A Hierarchical Knowledge Representation for Expert Finding on Social Media

Yanran Li[1], **Wenjie Li**[1], and **Sujian Li**[2]
[1]Computing Department, Hong Kong Polytechnic University, Hong Kong
[2]Key Laboratory of Computational Linguistics, Peking University, MOE, China
{csyli, cswjli}@comp.polyu.edu.hk
lisujian@pku.edu.cn

Abstract

Expert finding on social media benefits both individuals and commercial services. In this paper, we exploit a 5-level tree representation to model the posts on social media and cast the expert finding problem to the matching problem between the learned user tree and domain tree. We enhance the traditional approximate tree matching algorithm and incorporate word embeddings to improve the matching result. The experiments conducted on Sina Microblog demonstrate the effectiveness of our work.

1 Introduction

Expert finding has been arousing great interests among social media researchers after its successful applications on traditional media like academic publications. As already observed, social media users tend to follow others for professional interests and knowledge (Ramage et al, 2010). This builds the basis for mining expertise and finding experts on social media, which facilitates the services of user recommendation and question-answering, etc.

Despite the demand to access expertise, the challenges of identifying domain experts on social media exist. Social media often contains plenty of noises such as the tags with which users describe themselves. Noises impose the inherent drawback on the feature-based learning methods (Krishnamurthy et al, 2008). Data imbalance and sparseness also limits the performance of the promising latent semantic analysis methods such as the LDA-like topic models (Blei et al, 2003; Ramage et al, 2009). When some topics co-occur more frequently than others, the strict assumption of these topic models cannot be met and consequently many nonsensical topics will be generated (Zhao and Jiang, 2011; Pal et al, 2011; Quercia et al, 2012). Furthermore, not as simple as celebrities, the definition of experts introduces additional difficulties. Experts cannot be simply judged by the number of followers. The knowledge conveyed in what they say is essential. This leads to the failures of the network-based methods (Java et al, 2007; Weng et al, 2010; Pal et al, 2011).

The challenges mentioned above inherently come from insufficient representations. They motivate us to propose a more flexible domain expert finding framework to explore effective representations that are able to tackle the complexity lies in the social media data. The basic idea is as follows. Experts talk about the professional knowledge in their posts and these posts are supposed to contain more domain knowledge than the posts from the other ordinary users. We determine whether or not users are experts on specific domains by matching their professional knowledge and domain knowledge. The key is how to capture such information for both users and domains with the appropriate representation, which is, in our view, the reason why most of previous work fails.

To go beyond the feature-based classification methods and the vector representation inference in expert finding, a potential solution is to incorporate the semantic information for knowledge modeling. We achieve this goal by representing user posts using a hierarchical tree structure to capture correlations among words and topics. To tackle the data sparseness problem, we apply word embeddings to tree-nodes to further enhance semantic representation and to support semantic matching. Expert finding is then cast to the problem of determining the edit distance between the user tree and the domain tree, which is computed with an approximate tree matching algorithm.

The main contribution of this work is to integrate the hierarchical tree representation and structure matching together to profile users' and domains' knowledge. Using such trees allows us to flexibly incorporate more information into the data representation, such as the relations between latent topics and the semantic similarities between words. The experiments conducted on Sina Microblog demonstrate the effectiveness of the proposed framework and the corresponding methods.

2 Knowledge Representation with Hierarchical Tree

To capture correlations between topics, Pachinko Allocation Model (PAM) (Li and McCallum, 2006) uses a directed acyclic graph (DAG) with leaves representing individual words in the vocabulary and each interior node representing a correlation among its children. In particular, multi-level PAM is capable of revealing interconnection between sub-level nodes by inferencing corresponding super-level nodes. It is a desired property that enables us to capture hierarchical relations among both inner-level and inter-level nodes and thereby enhance the representation of users' posts. More important, the inter-level hierarchy benefits to distribute words from super-level generic topics to sub-level specific topics.

In this work, we exploit a 5-level PAM to learn the hierarchical knowledge representation for each individual user and domain. As shown in Figure 1, the 5-level hierarchy consists of one root topic r, I topics at the second level $X = \{x_1, x_2, \ldots, x_I\}$, J topics at the third level $Y = \{y_1, y_2, \ldots, y_J\}$, K topics at the fourth level $Z = \{z_1, z_2, \ldots, z_K\}$ and words at the bottom. The whole hierarchy is fully connected.

