SJTU-NLP at SemEval-2018 Task 9: Neural Hypernym Discovery with Term Embeddings

Zhuosheng Zhang[1,2], **Jiangtong Li**[3], **Hai Zhao**[1,2,*], **Bingjie Tang**[4]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China
[3]College of Zhiyuan, Shanghai Jiao Tong University, China
[4]School of Computer, Huazhong University of Science and Technology, China
{zhangzs, keep_moving-lee}@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn, alexistang@foxmail.com

Abstract

This paper describes a hypernym discovery system for our participation in the SemEval-2018 Task 9, which aims to discover the best (set of) candidate hypernyms for input concepts or entities, given the search space of a pre-defined vocabulary. We introduce a neural network architecture for the concerned task and empirically study various neural network models to build the representations in latent space for words and phrases. The evaluated models include convolutional neural network, long-short term memory network, gated recurrent unit and recurrent convolutional neural network. We also explore different embedding methods, including word embedding and sense embedding for better performance.

1 Introduction

Hypernym-hyponym relationship is an *is-a* semantic relation between terms as shown in Table 1. Various natural language processing (NLP) tasks, especially those semantically intensive ones aiming for inference and reasoning with generalization capability, such as question answering (Harabagiu and Hickl, 2006; Yahya et al., 2013) and textual entailment (Dagan et al., 2013; Roller and Erk, 2016), can benefit from identifying semantic relations between words beyond synonymy.

The *hypernym discovery* task (Camacho-Collados et al., 2018) aims to discover the most appropriate hypernym(s) for input concepts or entities from a pre-defined corpus. A relevant well-known scenario is *hypernym detection*, which is a binary task to decide whether a hypernymic relationship holds between a pair of words or not. A hypernym detection system should be capable of learning taxonomy and lexical semantics, including pattern-based methods (Boella and Caro, 2013; Espinosa-Anke et al., 2016b) and graph-based approaches (Fountain and Lapata, 2012; Velardi et al., 2013; Kang et al., 2016). However, our concerned task, hypernym discovery, is rather more challenging since it requires the systems to explore the semantic connection with all the exhausted candidates in the latent space and rank a candidate set instead of a binary classification in previous work. The other challenge is representation for terms, including words and phrases, where the phrase embedding could not be obtained by word embeddings directly. A simple method is to average the inner word embeddings to form the phrase embedding. However, this is too coarse since each word might share different weights. Current systems like (Espinosa-Anke et al., 2016a) commonly discover hypernymic relations by exploiting linear transformation matrix in embedding space, where the embedding should contain words and phrases, resulting to be parameter-exploded and hard to train. Besides, these systems might be insufficient to obtain the deep relationships between terms.

Hyponym	Hypernyms
Heming	actor, person, company
Kralendijk	town, city, provincial capital, capital
StarCraft	video game, pc game, computer game, videogaming, comic, electronic game, scientifiction

Table 1: Examples of hypernym-hyponym relationship.

Recently, neural network (NN) models have shown competitive or even better results than traditional linear models with handcrafted sparse fea-

* Corresponding author. This paper was partially supported by National Key Research and Development Program of China (No. 2017YFB0304100), National Natural Science Foundation of China (No. 61672343 and No. 61733011), Key Project of National Society Science Foundation of China (No. 15-ZDA041), The Art and Science Interdisciplinary Funds of Shanghai Jiao Tong University (No. 14JCRZ04).

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 903–908
New Orleans, Louisiana, June 5–6, 2018. ©2018 Association for Computational Linguistics

tures (Qin et al., 2016b; Pang et al., 2016; Qin et al., 2016a; Wang et al., 2016c; Zhao et al., 2017a; Wang et al., 2017; Qin et al., 2017; Cai and Zhao, 2017; Zhao et al., 2017b; Li et al., 2018). In this work, we introduce a neural network architecture for the concerned task and empirically study various neural networks to model the distributed representations for words and phrases.

In our system, we leverage an unambiguous vector representation via term embedding, and we take advantage of deep neural networks to discover the hypernym relationships between terms.

The rest of the paper is organized as follows: Section 2 briefly describes our system, Section 3 shows our experiments on the hyperym discovery task including the general-purpose and domain-specific one. Section 4 concludes this paper.

2 System Overview

Our hypernym discovery system can be roughly split into two parts, *Term Embedding* and *Hypernym Relationship Learning*. We first train term embeddings, either using word embedding or sense embedding to represent each word. Then, neural networks are used to discover and rank the hypernym candidates for given terms.

