $\{\boldsymbol{x}^w, \boldsymbol{x}^c, \boldsymbol{x}^p\}$. The $\boldsymbol{x}^w$ denotes the word embedding. To deal with unregistered words, we use a convolutional neural network to encode its character embedding $\boldsymbol{x}^c$, as [8] doing. The $\boldsymbol{x}^p$ represents the position embedding to specify entity pairs. Similarly to [7], it is defined as the combination of the relative distances from the current word to head or tail entities.

Relation Type Prototype. We achieve the prototype $\boldsymbol{\varphi}$ with the word embeddings of type labels' names. Word vectors capture distributional similarities from a large text corpus. Each prototype is an average of word embeddings of the core words (i.e. nouns, adjectives, etc., except prepositions, conjunctions) in its label name. We can fine-tune these embeddings along with training.

Learning Feature Representation from Text. The sample text has latent feature information that is category-invariant. We feed the text embeddings into the bidirectional Ordered Neurons Long Short-Term Memory Network (ONLSTM) [14] to encode feature representation $\boldsymbol{\chi}$. The ONLSTM performs tree-like syntactic structure composition operations on a sentence without destroying its sequence form. It can learn temporal semantics, meanwhile, capture potential syntactic information involved in natural language. Notably, the syntax is necessary for relation extraction to acquire the associations among entities and relational phrases.

Once the model is optimized, we determine the possible relation that a test instance x_j^U may represent, if any. The top ranked prediction from the candidate predicted types $\boldsymbol{U}$, denoted as $C(x_j^U, 1)$, is given by:

$$C(x_j^U, 1) = argmin\ Dis(\boldsymbol{\chi}_j^U, \boldsymbol{\varphi}_u^U), \ \ u = 1, 2, \ldots, N_U \tag{4}$$

Moreover, $C(x_j^U, K)$ denotes the K^{th} most probable relation type predicted for x_j^U.

4 EXPERIMENTS

4.1 Settings

Dataset. We evaluate models using the zero-shot relation extraction dataset of [6]. It consists of 120 relation types that is from the knowledge base Wikidata. We use the positive labeled relation instances in this dataset. There are 225,060 samples. By applying a similar process to [6] and [10], we randomly select 24 classes as a testing set, 10 classes as the dev set, and the rest as the training set. The results reported for each experiment are the average taken over five runs with independent random initializations. Given different thresholds regarding distance, we can measure the precision (P), recall (R), and F1 of the results. We report the optimal values.

Hyperparameters. We implement the neural network by the Keras. The word embedding is from the GloVe with 100 dimensions. The character embedding is initialized randomly as 50 dimensions. The size of the ONLSTM unit is 100. Parameter optimization is performed with Adam optimizer. To mitigate over-fitting, we apply the dropout and early-stopping methods. Besides, we set m_1=0.1, m_2=0.1, m_3=0.08, β= γ=1.

Comparison Systems. We examine several major components in our model. (1.1) We test the influence of word embedding on prototypes, increasing noise by randomly zeroing its value in varied proportions. (1.2) We compare the bidirectional ONLSTM to the bidirectional LSTM. (1.3) We verify the choice of distance, including logistic regression probability (*LR*), euclidean distance (*EU*), and cosine distance (*COS*). We compare our PNN-based systems to external systems. (2.1) *120-Softmax* is a conventional 120-dimensional softmax classifier, but we only use seen types to train it. (2.2) *NaiveMAP* learns the mapping between the samples and seen types, by using the single mapping distance as loss directly (*Single*) [13], or by adopting a tied network with triplet loss (*Triplet*) [3]. (2.3) Model of Levy et al. [6] [1] is via reading comprehension, by using different descriptions for relation types (i.e., *NL* - the label's name, *SQ* - only a single question template per relation type, *MQ* - multiple questions, and *QE* - an ensemble learning way). (2.4) Model of Obamuyide et al. [10] is based on textual entailment, where *TE* transforms external entailment corpus for training, and *MD* represents training with manual annotations.

4.2 Results and Analysis

Ablation Study. The upper part of Table 1 shows our PNN-based models with different factors. The relation prototype is crucial to the model, and it depends on the quality of the embedding. But don't worry, just using the usual embedding GloVe, we have achieved the F1 value of 58%. Compared with the LSTM, the ONLSTM improves model performance by 7%. It shows that the potential syntactic information captured by the ONLSTM is pretty useful for relation extraction. However, the LSTM explicitly imposes a chain structure that cannot discern the hierarchical syntactic knowledge. The choice of distance metric is also important, where the cosine distance (COS) can well improve the effectiveness of a PNN.

[1] Notably, the methods of Levy et al. input an instance with head entity and relation (question) to predict tail entity (answer), however, our model inputs an instance with head and tail entities to predict relation. But from the perspective that we all aim to obtain relation triples, their results are valuable for reference.

Table 1: The performance of the PNN-based models (ablation study) and external systems (for comparison). $*$ indicates the model via the indirect-trick.

Models				P	R	F1
PNN	10%Noise	COS	ONLSTM	50.91	44.97	47.75
	5%Noise	COS	ONLSTM	57.68	49.48	53.26
	0Noise	COS	LSTM	58.47	45.45	51.13
	0Noise	COS	ONLSTM	**63.40**	**53.79**	**58.20**
	0Noise	LR	ONLSTM	57.96	43.28	49.55
	0Noise	EU	ONLSTM	61.75	52.32	56.64
120-Softmax (with ONLSTM)				3.30	3.30	3.30
NaiveMAP (with COS and ONLSTM)			Single	10.12	8.97	9.51
			Triplet	55.15	50.93	52.95
Levy et al. [6]			NL	40.50	28.56	33.40
			SQ $*$	37.18	31.24	33.90
			MQ $*$	43.61	36.45	39.61
			QE $*$	45.85	37.44	41.11
Obamuyide et al. [10]			TE $*$	-	-	44.38
			MD $*$	-	-	64.78

Table 2: The effects of our model trained with varying number of seen relation types. The hits@K represents the F1 of correct extractions ranked in the top K in eq. 4.

