Learning Entity Representation for Entity Disambiguation

Zhengyan He[†] **Shujie Liu**[‡] **Mu Li**[‡] **Ming Zhou**[‡] **Longkai Zhang**[†] **Houfeng Wang**[†*]
[†] Key Laboratory of Computational Linguistics (Peking University) Ministry of Education,China
[‡] Microsoft Research Asia
hezhengyan.hit@gmail.com {shujliu,muli,mingzhou}@microsoft.com
zhlongk@qq.com wanghf@pku.edu.cn

Abstract

We propose a novel entity disambiguation model, based on Deep Neural Network (DNN). Instead of utilizing simple similarity measures and their disjoint combinations, our method directly optimizes document and entity representations for a given similarity measure. Stacked Denoising Auto-encoders are first employed to learn an initial document representation in an unsupervised pre-training stage. A supervised fine-tuning stage follows to optimize the representation towards the similarity measure. Experiment results show that our method achieves state-of-the-art performance on two public datasets without any manually designed features, even beating complex collective approaches.

1 Introduction

Entity linking or disambiguation has recently received much attention in natural language processing community (Bunescu and Pasca, 2006; Han et al., 2011; Kataria et al., 2011; Sen, 2012). It is an essential first step for succeeding sub-tasks in knowledge base construction (Ji and Grishman, 2011) like populating attribute to entities. Given a sentence with four mentions, "The [[Python]] of [[Delphi]] was a creature with the body of a snake. This creature dwelled on [[Mount Parnassus]], in central [[Greece]]." How can we determine that Python is an earth-dragon in Greece mythology and not the popular programming language, Delphi is not the auto parts supplier, and Mount Parnassus is in Greece, not in Colorado?

A most straightforward method is to compare the context of the mention and the definition of candidate entities. Previous work has explored many ways of measuring the relatedness of context d and entity e, such as dot product, cosine similarity, Kullback-Leibler divergence, Jaccard distance, or more complicated ones (Zheng et al., 2010; Kulkarni et al., 2009; Hoffart et al., 2011; Bunescu and Pasca, 2006; Cucerzan, 2007; Zhang et al., 2011). However, these measures are often duplicate or over-specified, because they are disjointly combined and their atomic nature determines that they have no internal structure.

Another line of work focuses on collective disambiguation (Kulkarni et al., 2009; Han et al., 2011; Ratinov et al., 2011; Hoffart et al., 2011). Ambiguous mentions within the same context are resolved simultaneously based on the coherence among decisions. Collective approaches often undergo a non-trivial decision process. In fact, (Ratinov et al., 2011) show that even though global approaches can be improved, local methods based on only similarity $sim(d, e)$ of context d and entity e are hard to beat. This somehow reveals the importance of a good modeling of $sim(d, e)$.

Rather than learning context entity association at word level, topic model based approaches (Kataria et al., 2011; Sen, 2012) can learn it in the semantic space. However, the one-topic-per-entity assumption makes it impossible to scale to large knowledge base, as every entity has a separate word distribution $P(w|e)$; besides, the training objective does not directly correspond with disambiguation performances.

To overcome disadvantages of previous approaches, we propose a novel method to learn context entity association enriched with deep architecture. Deep neural networks (Hinton et al., 2006; Bengio et al., 2007) are built in a hierarchical manner, and allow us to compare context and entity at some higher level abstraction; while at lower levels, general concepts are shared across entities, resulting in compact models. Moreover, to make our model highly correlated with disambiguation performance, our method directly optimizes doc-

*Corresponding author

Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 30–34,
Sofia, Bulgaria, August 4-9 2013.

ument and entity representations for a fixed similarity measure. In fact, the underlying representations for computing similarity measure add internal structure to the given similarity measure. Features are learned leveraging large scale annotation of Wikipedia, without any manual design efforts. Furthermore, the learned model is compact compared with topic model based approaches, and can be trained discriminatively without relying on expensive sampling strategy. Despite its simplicity, it beats all complex collective approaches in our experiments. The learned similarity measure can be readily incorporated into any existing collective approaches, which further boosts performance.

2 Learning Representation for Contextual Document

Given a mention string m with its context document d, a list of candidate entities $C(m)$ are generated for m, for each candidate entity $e_i \in C(m)$, we compute a ranking score $sim(d_m, e_i)$ indicating how likely m refers to e_i. The linking result is $e = \arg\max_{e_i} sim(d_m, e_i)$.

Our algorithm consists of two stages. In the pre-training stage, Stacked Denoising Auto-encoders are built in an unsupervised layer-wise fashion to discover general concepts encoding d and e. In the supervised fine-tuning stage, the entire network weights are fine-tuned to optimize the similarity score $sim(d, e)$.