2.1 Embedding

To use deep neural networks, symbolic data needs to be transformed into distributed representations(Wang et al., 2016a; Qin et al., 2016b; Cai and Zhao, 2016; Zhang et al., 2016; Wang et al., 2016b, 2015; Cai et al., 2017). We use *Glove* toolkit to train the word embeddings using *UMBC* corpus (Han et al., 2013). Moreover, in order to perform word sense induction and disambiguation, the word embedding could be transformed to sense embedding, which is induced from exhisting word embeddings via clustering of ego-networks (Pelevina et al., 2016) of related words. Thus, each input word or phrase is embedded into vector sequence, $w = \{x_1, x_2, \dots, x_l\}$ where l denotes the sequence length. If the input term is a word, then $l = 1$ while for phrases, l means the number of words.

2.2 Hypernym Learning

Previous work like TAXOEMBED (Espinosa-Anke et al., 2016a) uses transformation matrix for hypernm relationship learning, which might be not optimal due to the lack of deeper nonlinear feature extraction. Thus, we empirically survey various neural networks to represent terms in latent space. After obtaining the representation for input term and all the candidate hypernyms, to give the ranked hypernym list, the cosine similarity between the term and the candidate hypernym is computed by,

$$cosine = \frac{\sum_{i=1}^{n}(x_i \times y_i)}{\sum_{i=1}^{n} x_i^2 \times \sum_{i=1}^{n} y_i^2}$$

where x_i and y_i denote the two concerned vectors. Our candidate neural networks include Convolutional Neural Network (CNN), Long-short Term Memory network (LSTM), Gated Recurrent Unit (GRU) and Recurrent Convolutional Neural Network (RCNN).

GRU The structure of GRU (Cho et al., 2014) used in this paper are described as follows.

$$\begin{aligned}
r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r), \\
z_t &= \sigma(W_z x_t + U_z h_{t-1} + b_z), \\
\tilde{h}_t &= \tanh(W_h x_t + U_h(rt \odot h_{t-1}) + b_h) \\
h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t
\end{aligned}$$

where $\odot$ denotes the element-wise multiplication. r_t and z_t are the reset and update gates respectively, and $\tilde{h}_t$ the hidden states.

LSTM LSTM (Hochreiter and Schmidhuber, 1997) unit is defined as follows.

$$\begin{aligned}
i_t &= \sigma(W_i x_t + W_h h_{t-1} + b_i), \\
f_t &= \sigma(W_f x_t + W_f h_{t-1} + b_f), \\
u_t &= \sigma(W_u x_t + W_u h_{t-1} + b_u), \\
c_t &= f_t \odot c_{t-1} + i_t \odot \tanh(W_c x_t + W_c h_{t-1} + b_c), \\
h_t &= \tanh(c_t) \odot u_t,
\end{aligned}$$

where σ stands for the sigmoid function, $\odot$ represents element-wise multiplication and $W_i, W_f, W_u, W_c, b_i, b_f, b_u, b_c$ are model parameters. i_t, f_t, u_t, c_t, h_t are the input gates, forget gates, memory cells, output gates and the current state, respectively.

CNN Convolutional neural networks have also been successfully applied to various NLP tasks, in which the temporal convolution operation and associated filters map local chunks (windows) of the input into a feature representation.

Concretely, let n denote the filter width, filter matrices $[W_1, W_2, \dots, W_k]$ with several variable sizes $[l_1, l_2, \dots, l_k]$ are utilized to perform the convolution operations for input embeddings. For the sake of simplicity, we will explain the procedure for only one embedding sequence. The embedding will be transformed to sequences $c_j (j \in [1, k])$:

$$c_j = [\dots; \tanh(W_j \cdot x_{[i:i+l_j-1]} + b_j); \dots]$$

where $[i : i + l_j - 1]$ indexes the convolution window. Additionally, we apply wide convolution operation between embedding layer and filter matrices, because it ensures that all weights in the filters reach the entire sentence, including the words at the margins.

A *one-max-pooling* operation is adopted after convolution and the output vector s is obtained through concatenating all the mappings for those k filters.

$$s_j = max(c_j)$$
$$s = [s_1 \oplus \dots \oplus s_j \oplus \dots \oplus s_k]$$

In this way, the model can capture the critical features in the sentence with different filters.

