Word Similarity Fails in Multiple Sense Word Embedding

Yong Shi[1,3,4,5], Yuanchun Zheng[2,3,4], Kun Guo[*,3,4,5], Wei Li[3,4,5], and Luyao Zhu[3,4,5]

[1] College of Information Science and Technology, University of Nebraska at Omaha, NE 68182, USA
[2] School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing, 101408, China
[3] Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences, Beijing 100190, China
[4] Research Center on Fictitious Economy & Data Science, Chinese Academy of Sciences, Beijing 100190, China
[5] School of Economics and Management, University of Chinese Academy of Sciences, Beijing, 100190, China

Abstract. Word representation is one foundational research in natural language processing which full of challenges compared to other fields such as image and speech processing. It embeds words to a dense low-dimensional vector space and is able to learn syntax and semantics at the same time. But this representation only get one single vector for a word no matter it is polysemy or not. In order to solve this problem, sense information are added in the multiple sense language models to learn alternative vectors for each single word. However, as the most popular measuring method in single sense language models, word similarity did not get the same performance in multiple situation, because word similarity based on cosine distance doesn't match annotated similarity scores. In this paper, we analyzed similarity algorithms and found there is obvious gap between cosine distance and benchmark datasets, because the negative internal in cosine space does not correspond to manual scores space and cosine similarity did not cover semantic relatedness contained in datasets. Based on this, we proposed a new similarity methods based on mean square error and the experiments showed that our new evaluation algorithm provided a better method for word vector similarity evaluation.

1 Introduction

Word embedding is an effective distributed method for word representation in natural language precessing (NLP) which can obtain syntax and semantic information from amount of unlabeled corpus. Comparing with local representation like one-hot encoding, distributed representation maps word or sentence to a dense low dimensional vector space. And properties like word syntax and semantic information distributed on all dimensions. However, these models assumed

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

that each word only have one single vector and ignored polysemy situation in languages which called word disambiguation. For instance, '*apple*' is not only a technology company but also a kind of fruits. In a sentence, once the context environment determined, the meaning of the word is also determined. According to this, multiple sense word embedding is proposed, each word has various number of word vectors and corresponding sense vectors. And the biggest difference between single sense and multiple sense language model is the sense information. Therefore, we can choose proper word vector based on sense information and get more accurate representation in sentences.

Another problem after get embedding is the evaluation process. Different language models has different patterns and structures, some of them focus on word similarity and others focus on word syntax and semantic. To summarize, researchers proposed two different evaluations called intrinsic and extrinsic evaluation, which evaluated word vectors with different priorities. Intrinsic evaluation includes word similarity, word analogy and synonym question, extrinsic evaluation includes experiments which using embedding as the initialization of neural networks like text classification, semantic analysis and named entity recognition. But there are none one evaluation algorithm covered all these evaluations, intrinsic evaluation methods may fail in extrinsic measuring, so evaluation methods become more and more important in language models and NLP tasks.

In intrinsic evaluation, word similarity algorithm is the most-used method to measure semantic similarity. But the vector scores computed by cosine are different with manual annotated similarity scores, because cosine value only measure similarity while annotated value contained similarity and relatedness at the same time, which caused mismatch in similarity evaluation. In this paper, we analyzed multiple sense language models and word similarity evaluations, and studied the reason which caused mismatch between cosine scores and manual scores. Based on this, we proposed a new method based on mean square error to evaluate the performance of different language models. We also proposed a new word similarity benchmark datasets with our new annotation method.

This paper is organized as follows: section 2 reviewed multiple sense word embedding models, and word similarity evaluation methods are analyzed and our new evaluation method is proposed in section 3, section 4 listed some experiments results between different word similarity evaluations, and we gave our conclusion in the final section.