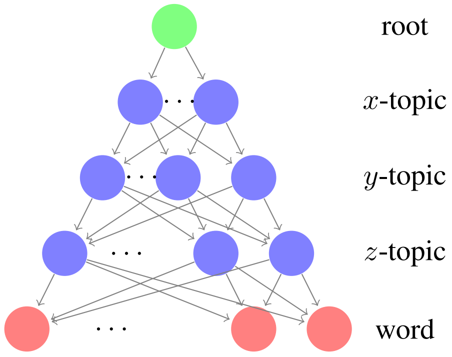

Figure 1: 5-level PAM

Each topic in 5-level PAM is associated with a distribution $g(\cdot)$ over its children. In general, $g(\cdot)$ can be any distribution over discrete variables. Here, we use a set of Dirichlet compound multinomial distributions associated with the root, the second-level and the third-level topics. These distributions are $\{g_r(\alpha)\}$, $\{g_i(\gamma_i)\}_{i=1}^{I}$ and $\{g_i(\delta_j)\}_{j=1}^{J}$. They are used to sample the multinomial distributions θ_x, θ_y and θ_z over the corresponding sub-level topics. As to the fourth-level topics, we use a fixed multinomial distribution $\{\phi_{z_k}\}_{k=1}^{K}$ sampled once for the whole data from a single Dirichlet distribution $g(\beta)$. Figure 2 illustrates the plate notation of this 5-level PAM.

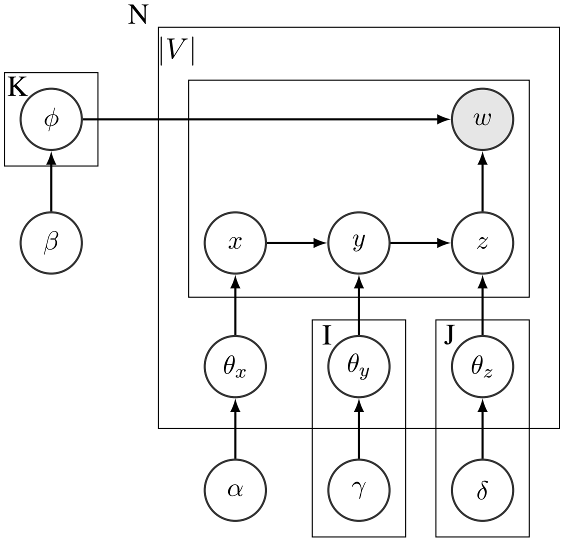

Figure 2: Plate Notation of 5-level PAM

By integrating out the sampled multinomial distributions θ_x, θ_y, θ_z, ϕ and summing over $\mathbf{x}, \mathbf{y}, \mathbf{z}$, we obtain the Gibbs sampling distribution for word $w = w_m$ in document d as:

$$\begin{aligned}
&P\left(x_w{=}x_i, y_w{=}y_j, z_w{=}z_k|\mathbf{D}, \mathbf{x}_{-w}, \mathbf{y}_{-w}, \mathbf{z}_{-w}, \alpha, \gamma, \delta, \beta\right)\\
&\propto P\left(w, x_w, y_w, z_w|\mathbf{D}_{-w}, \mathbf{x}_{-w}, \mathbf{y}_{-w}, \mathbf{z}_{-w}, \alpha, \gamma, \delta, \beta\right)\\
&= \frac{P(\mathbf{D}, \mathbf{x}, \mathbf{y}, \mathbf{z}|\alpha, \gamma, \delta, \beta)}{P(\mathbf{D}_{-w}, \mathbf{x}_{-w}, \mathbf{y}_{-w}, \mathbf{z}_{-w}|\alpha, \gamma, \delta, \beta)}\\
&= \frac{n_i^{(d)} + \alpha_i}{n_r^{(d)} + \sum_{i'=1}^{K} \alpha_{i'}} \times \frac{n_{ij}^{(d)} + \gamma_{ij}}{n_i^{(d)} + \sum_{j'=1}^{L} \gamma_{ij'}}\\
&\quad \times \frac{n_{jk}^{(d)} + \delta_{jk}}{n_j^{(d)} + \sum_{k'=1}^{J} \delta_{jk'}} \times \frac{n_{km}^{(d)} + \beta_m}{n_k + \sum_{m'=1}^{n} \beta_{m'}}
\end{aligned}$$

where $n_r^{(d)}$ is the number of occurrences of the root r in document d, which is equivalent to the number of tokens in the document. $n_i^{(d)}$, $n_{ij}^{(d)}$ and $n_{jk}^{(d)}$ are respectively the number of occurrences of x_i, y_j and z_k sampled from their upper-level topics. n_k is the number of occurrences of the fourth-level topics z_k in the whole dataset and n_{km} is the number of occurrences of word w_m in z_k. $-w$

indicates all observations or topic assignments except word w.