RCNN Since some input terms are phrases, whose member words share different weights. In RCNN, an adaptive gated decay mechanism is used to weight the words in the convolution layer. Following (Lei et al., 2016), we introduce neural gates similar λ to LSTMs to specify when and how to average the observed signals. The resulting architecture integrates recurrent networks with non-consecutive convolutions:

$$\lambda = \sigma(W^{\lambda} x_t + U^{\lambda} h_{t-1} + b^{\lambda})$$
$$c_t^1 = \lambda_t \odot c_{t-1}^1 + (1 - \lambda_t) \odot W_1 x_t$$
$$c_t^2 = \lambda_t \odot c_{t-1}^2 + (1 - \lambda_t) \odot (c_{t-1}^1 + W_2 x_t)$$
$$\dots$$
$$c_t^n = \lambda_t \odot c_{t-1}^n + (1 - \lambda_t) \odot (c_{n-1}^1 + W_n x_t)$$
$$h_t = \tanh(c_t^n + b)$$

where $c_t^1, c_t^2, \cdots, c_t^n$ are accumulator vectors that store weighted averages of 1-gram to n-gram features.

For discriminative training, we use a max-margin framework for learning (or fine-tuning) parameters θ. Specifically, a scoring function $\varphi(\cdot, \cdot; \theta)$ is defined to measure the semantic similarity between the corresponding representations of input term and hypernym. Let $p = \{p_1, ...p_n\}$ denote the hypernym corpus and $h \in p$ is the ground-truth hypernym to the term t_i, the optimal parameters θ are learned by minimizing the max-margin loss:

$$\max\{\varphi(t_i, p_i; \theta) - \varphi(t_i, a; \theta) + \delta(p_i, a)\}$$

where $\delta(.,.)$ denotes a non-negative margin and $\delta(p_i, a)$ is a small constant when $a \neq p_i$ and 0 otherwise.

3 Experiment

In the following experiments, besides the neural networks, we also simply average the embeddings of an input phrase as our baseline to discover the relationship of terms and their corresponding hypernyms for comparison (denoted as *term embedding averaging, TEA*).

3.1 Setting

Our hypernym discovery experiments include general-purpose substask for English and domain-specific ones for medical and music. Our evaluation is based on the following information retrieval metrics: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Precision at 1 (P@1), Precision at 3 (P@3), Precision at 5 (P@5), Precision at 15 (P@15).

For the sake of computational efficiency, we simply average the sense embedding if a word has more than one sense embedding (among various domains). Our model was implemented using the Theano[1] . The diagonal variant of AdaGrad (Duchi et al., 2011) is used for neural network training. We tune the hyper-parameters with the following range of values: learning rate $\in \{1e-3, 1e-2\}$, dropout probability $\in \{0.1, 0.2\}$, CNN filter width $\in \{2, 3, 4\}$. The hidden dimension of all neural models are 200. The batch size is set to 20 and the word embedding and sense embedding sizes are set to 300. All of our models are trained on a single GPU (NVIDIA GTX 980Ti), with roughly 1.5h for general-purpose subtask for English and 0.5h domain-specific domain-specific ones for medical and music. We run all our models up to 50 epoch and select the best result in validation.

3.2 Result and analysis

Table 2 shows the result on general-domain subtask for English. All the neural models outperform term embedding averaging in terms of

[1]https://github.com/Theano/Theano

Embedding	Model	MAP	MRR	P@1	P@3	P@5	P 15
Word	TEA	6.10	11.13	4.00	6.00	5.40	5.14
	GRU	8.13	16.22	8.00	**8.00**	6.67	6.94
	LSTM	3.95	7.52	4.00	4.33	3.97	3.97
	CNN	7.32	13.33	**8.00**	9.00	7.80	6.94
	RCNN	**8.74**	**12.83**	6.00	**9.67**	**8.87**	**9.15**
Sense	TEA	4.42	8.71	0.00	4.04	4.19	5.31
	GRU	5.42	9.44	0.00	4.44	4.89	5.83
	LSTM	5.62	9.97	4.00	4.35	5.01	6.83
	CNN	6.41	10.92	2.00	5.01	5.67	6.29
	RCNN	5.79	9.24	0.00	4.71	5.29	6.43

Table 2: Gold standard evaluation on general-purpose subtask.