2.1 Greedy Layer-wise Pre-training

Stacked Auto-encoders (Bengio et al., 2007) is one of the building blocks of deep learning. Assume the input is a vector x, an auto-encoder consists of an encoding process $h(x)$ and a decoding process $g(h(x))$. The goal is to minimize the reconstruction error $\mathcal{L}(x, g(h(x)))$, thus retaining maximum information. By repeatedly stacking new auto-encoder on top of previously learned $h(x)$, stacked auto-encoders are obtained. This way we learn multiple levels of representation of input x.

One problem of auto-encoder is that it treats all words equally, no matter it is a function word or a content word. Denoising Auto-encoder (DA) (Vincent et al., 2008) seeks to reconstruct x given a random corruption $\tilde{x}$ of x. DA can capture global structure while ignoring noise as the author shows in image processing. In our case, we input each document as a binary bag-of-words vector (Fig. 1). DA will capture general concepts and ignore noise like function words. By applying masking noise (randomly mask 1 with 0), the model also exhibits a fill-in-the-blank property (Vincent et al., 2010): the missing components must be recovered from partial input. Take "greece" for example, the model must learn to predict it with "python" "mount", through some hidden unit. The hidden unit may somehow express the concept of Greece mythology.

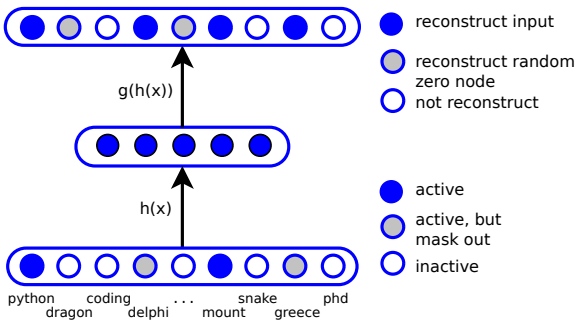

Figure 1: DA and reconstruction sampling.

In order to distinguish between a large number of entities, the vocabulary size must be large enough. This adds considerable computational overhead because the reconstruction process involves expensive dense matrix multiplication. Reconstruction sampling keeps the sparse property of matrix multiplication by reconstructing a small subset of original input, with no loss of quality of the learned representation (Dauphin et al., 2011).

2.2 Supervised Fine-tuning

This stage we optimize the learned representation ("hidden layer n" in Fig. 2) towards the ranking score $sim(d, e)$, with large scale Wikipedia annotation as supervision. We collect hyperlinks in Wikipedia as our training set $\{(d_i, e_i, m_i)\}$, where m_i is the mention string for candidate generation. The network weights below "hidden layer n" are initialized with the pre-training stage.

Next, we stack another layer on top of the learned representation. The whole network is tuned by the final supervised objective. The reason to stack another layer on top of the learned representation, is to capture problem specific structures. Denote the encoding of d and e as $\hat{d}$ and $\hat{e}$ respectively, after stacking the problem-specific layer, the representation for d is given as $f(d) = \text{sigmoid}(W \times \hat{d} + b)$, where W and b are weight and bias term respectively. $f(e)$ follows the same

encoding process.

The similarity score of (d, e) pair is defined as the dot product of $f(d)$ and $f(e)$ (Fig. 2):

$$sim(d, e) = \text{Dot}(f(d), f(e)) \quad (1)$$

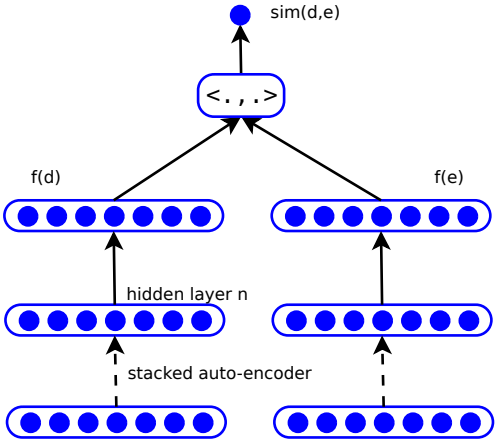

Figure 2: Network structure of fine-tuning stage.