With the fixed Dirichlet parameter α for the root and β as the prior, what's left is to estimate (learn from data) γ and δ to capture the different correlations among topics. To avoid the use of iterative methods which are often computationally extensive, instead we approximate these two Dirichlet parameters using the moment matching algorithm, the same as (Minka, 2000; Casella and Berger, 2001; Shafiei and Milios, 2006). With smoothing techniques, in each iteration of Gibbs sampling we update:

$$\begin{aligned}
mean_{ij} =& \frac{1}{N_i + 1} \times \left(\sum_d \frac{n_{ij}^{(d)}}{n_i^{(d)}} + \frac{1}{L} \right) \\
var_{ij} =& \frac{1}{N_i + 1} \times \left(\sum_d (\frac{n_{ij}^{(d)}}{n_i^{(d)}} - mean_{ij})^2 \right. \\
& \left. + (\frac{1}{L} - mean_{ij})^2 \right) \\
m_{ij} =& \frac{mean_{ij} \times (1 - mean_{ij})}{var_{ij}} - 1 \\
\gamma_{ij} =& \frac{mean_{ij}}{\exp\left(\frac{\sum_j \log(m_{ij})}{L-1} \right)}
\end{aligned}$$

where N_i is the number of documents with non-zero counts of super-level topic x_i. Parameter estimation of δ is the same as γ.

3 Expert Finding with Approximate Tree Matching

Once the hierarchical representations of users and domains have been generated, we can determine whether or not a user is an expert on a domain based on their matching degree, which is a problem analogous to tree-to-tree correction using *edit distance* (Selkow, 1977; Shasha and Zhang, 1990; Wagner, 1975; Wagner and Fischer, 1974; Zhang and Shasha, 1989). Given two trees T_1 and T_2, a typical *edit distance*-based correction approach is to transform T_1 to T_2 with a sequence of *editing operations* $S =< s_1, s_2, \ldots, s_k >$ such that $s_k\left(s_{k-1}\left(\ldots\left(s_1\left(T_1\right)\right)\ldots\right)\right) = T_2$. Each operation is assigned a *cost* $\sigma(s_i)$ that represents the difficulty of making that operation. By summing up the *costs* of all necessary operations, the total cost $\sigma(S) = \sum_{i=1}^{k} \sigma(s_i)$ defines the matching degree of T_1 and T_2.

We assume that an expert could only master a part of professional domain knowledge rather than the whole and thereby revise a traditional approximate tree matching algorithm (Zhang and Shasha, 1989) to calculate the matching degree. This assumption especially makes sense when the domain we are concerned with is quite general. Let T_d and T_u denote the learned domain knowledge tree and the user knowledge tree, we match T_d to the remaining trees resulting from cutting all possible sets of disjoint sub-trees of T_u. We specifically penalize *no cost* if some sub-trees are missing in matching process. We define two types of operations. The *substitution* operations edit the dissimilar words on tree-nodes, while the *insertion* and *deletion* operations perform on tree-structures. Expert finding is then to calculate the minimum matching cost on T_d and T_u. If the cost is smaller than an empirically defined threshold λ_d, we identify user u as an expert on domain d.

To alleviate the sparseness problem caused by direct letter-to-letter matching in tree-node mapping, we embed word embeddings (Bengio et al, 2003) into the substitution operation. We apply the `word2vec` skip-gram model (Mikolov et al, 2013(a); Mikolov et al, 2013(b)) to encode each word in our vocabulary with a probability vector and directly use the similarity generated by `word2vec` as the tree-node similarity. The costs of insertion and deletion operations will be explained in Section 4. Actually all these three costs can be defined in accordance with applicant needs. In brief, by combining both hierarchical representation of tree-structure and word embeddings of tree-nodes, we achieve our goal to enhance semantics.