Embed	Model	medical						music					
		MAP	MRR	P@1	P@3	P@5	P 15	MAP	MRR	P@1	P@3	P@5	P 15
Word	TEA	8.91	16.77	0.00	8.79	9.41	9.39	7.11	14.32	0.00	10.01	10.77	9.21
	GRU	13.27	21.89	0.00	13.33	14.89	14.06	15.20	20.33	0.00	17.78	18.67	15.45
	LSTM	11.49	21.11	0.00	**17.78**	12.22	11.83	14.08	20.77	0.07	13.33	16.00	15.00
	CNN	**18.31**	**24.52**	0.00	15.56	**20.44**	**20.00**	**17.58**	**27.15**	0.00	**20.00**	**20.00**	**16.04**
	RCNN	16.78	23.40	0.00	13.33	13.00	14.50	13.60	21.67	0.07	13.33	14.67	13.08
Sense	TEA	2.01	4.77	0.00	2.91	2.77	3.21	2.59	5.28	0.00	2.12	3.01	2.93
	GRU	4.88	9.11	0.00	6.67	6.42	6.91	5.32	10.74	**2.00**	4.44	5.33	4.95
	LSTM	5.10	10.22	0.00	6.67	6.12	6.94	4.39	10.21	0.00	8.89	5.33	3.61
	CNN	4.15	7.84	0.00	4.44	6.09	6.42	4.75	9.61	0.00	6.67	6.67	4.43
	RCNN	4.63	9.84	0.00	6.67	6.89	6.43	4.73	8.56	0.00	4.44	6.22	4.94

Table 3: Gold standard evaluation on domain-specific subtask. "Embed" is short for "Embedding".

all the metrics. This result indicates simply averaging the embedding of words in a phrase is not an appropriate solution to represent a phrase. Convolution or recurrent gated mechanisms in either CNN-based (CNN, RCNN) or RNN (GRU, LSTM) based neural networks could essentially be helpful of modeling the semantic connections between words in a phrase, and guide the networks to discover the hypernym relationships. We also observe CNN-based network performance is better than RNN-based, which indicates local features between words could be more important than long-term dependency in this task where the term length is up to trigrams.

To investigate the performance of neural models on specific domains, we conduct experiments on medical and medicine subtask. Table 3 shows the result. All the neural models outperform *term embedding averaging* in terms of all the metrics and CNN-based network also performs better than RNN-based ones in most of the metrics using word embedding, which verifies our hypothesis in the general-purpose task. Compared with word embedding, the sense embedding shows a much poorer result though they work closely in general-purpose subtask. The reason might be the simple averaging of sense embedding of various domains for a word, which may introduce too much noise and bias the overall sense representation. This also discloses that modeling the sense embedding of specific domains could be quite important for further improvement.

4 Conclusion

In this paper, we introduce a neural network architecture for the hypernym discovery task and empirically study various neural network models to model the representations in latent space for words and phrases. Experiments on three subtasks show the neural models can yield satisfying results. We also evaluate the performance of word embedding and sense embedding, showing that in domain-specific tasks, sense embedding could be much more volatile.

References

Guido Boella and Luigi Di Caro. 2013. Supervised learning of syntactic contexts for uncovering definitions and extracting hypernym relations in text databases. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 64–79.

Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for Chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 409–420.

Deng Cai and Hai Zhao. 2017. *Pair-Aware Neural Sentence Modeling for Implicit Discourse Relation Classification*. IEA/AIE 2017, Part II, LNAI 10351.

Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for Chinese. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 608–615.

Jose Camacho-Collados, Claudio Delli Bovi, Luis Espinosa-Anke, Sergio Oramas, Tommaso Pasini, Enrico Santus, Vered Shwartz, Roberto Navigli, and Horacio Saggion. 2018. SemEval-2018 Task 9: Hypernym Discovery. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, United States. Association for Computational Linguistics.

Kyunghyun Cho, Bart Van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*, pages 1724–1734.

Ido Dagan, Roth Dan, Fabio Zanzotto, and Mark Sammons. 2013. Recognizing textual entailment:models and applications. *Computational Linguistics*, 41(1):157–160.

John C. Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(39):2121–2159.

Luis Espinosa-Anke, Jose Camacho-Collados, Claudio Delli Bovi, and Horacio Saggion. 2016a. Supervised distributional hypernym discovery via domain adaptation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, page 424435.

Luis Espinosa-Anke, Horacio Saggion, Francesco Ronzano, and Roberto Navigli. 2016b. Extasem! extending, taxonomizing and semantifying domain terminologies. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 2594–2600.