Our goal is to rank the correct entity higher than the rest candidates relative to the context of the mention. For each training instance (d, e), we contrast it with one of its negative candidate pair (d, e'). This gives the pairwise ranking criterion:

$$\mathcal{L}(d, e) = \max\{0, 1 - sim(d, e) + sim(d, e')\} \quad (2)$$

Alternatively, we can contrast with all its candidate pairs (d, e_i). That is, we raise the similarity score of true pair $sim(d, e)$ and penalize all the rest $sim(d, e_i)$. The loss function is defined as negative log of $softmax$ function:

$$\mathcal{L}(d, e) = -\log \frac{\exp sim(d, e)}{\sum_{e_i \in C(m)} \exp sim(d, e_i)} \quad (3)$$

Finally, we seek to minimize the following training objective across all training instances:

$$\mathcal{L} = \sum_{d,e} \mathcal{L}(d, e) \quad (4)$$

The loss function is closely related to contrastive estimation (Smith and Eisner, 2005), which defines where the positive example takes probability mass from. We find that by penalizing more negative examples, convergence speed can be greatly accelerated. In our experiments, the $softmax$ loss function consistently outperforms pairwise ranking loss function, which is taken as our default setting.

However, the $softmax$ training criterion adds additional computational overhead when performing mini-batch Stochastic Gradient Descent (SGD). Although we can use a plain SGD (i.e. mini-batch size is 1), mini-batch SGD is faster to converge and more stable. Assume the mini-batch size is m and the number of candidates is n, a total of $m \times n$ forward-backward passes over the network are performed to compute a similarity matrix (Fig. 3), while pairwise ranking criterion only needs $2 \times m$. We address this problem by grouping training pairs with same mention m into one mini-batch $\{(d, e_i)|e_i \in C(m)\}$. Observe that if candidate entities overlap, they share the same forward-backward path. Only $m + n$ forward-backward passes are needed for each mini-batch now.

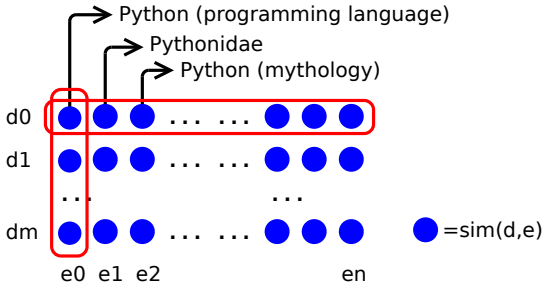

Figure 3: Sharing path within mini-batch.

The re-organization of mini-batch is similar in spirit to Backpropagation Through Structure (BTS) (Goller and Kuchler, 1996). BTS is a variant of the general backpropagation algorithm for structured neural network. In BTS, parent node is computed with its child nodes at the forward pass stage; child node receives gradient as the sum of derivatives from all its parents. Here (Fig. 2), parent node is the score node $sim(d, e)$ and child nodes are $f(d)$ and $f(e)$. In Figure 3, each row shares forward path of $f(d)$ while each column shares forward path of $f(e)$. At backpropagation stage, gradient is summed over each row of score nodes for $f(d)$ and over each column for $f(e)$.

Till now, our input simply consists of bag-of-words binary vector. We can incorporate any handcrafted feature $f(d, e)$ as:

$$sim(d, e) = \text{Dot}(f(d), f(e)) + \vec{\lambda}\vec{f}(d, e) \quad (5)$$

In fact, we find that with only $\text{Dot}(f(d), f(e))$ as ranking score, the performance is sufficiently good. So we leave this as our future work.

3 Experiments and Analysis

Training settings: In pre-training stage, input layer has 100,000 units, all hidden layers have 1,000 units with rectifier function $max(0, x)$. Following (Glorot et al., 2011), for the first reconstruction layer, we use sigmoid activation function and cross-entropy error function. For higher reconstruction layers, we use $softplus$ ($\log(1 + \exp(x))$) as activation function and squared loss as error function. For corruption process, we use a masking noise probability in {0.1,0.4,0.7} for the first layer, a Gaussian noise with standard deviation of 0.1 for higher layers. For reconstruction sampling, we set the reconstruction rate to 0.01. In fine-tuning stage, the final layer has 200 units with sigmoid activation function. The learning rate is set to 1e-3. The mini-batch size is set to 20.

We run all our experiments on a Linux machine with 72GB memory 6 core Xeon CPU. The model is implemented in Python with C extensions, numpy configured with Openblas library. Thanks to reconstruction sampling and refined mini-batch arrangement, it takes about 1 day to converge for pre-training and 3 days for fine-tuning, which is fast given our training set size.

Datasets: We use half of Wikipedia [1] plain text (~1.5M articles split into sections) for pre-training. We collect a total of 40M hyperlinks grouped by name string m for fine-tuning stage. We holdout a subset of hyperlinks for model selection, and we find that 3 layers network with a higher masking noise rate (0.7) always gives best performance.