4 Experiments

The experiments are conducted on 5 domains (i.e., *Beauty Blogger*, *Beauty Doctor*, *Parenting*, *E-Commerce*, and *Data Science*) in Sina Microblog, a Twitter-like microblog in China. To learn PAM, we manually select 40 users in each domain from the official expert lists released by Sina Microblog[1], and crawl all of their posts. In average, there are 113,924 posts in each domain. Notice that the expert lists are not of high quality. We have to do manual verification to filter out noises. For evaluation, we select another 80 users in each domain from the expert list, with 40 verified as experts and the other 40 as non-experts.

Since there is no state-of-art Chinese word embeddings publicly available, we use another Sina

[1] `http://d.weibo.com/1087030002_558_3_2014#`

Table 1: Classification Results

Approach	Precision		Recall		F-Score	
	Macro	Micro	Macro	Micro	Macro	Micro
unigram	0.380	0.484	0.615	0.380	0.469	0.432
bigram	0.435	0.537	0.615	0.435	0.507	0.486
LDA	0.430	0.473	0.540	0.430	0.474	0.451
Twitter-LDA	0.675	0.763	0.680	0.430	0.675	0.451
PAM	**0.720**	**0.818**	**0.720**	**0.720**	**0.714**	**0.769**

Microblog dataset provided by `pennyliang`[2], which contains 25 million posts and nearly 100 million tokens in total, to learn the word embeddings of 50-dimension. We pre-process the data with the `Rwordseg` segmentation package[3] and discard nonsensical words with the `pullword` package[4].

When learning 5-level PAM, we set fixed parameters $\alpha = 0.25$, $\beta = 0.25$ and from top to down, $I = 10$, $J = 20$, $K = 20$ for the number of second, third and fourth levels of topics, respectively. And we initialize γ and δ with 0.25. For tree matching, we define the cost of tree-node *substitution* operation between word a and b as Eq (1). The costs of *insertion* and *deletion* operations for tree-structure matching are `MAX_VALUE`. Here we set `MAX_VALUE` as 100 experimentally. The threshold λ_d used to determine the expert is set to be 12 times of `MAX_VALUE`.

$$\sigma(a \rightarrow b) = \begin{cases} 0, & a = b \\ \text{sim}\,(a, b)\,, & \text{sim}(a, b) > 0.55 \\ \texttt{MAX\_VALUE}, & \textit{otherwise} \end{cases} \quad (1)$$

We compare PAM with n-gram (unigram and bigram), LDA (Blei et al, 2003) and Twitter-LDA (Zhao and Jiang, 2011). We set β in LDA and Twitter-LDA to 0.01, γ in Twiitter-LDA to 20. For α, we adopt the commonly used $50/T$ heuristics where the number of topics $T = 50$. To be fair, we all use the tokens after pullword preprocessing as the input to extract features for classification. Following Zhao and Jiang (2011), we train four ℓ_2-regularized logistic regression classifiers using the `LIBLINEAR` package (Fan et al, 2008) on the top 200 unigrams and bigrams ranked according to Chi-squared and 100-dimensional topic vectors induced by LDA and Twitter-LDA, respectively. We also compare our model with/without word embeddings to demonstrate the effectiveness of this semantic enhancement. The results are presented in Table 1.

In general, LDA, Twitter-LDA and PAM outperform unigram and bigram, showing the strength of latent semantic modeling. Within the first two models, Twitter-LDA yields better precisions than LDA because of its ability to overcome the difficulty of modeling short posts on social media. It designs an additional background word distribution to remove the noisy words and assumes that a single post can belong to several topics.

Our 5-level PAM gains observed improvement over Twitter-LDA. We attribute this to the advantages of tree representations over vector feature representations, the effective approximate tree matching algorithm and the complementary word embeddings. As mentioned in Section 1, LDA and other topic models like Twitter-LDA share the same assumption that each topic should be independent with each other. This assumption however is too strict for the real world data. Our tree-like 5-level PAM relaxes such assumption with two additional layers of super-topics modeled with Dirichlet compound multinomial distributions, which is the key to capture topic correlations. Furthermore, by allowing partial matching and incorporating word embeddings, we successfully overcome the sparseness problem.