Trevor Fountain and Mirella Lapata. 2012. Taxonomy induction using hierarchical random graphs. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 466–476.

Lushan Han, Abhay Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. Umbc ebiquity-core: Semantic textual similarity systems. *Second Joint Conference on Lexical and Computational Semantics (*SEM)*, 1:4452.

Sanda Harabagiu and Andrew Hickl. 2006. Methods for using textual entailment in open-domain question answering. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL (ACL 2006)*, pages 905–912.

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

Yong Bin Kang, Pari Delir Haghighi, and Frada Burstein. 2016. Taxofinder: A graph-based approach for taxonomy learning. *IEEE Transactions on Knowledge & Data Engineering*, 28(2):524–536.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi Jaakkola, Katerina Tymoshenko, Alessandro Moschitti, and Lluis Marquez. 2016. Semi-supervised question retrieval with gated convolutions. In *Proceedings of NAACL-HLT 2016*, pages 1279–1289.

Haonan Li, Zhisong Zhang, Yuqi Ju, and Hai Zhao. 2018. Neural character-level dependency parsing for Chinese. In *The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*.

Chenxi Pang, Hai Zhao, and Zhongyi Li. 2016. I can guess what you mean: A monolingual query enhancement for machine translation. In *The Fifteenth China National Conference on Computational Linguistics (CCL 2016)*.

Maria Pelevina, Nikolay Arefiev, Chris Biemann, and Alexander Panchenko. 2016. Making sense of word embeddings. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 174–183.

Lianhui Qin, Zhisong Zhang, and Hai Zhao. 2016a. Shallow discourse parsing using convolutional neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 70–77.

Lianhui Qin, Zhisong Zhang, and Hai Zhao. 2016b. A stacking gated neural architecture for implicit discourse relation classification. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, pages 2263–2270.

Lianhui Qin, Zhisong Zhang, Hai Zhao, Zhiting Hu, and Eric P. Xing. 2017. Adversarial connective-exploiting networks for implicit discourse relation classification. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 1006–1017.

Stephen Roller and Katrin Erk. 2016. Relations such as hypernymy: Identifying and exploiting hearst patterns in distributional vectors for lexical entailment. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, page 21632172.

Paola Velardi, Stefano Faralli, and Roberto Navigli. 2013. Ontolearn reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics*, 39(3):665–707.

Hao Wang, Hai Zhao, and Zhisong Zhang. 2017. A transition-based system for universal dependency parsing. In *CONLL 2017 Shared Task: Multilingual Parsing From Raw Text To Universal Dependencies (CONLL 2017)*, pages 191–197.

Peilu Wang, Yao Qian, Frank K. Soong, Lei He, and Hai Zhao. 2016a. Learning distributed word representations for bidirectional lstm recurrent neural network. In *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL 2016)*, pages 527–533.

Rui Wang, Masao Utiyama, Isao Goto, Eiichiro Sumita, Hai Zhao, and Bao Liang Lu. 2016b. Converting continuous-space language models into n-gram language models with efficient bilingual pruning for statistical machine translation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 15(3):11.

Rui Wang, Hai Zhao, Bao Liang Lu, Masao Utiyama, and Eiichiro Sumita. 2015. Bilingual continuous-space language model growing for statistical machine translation. *IEEE/ACM Transactions on Audio Speech & Language Processing*, 23(7):1209–1220.

Rui Wang, Hai Zhao, Bao Liang Lu, Masao Utiyama, and Eiichro Sumita. 2016c. Connecting phrase based statistical machine translation adaptation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers (COLING 2016)*, page 31353145.

Mohamed Yahya, Klaus Berberich, Shady Elbassuoni, and Gerhard Weikum. 2013. Robust question answering over the web of linked data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management (CIKM 2013)*, pages 1107–1116.

Zhisong Zhang, Hai Zhao, and Lianhui Qin. 2016. Probabilistic graph-based dependency parsing with convolutional neural network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1382–1392.

Hai Zhao, Deng Cai, Changning Huang, and Chunyu Kit. 2017a. *Chinese Word Segmentation, a decade review (2007-2017)*. China Social Sciences Press, Beijing, China.

Hai Zhao, Deng Cai, Yang Xin, Yuzhu Wang, and Zhongye Jia. 2017b. A hybrid model for Chinese spelling check. *ACM Transactions on Asian Low-Resource Language Information Process*, pages 1–22.