We select TAC-KBP 2010 (Ji and Grishman, 2011) dataset for non-collective approaches, and AIDA [2] dataset for collective approaches. For both datasets, we evaluate the non-NIL queries. The TAC-KBP and AIDA testb dataset contains 1020 and 4485 non-NIL queries respectively.

For candidate generation, mention-to-entity dictionary is built by mining Wikipedia structures, following (Cucerzan, 2007). We keep top 30 candidates by prominence $P(e|m)$ for speed consideration. The candidate generation recall are 94.0% and 98.5% for TAC and AIDA respectively.

Analysis: Table 1 shows evaluation results across several best performing systems. (Han et al., 2011) is a collective approach, using Personalized PageRank to propagate evidence between different decisions. To our surprise, our method with only local evidence even beats several complex collective methods with simple word similarity. This reveals the importance of context modeling in semantic space. Collective approaches can improve performance only when local evidence is not confident enough. When embedding our similarity measure $sim(d, e)$ into (Han et al., 2011), we achieve the best results on AIDA.

A close error analysis shows some typical errors due to the lack of prominence feature and name matching feature. Some queries accidentally link to rare candidates and some link to entities with completely different names. We will add these features as mentioned in Eq. 5 in future. We will also add NIL-detection module, which is required by more realistic application scenarios. A first thought is to construct pseudo-NIL with Wikipedia annotations and automatically learn the threshold and feature weight as in (Bunescu and Pasca, 2006; Kulkarni et al., 2009).

Methods	micro P@1	macro P@1
TAC 2010 eval		
Lcc (2010) (top1, noweb)	79.22	-
Siel 2010 (top2, noweb)	71.57	-
our best	80.97	-
AIDA dataset (collective approaches)		
AIDA (2011)	82.29	82.02
Shirakawa et al. (2011)	81.40	83.57
Kulkarni et al. (2009)	72.87	76.74
wordsim (cosine)	48.38	37.30
Han (2011) +wordsim	78.97	75.77
our best (non-collective)	84.82	83.37
Han (2011) + our best	85.62	83.95

Table 1: Evaluation on TAC and AIDA dataset.

4 Conclusion

We propose a deep learning approach that automatically learns context-entity similarity measure for entity disambiguation. The intermediate representations are learned leveraging large scale annotations of Wikipedia, without any manual effort of designing features. The learned representation of entity is compact and can scale to very large knowledge base. Furthermore, experiment reveals the importance of context modeling in this field. By incorporating our learned measure into collective approach, performance is further improved.

[1] available at http://dumps.wikimedia.org/enwiki/, we use the 20110405 xml dump.

[2] available at http://www.mpi-inf.mpg.de/yago-naga/aida/

Acknowledgments

We thank Nan Yang, Jie Liu and Fei Wang for helpful discussions. This research was partly supported by National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), National Natural Science Foundation of China (No.91024009) and Major National Social Science Fund of China(No. 12&ZD227).

References

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. 2007. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153.

R. Bunescu and M. Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, volume 6, pages 9–16.

S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 6, pages 708–716.

Y. Dauphin, X. Glorot, and Y. Bengio. 2011. Large-scale learning of embeddings with reconstruction sampling. In *Proceedings of the Twenty-eighth International Conference on Machine Learning (ICML11)*.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning*.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE.

X. Han, L. Sun, and J. Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

G.E. Hinton, S. Osindero, and Y.W. Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

J. Hoffart, M.A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Heng Ji and Ralph Grishman. 2011. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1148–1158, Portland, Oregon, USA, June. Association for Computational Linguistics.

S.S. Kataria, K.S. Kumar, R. Rastogi, P. Sen, and S.H. Sengamedu. 2011. Entity disambiguation with hierarchical topic models. In *Proceedings of KDD*.

S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

J. Lehmann, S. Monahan, L. Nezda, A. Jung, and Y. Shi. 2010. Lcc approaches to knowledge base population at tac 2010. In *Proc. TAC 2010 Workshop*.

L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL)*.

P. Sen. 2012. Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st international conference on World Wide Web*, pages 729–738. ACM.

M. Shirakawa, H. Wang, Y. Song, Z. Wang, K. Nakayama, T. Hara, and S. Nishio. 2011. Entity disambiguation based on a probabilistic taxonomy. Technical report, Technical Report MSR-TR-2011-125, Microsoft Research.

N.A. Smith and J. Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

P. Vincent, H. Larochelle, Y. Bengio, and P.A. Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408.

W. Zhang, Y.C. Sim, J. Su, and C.L. Tan. 2011. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1909–1914. AAAI Press.

Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 483–491, Los Angeles, California, June. Association for Computational Linguistics.