Proportion	Hit@K			
	K=1	K=2	K=3	K=5
22/86 (25%)	31.48	40.03	44.97	56.02
43/86 (50%)	43.55	52.68	56.88	63.68
65/86 (75%)	51.47	63.12	66.87	70.06
86/86 (100%)	58.20	66.55	70.36	73.47

Comparison with Other Methods. The lower part of Table 1 presents the results of several direct- or indirect- trick models. As for the models based on the direct-trick, our PNN (with ONLSTM and COS) remarkably outperforms others. As expected, the conventional classifier *120-Softmax* has almost no effect and is at the level of random guessing. The *NaiveMAP+single* is insufficient in a zero-shot setting since it cannot capture the association information between types. The *NaiveMAP+Triplet* tends to over-fit seen types. Our model can alleviate this over-fitting effectively by learning the semantic distributions of unseen relation types explicitly. Our model achieves effect improvement more than 20% than the model of Levy et al. when there are no manual annotations or additional resources. Most of the methods by using the indirect-trick are inferior to our model. These indirect-trick methods are constrained by extra annotation effort. The less quantity and lower quality of annotation information, the worse the models will perform.

Analyze the Impact of Training Set Size. Table 2 shows the results of our model after being trained with varying proportions of seen types. As the seen types in the training set increasing, the performance of unseen relation extraction will become better. The reason may be that the diversity of training set reduces the tendency of the model to over-fit seen types. In addition, most of the correct extractions appear in the front part (i.e. top K=5) of the candidate type ranks. It proves the validity of our premises, where the semantic distance between each sample and its corresponding prototype tends to be minimal.

Table 3: Examples of unseen relation type "father".

father (0)	named_after (0.361)	employer (0.644)	chairperson (0.889)
[Samuel Dirksz van Hoogstraten]$_{entity}$ trained first with his father [Dirk van Hoogstraten]$_{entity}$ and stayed in Dordrecht until about 1640.			
0.403	0.562	0.726	1.119
[Bertrade de Montfort]$_{entity}$ was the daughter of [Simon I de Montfort]$_{entity}$ and Agnes , Countess of Evreux.			
0.362	0.531	0.728	1.122

Case Study. We sample an unseen relation type "father" and its corresponding instances from the test set. The 1^{st} row of Table 3 presents several unseen types and their respective semantic distance from the target type "father". The 3^{rd} and 5^{th} rows of Table 3 show the semantic distances between each text instance and the relation types. The distance between each instance and the target type is minimum. Besides, the smaller the semantic distance between a relation type and the target type is, the smaller the semantic distance between it and the instance corresponding to the target type can be. Therefore, the conclusion of the test results is consistent with the premises mentioned above.

5 CONCLUSION

In this paper, we propose a general zero-shot relation extraction framework via the direct-trick to identify unseen relations. Furthermore, we propose the parasitic neural network. Inspired by parasitism, it owns a tied network structure and expands annotations automatically for unseen relation types to learn their distributions. The experiment performance is conspicuous. We will release the source code when the paper is openly available.

REFERENCES

[1] Orpaz Goldstein. 2018. *Zero-Shot relation extraction from word embeddings.* Ph.D. Dissertation. UCLA.

[2] Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. Fewrel: A large-scale supervised few-shot relation classification dataset with state-of-the-art evaluation. In *EMNLP*.

[3] Lifu Huang, Heng Ji, Kyunghyun Cho, Ido Dagan, Sebastian Riedel, and Clare Voss. 2018. Zero-Shot Transfer Learning for Event Extraction. In *ACL*. 2160–2170.

[4] Stanley Kok and Pedro Domingos. 2008. Extracting semantic networks from text via relational clustering. In *ECML*. 624–639.

[5] Shantanu Kumar. 2017. A survey of deep learning methods for relation extraction. *arXiv preprint arXiv:1705.03645* (2017).

[6] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. 2017. Zero-Shot Relation Extraction via Reading Comprehension. In *CoNLL*. 333–342.

[7] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL*. 2124–2133.

[8] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *ACL*. 1064–1074.

[9] Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. 2012. Ensemble semantics for large-scale unsupervised relation extraction. In *EMNLP*. 1027–1037.

[10] Abiola Obamuyide and Andreas Vlachos. 2018. Zero-shot Relation Classification as Textual Entailment. In *EMNLP Workshop on FEVER*. 72–78.

[11] Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-Based Zero-shot Fine-Grained Entity Typing. In *NAACL*. 807–814.

[12] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*. 815–823.

[13] Haseeb Shah, Johannes Villmow, Adrian Ulges, Ulrich Schwanecke, and Faisal Shafait. 2019. An Open-World Extension to Knowledge Graph Completion Models. In *AAAI*.

[14] Yikang Shen, Shawn Tan, Alessandro Sordoni, and Aaron Courville. 2019. Ordered neurons: Integrating tree structures into recurrent neural networks. In *ICLR*.

[15] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *NeurIPS*. 4077–4087.

[16] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *NeurIPS*. 935–943.

[17] Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM TIST* 10, 2 (2019).

[18] Yulan Yan, Naoaki Okazaki, Yutaka Matsuo, Zhenglu Yang, and Mitsuru Ishizuka. 2009. Unsupervised relation extraction by mining Wikipedia texts using information from the web. In *ACL*. 1021–1029.