While macro-averages give equal weight to each domain, micro-averages give equal weight to each user. The significant difference between the macro- and micro- scores in Table 1 is caused by the different nature of 5 domains. In fact, the posts of experts on the domain *E-Commerce* are to some extent noisy and contain lots of words irrelevant to the domain knowledge. Meanwhile, the posts of experts on the domain *Data Science* are less distinguishable. The higher micro-recalls of PAM demonstrate its generalization ability over

[2] `http://chuansong.me/account/pennyjob`

[3] `http://jliblog.com/app/rwordseg`

[4] `http://pullword.com/`

LDA and Twitter-LDA.

5 Conclusion

In this paper, we formulate the expert finding task as a tree matching problem with the hierarchical knowledge representation. The experimental results demonstrate the advantage of using 5-level PAM and semantic enhancement against n-gram models and LDA-like models. To further improve the work, we will incorporate more information to enrich the hierarchical representation in the future.

Acknowledgements

The work described in this paper was supported by the grants from the Research Grants Council of Hong Kong (PolyU 5202/12E and PolyU 152094/14E) and the grants from the National Natural Science Foundation of China (61272291 and 61273278).

References

Eugene Agichtein, Carlos Castillo, Debora Donato, et al. 2008. Finding high-quality content in social media. In *Proc. of WSDM*.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proc. of ACL*.

Yoshua Bengio, Rjean Ducharme, Pascal Vincent, et al. 2003. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3: 1137-1155.

Marc Bernard, Laurent Boyer, et al. 2008. Learning probabilistic models of tree edit distance. *Pattern Recognition*, 41(8): 2611-2629.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *The Journal of Machine Learning Research*, 3: 993-1022.

Mohamed Bouguessa, Benot Dumoulin, and Shengrui Wang. 2008. Identifying authoritative actors in question-answering forums: the case of yahoo! answers. In *Proc. of SIGKDD*.

George Casella and Roger L. Berger. 2001. Statistical Inference. *Duxbury Press*.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proc. of EMNLP*, pages 740750.

Fei Cheng, Kevin Duh, Yuji Matsumoto. 2014. Parsing Chinese Synthetic Words with a Character-based Dependency Model. *LREC*.

Allan M. Collins and M. Ross. Quillian. 1969. Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behaviour*, 8: 240247.

Ronan Collobert, Jason Weston, Leon Bottou, et al. 2011. Natural language processing (almost) from scratch. *JMLR*, 12.

Paramveer Dhillon, Dean P Foster, and Lyle H Ungar. 2011. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems*, pages 199207.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, et al. 2008. LIBLINEAR: A library for large linear classification. *The Journal of Machine Learning Research*, 9: 1871-1874.

Eric H. Huang, Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2012. Improving Word Representations via Global Context and Multiple Word Prototypes. In *Proc. of ACL*.

Akshay Java, Pranam Kolari, Tim Finin, et al. 2006. Modeling the spread of influence on the blogosphere. In *Proc. of WWW*.

Akshay Java, Xiaodan Song, Tim Finin, and Belle Tseng. 2007. Why we Twitter: Understanding Microblogging Usage and Communities. In *Proc. WebKDD-SNA-KDD*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global Vectors for Word Representation. In *Proc. of EMNLP*.

Pawel Jurczyk and Eugene Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proc. of CIKM*.

David Kempe, Jon Kleinberg, and Eva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proc. of SIGKDD*.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, et al. 2014. A dependency parser for tweets. In *Proc. of EMNLP*, pages 10011012, Doha, Qatar, October.

Balachander Krishnamurthy, Phillipa Gill, and Martin Arlitt. 2008. A few chirps about Twitter. In *Proc. of the first workshop on Online social networks. ACM*, pages 19-24.

Remi Lebret, Jo el Legrand, and Ronan Collobert. 2013. Is deep learning really necessary for word embeddings? In *Proc. of NIPS*.

Omer Levy and Yoav Goldberg. 2014. Dependency-based word embeddings. In *Proc. of ACL*.

Omer Levy, Yoav Goldberg, And Ido Dagan. 2015. Improving Distributional Similarity with Lessons Learned from Word Embeddings. In *Proc. of TACL*.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: DAG-structured mixture models of topic correla-tions. In *Proc. of the 23rd international conference on Machine learning. ACM*, pages 577-584.

Wei Li and Andrew McCallum. 2008. Pachinko allocation: Scalable mixture models of topic correlations. *Journal of Machine Learning Research.*

Shujie Liu, Nan Yang, Mu Li, and Ming Zhou. 2014. A recursive recurrent neural network for statistical machine translation. In *Proc. of ACL*, pages 1491 1500.