2 Related Work

Raw words or sentences need to be represented by vectors before input into algorithms, therefore representation learning had became foundational field in NLP. In 1954, Harris [1] proposed distributed hypothesis which point that syntax would be similar if words had the same contexts, and this hypothesis was a footstone in language modeling. Researcher designed lots of models according to the distributed hypothesis. Assume $S = (w_1, w_2, \ldots, w_n)$ is a sentence with n words. We use $P(S)$ in 1 to represent the joint probability of this sentence, and proba-

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

bility would be decrease if replacing any word in S. The gold of language model is to maximum joint probability of a sentence, but the difficult of conditional probability would be growth exponential in the tail of S because we need to consider more history words. Therefore, a context window cover $2k$ words would be chosen to replace the whole sentence, $C(w_i) = w_{i-k}, \ldots, w_{i-1}, w_{i+1}, \ldots, w_{i+k}$ means all context words in this window.

$$P(S) = \prod_{i=1}^{n} p(w_i|w_1, w_2, \ldots, w_{i-1}) \approx \prod_{i=1}^{n} p(w_i|C(w_i)) \tag{1}$$

$$P(S) = \prod_{i=1}^{n} \prod_{j=1}^{a_i} p(w_i^j|C(w_i^j)) \approx \prod_{i=1}^{n} \prod_{j=1}^{a_i} p(w_i^j|C(w_i)) \tag{2}$$

In single sense models, each word has only one vector w_i, the goal of language model is to maximum the probability of S. When extend to multiple sense language model, each word has a_i word vectors $(w_i^1, w_i^2, \ldots, w_i^{a_i})$ and corresponding sense vectors $(c_i^1, c_i^2, \ldots, c_i^{a_i})$, models must decide which sense is the best one in current context from candidate senses. Once determine sense index, word vector would be updated by the same algorithm with single sense models. Therefore, we can modified sentence probability with 2. In this equation, sense information and corresponding vectors are added as extra data in the inner multiplication.

Bengio [2] first exploited a feed-forward neural network for modeling language, they used a three layer neural network to modeling sentence and got a word embedding at the same time. But it is very time consuming because the output layer is a softmax layer and the number of neural units are the same with dimension of dictionary. Mnih [3] proposed a Log-Bilinear language model (LBL), in their model, a bilinear structure replace the three layers and accelerate the algorithms effectively. After that, Mikolov [4] tried to used recurrent neural network to model language because sentences are sequence data essentially. They [5] also proposed two popular language models named continuous bag-of-words (CBOW) model and skip-gram model, and two efficient accelerate algorithms named hierarchical softmax and negative sampling. 2(a) indicated two model structures in word2vec, the CBOW model predict target word w_i by its neighbor context words $C(w_i) = \{w_{i-k}, \ldots, w_{i-1}, w_{i+1}, w_{i+k}\}$, and Skip-gram had a symmetrically structure that predict context word by target word.

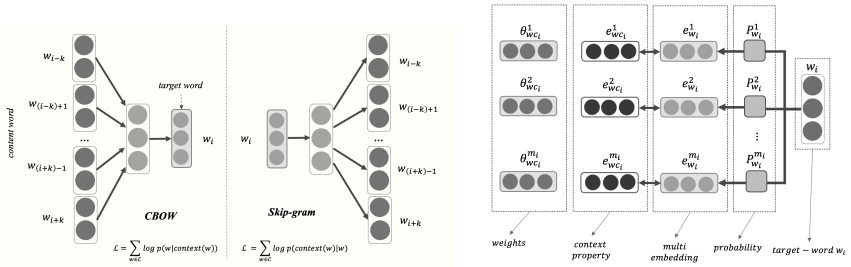
Collobert [6] designed a more complex network with a score function to replace the softmax layer. They used a union probability $p(w_1, w_2, \ldots, w_n)$ in 3 rather than a approximate conditional probability, and $f_\theta(x)$ is a neural network based score function. Original sentence S is a positive sample, and $S^{(w)}$ which central word was replaced by a random chosen word is a negative sample. Then made the score of positive sample at least one point more than negative samples.

$$P(S) = \sum_{w \in \mathcal{D}} \max\{0, 1 - f_\theta(S) + f_\theta(S^{(w)})\} \tag{3}$$

All the previous models only considered local context and ignore the global information, and Pennington [7] proposed Glove model which has the same

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

Fig. 1. Single sense and multiple sense language models



(a) Two models in word2vec (b) Multiple sense unit

thought as previous models but considered more situation of the concurrence of words. This mode adapted advantages both from latent semantic analysis [8] and CBOW model. The results showed that Glove perform well in small corpus and more flexible in big corpus than word2vec. Except unsupervised learning models, Joulin [9] proposed a supervised method based on text classification to train word vectors and reduce the learning time from days to several hours, Bojanowski [10] put sub-words into models as extra information and enriched samples for training, they also released fastText[1] toolkits to train embeddings or to finish text classification task. No matter predicate-based or score function-based models, they all take advantage of the great power of the neural network. More and more complex and useful networks began adapted in language models. As a special design of recurrent neural network, long-short memory network (LSTM) is a better way to model sequence data with different gates.

Models mentioned in previous part are all single sense language models which assumed that one word have only one vector no matter how many different context environment may occur in corpus. Several efforts have been made in multiple sense language modeling. Huang [11] proposed a two-stage learning methods based on cluster algorithms, they first clustered words to many groups and labeled word using theirs cluster center, then trained word vector by these replaced centers. But clustering process would be the biggest block in this algorithm because it spend lots of time. Effort have been made to solve this problem. Neelakantan [12] proposed a non-parametric estimation methods based on skip-gram model to learn multiple prototype vectors, they used a latent variable to represent the sense vector and get better performance in word similarity benchmark. Zheng [13] designed a universal multiple sense unit and embed into CBOW model to learning multiple word embedding and got different representations for each word. Tian [14] proposed a new method based on EM algorithm Li [15] proposed a new model to learn sense information and used Chinese Restaurant Process to recognize the number of senses for different word. 2(b) is a basic cell

[1] https://github.com/facebookresearch/fastText

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

structure in multiple sense language model, it is more complex than an ordinary neural unit.

Some researchers began to explore the internal explanation of similarity evaluation. Gladkova [16] discussed methods of intrinsic evaluation of word embeddings and hoped to draw attention of both computational and theoretical linguists to get a better evaluation method. Chiu [17] found word embeddings can't get the same performance in intrinsic or extrinsic evaluations. Li [15] explored if multiple sense word embedding can improve natural language understanding and found that single sense word embedding can beat multiple sense word embedding with a bigger dimension size. All of this lead to the suspicion of multiple sense models.

3 Methodology

Another important part in natural language processing is the evaluation because we don't know if vectors contains similarity or analogy properties or not. Language models based on distributed hypothesis consider words have same meaning if they have same context, therefore, similarity evaluation is the most intuitive test for the quality of language models and word vectors. As one of the most used intrinsic evaluation, cosine distance indicated in 4 measured semantic distance between word pairs, and this test has been the most popular evaluations.

$$cos(w_1, w_2) = \frac{w_1 w_2}{\|w_1\|\|w_2\|} \tag{4}$$

In multiples sense models, each word may contains many vectors, Huang proposed another four different similarity distances with multiple vectors. In 5, $AvgSim$ means average of all the possible match of word vectors, and $AvgSimC$ adds the probability of every sense occur in corpus. And the $LocalSim$ only use the best match one to represent all the situations, this may be the most close to multiple sense vectors at all. And another special distance is global similarity which is the cosine distance between two global vectors.

$$AvgSim(w_1, w_2) = \frac{1}{K^2} \sum_{i=1}^{K} \sum_{j=1}^{K} d(v_i(w_1), v_j(w_2)) \tag{5}$$

$$AvgSimC(w_1, w_2) = \frac{1}{K^2} \sum_{i=1}^{K} \sum_{j=1}^{K} p(c_1, w_1, i) p(c_2, w_2, j) d(v_i(w_1), v_j(w_2)) \tag{6}$$

$$LocalSim(w_1, w_2) = d(v_s(w_1, k_1), v_s(w_2, k_2) \tag{7}$$

$$GlobalSim(w_1, w_2) = d(v_g(w_1), v_g(w_2)) \tag{8}$$

Embedding trained by neural networks whose object function is 1 or 2 would catch semantic information like word similarity and word relatedness. But these vectors represent words' context situations other than word property itself. For

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

example, 'good' and 'bad' would be have a high cosine score because they always appear in the same context environment. Similarity score computed by word vector pairs is in the range $[-1, 1]$ while manual annotated scores in the range $[0, 10]$. The negative part would be difficult to map in the same scope.