Minh-Thang Luong, Richard Socher, and Christopher D. Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology. In *Proc. of CoNLL.*

George A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):3941.

Thomas P. Minka. 2000. Estimating a Dirichlet distribution. Technical report, MIT.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013(a). Efficient estimation of word representations in vector space. CoRR, abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013(b). Distributed representations of words and phrases and their composition-ality. In *Advances in Neural Information Processing Systems*. pages 3111-3119.

Aditya Pal and Joseph A. Konstan. 2010. Expert Identification in Community Question Answering: Exploring Question Selection Bias. In *Proc. of the 19th ACM international conference on Information and knowledge management. ACM*, pages 1505-1508.

Aditya Pal and Scott Counts. 2011. Identifying topical authorities in microblogs. In *Proc. of the fourth ACM international conference on Web search and data mining. ACM*, pages 45-54.

Siyu Qiu, Qing Cui, Jiang Bian, and et al. 2014. Co-learning of Word Representations and Morpheme Representations. In *Proc. of COLING.*

Daniele Quercia, Harry Askham, and Jon Crowcroft. 2012. TweetLDA: supervised topic classification and link prediction in Twitter. In *Proc. of the 4th Annual ACM Web Science Conference. ACM*, pages 247-250.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-label corpora. In *Proc. of EMNLP.*

Daniel Ramage Susan Dumais, and Dan Liebling. 2010. Characterizing Microblogs with Topic Models. In ICWSM, 5(4): 130-137.

Ana Raposo, Mafalda Mendes, and J. Frederico Marques. 2012. The hierarchical organization of semantic memory: Executive function in the processing of superordinate concepts. *NeuroImage*, 59: 18701878.

Stanley M. Selkow. 1977. The tree-to-tree editing problem. *Information processing letters*, 6(6): 184-186.

Mahdi M. Shafiei and Evangelos E. Milios. 2006. Latent Dirichlet coclustering. In *Proc. of International Conference on Data Mining*, pages 542-551.

Dennis Shasha and Kaizhong Zhang. 1990. Fast algorithms for the unit cost editing distance between trees. *Journal of algorithms*, 11(4): 581-621.

Yaming Sun, Lei Lin, Duyu Tang, and et al. 2014. Radical-enhanced chinese character embedding. *arXiv preprint arXiv:1404.4714.*

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 31043112.

Jie Tang, Jing Zhang, Limin Yao, et al. 2008. Arnetminer: Extraction and mining of academic social networks. In *Proc. of SIGKDD.*

Duyu Tang, Furu Wei, Nan Yang, et al. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proc. of ACL.*

Robert A. Wagner. 1975. On the complexity of the extended string-to-string correction problem. In *Proc. of seventh annual ACM symposium on Theory of computing*. pages 218-223. ACM.

Robert A. Wagner and Michael J. Fischer. 1974. The string-to-string correction problem. *Journal of the ACM (JACM)*, 21(1), 168-173.

Wang Ling, Chris Dyer, Alan Black, and Isabel Trancoso. 2015. Two/too simple adaptations of word2vec for syntax problems. In *Proc. of NAACL*, Denver, CO.

Jianshu Weng, Ee Peng Lim, Jing Jiang and Qi He. 2010. Twitterrank: finding topic-sensitive influential twitterers. In *Proc. of WSDM.*

Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. 2013. Connecting language and knowledge bases with embedding models for relation extraction. In *Proc. of Computation and Language.*

Yi Yang and Jacob Eisenstein. 2015. Unsupervised multi-domain adaptation with feature embeddings. In *Proc. of NAACL-HIT.*

Mo Yu and Mark Dredze. 2014. Improving lexical embeddings with semantic knowledge. In *Proc. of ACL.*

Jun Zhang, Mark S. Ackerman, and Lada Adamic. 2007. Expertise networks in online communities: structure and algorithms. In *Proc. of WWW.*

Kaizhong Zhang and Dennis Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM journal on computing*, 18(6): 1245-1262.

Meishan Zhang, Yue Zhang, Wan Xiang Che, and et al. 2013. Chinese parsing exploiting characters. In *Proc. of ACL.*

Xin Zhao and Jing Jiang. 2011. An empirical comparison of topics in twitter and traditional media. *Singapore Management University School of Information Systems Technical paper series. Retrieved November*, 10: 2011.