WS353 is a common used dataset with 353 word pairs and their manual annotated similarity scores. For example, '*tiger cat 7.35*' is one record in WS353, two words '*tiger*' and '*cat*' are word pairs used in similarity algorithms, value in the third column means annotated similarity score of word pairs which range from 0 to 10, higher score represent two word are more similar. After get similarity score of each word vector pairs, correlation coefficient in 9 is one index that can measure the degree of correlation between estimated scores and human language. And X is a cosine similarity vector based on cosine distance, Y is a same size vector of annotated similarity scores. A higher value means that word embedding catch more semantic information and grammar structures.

$$\rho(X, Y) = \frac{\sum_{i=1}^{n}(x_i - \bar{X})(y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{X})\sum_{i=1}^{n}(y_i - \bar{Y})}} \tag{9}$$

In order to analysis the effectiveness of word embeddings, two new evaluation algorithms were proposed in this paper. The first one is fake similarity, we used manual annotated scores to find the best match word vectors and recomputed cosine distance and pearson correlation coefficient. The fake similarity was designed as 10, $v_f(x)$ means vector x are the most similar to manual scores from word's multiple embeddings. This fake similarity measures the best conditions that embedding can achieve compared to human language. 'Fake' means that annotated scores were already used in advance.

$$FakeSim(w_1, w_2) = \frac{1}{K^2} d(v_f(w_1), v_f(w_2)) \tag{10}$$

The second evaluation method is mean square error (MSE) in 11, it is the difference between manual scores and what is estimated.

$$MSE(X, Y) = \frac{1}{n}\sum_{i=1}^{n}(X^i - Y^i)^2 \tag{11}$$

There are many similarity benchmark datasets in intrinsic evaluation, 3 listed some similarity benchmark datasets that always used in intrinsic evaluation. WS353 is the most used dataset included 353 word pairs and their annotated similarity scores. Another two datasets created from WS353 are WS353Sim and WS353Rel, and WS353Sim include 203 records that each word pair have similarity while WS353Rel contains 252 records that each word pair are more related to each other. As for multiple sense word embedding, SCWS dataset provide 2003 word pairs with their context environment and annotated scores. Each record has two words and two whole sentences including two words. Datasets like this can be used both in single sense word embedding and multiple sense embedding.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

Table 1. Datasets used in word similarity evaluation

Dataset	Word Paris	Evaluation	Source
WS353	353	single sense	Finkelstein [18]
WS353Sim	203	single sense	Agirre [19]
WS353Rel	252	single sense	Agirre [19]
SCWS	2003	multiple sense	Huang [11]
MEN	3000	single sense	Bruni [20]

Cosine distance is range from -1 to 1 while manual annotated scores did not cover the negative scope. Therefore, there must be a mismatch between two scores. To solve this problem, we can shift all the values to positive part, but zero is a special point in this situation. And there's another method, we can mirror all negative scores to positive range.

4 Experiments

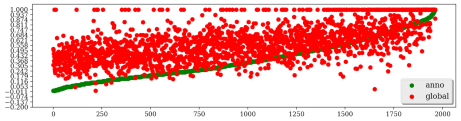
Multiple sense word embedding is the best method for word disambiguation in unsupervised learning filed. Many researchers proposed their own well-designed models for learning word vectors and share their own datasets for word similarity evaluation. Multiple sense skip gram is one of the best multiple sense language model, we download their trained word vectors and got computed four correlation coefficient using similarity algorithms mentioned in 5 and 10. In the experiment, we choose pre-trained multiple sense word vectors with 50 dimension size choose sense window is 5 (with 5 words in both left and right fo current word). In MSSG word vectors, each word has a global vector and corresponding sense vectors, global vector updated every time while sense vector updated when match its context vector in the training process. In fact, global vector is the same as single sense word embedding.

As for benchmark dataset, we removed all the records that didn't not appear in MSSG dictionary. After getting the cosine scores and annotated scores, we reorder the annotated scores from small to large and just showed by the green point line in 4, and the red point means the paired cosine scores.

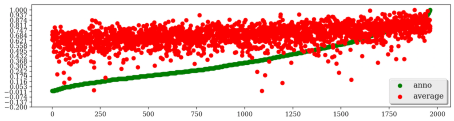
In 4, the distance between red point and green point reflect the offset between manual scores and cosine scores. We can find in 3(a) 3(b) 3(c) 3(d) that computed cosine scores are a little bit close to the positive range. We can get more accurate numerical comparison in 4. In this table, subscript number 1 means we used 1947 pairs of valid words from SCWS datasets, and subscript number 2 means the left 1723 word pairs after removing 241 word pairs with duplicate word. $]rho$ is the correlation coefficient and e means MSE while $e^{'}$ is the result after normalization. From the result we can find that local similarity did not get best performance while global vector get the highest value in correlation measuring. And we add global vector into sense vectors to enrich the source of $LocalSim2$, but it also fails to overtaken the $GlocalSim$. There are many reasons that can influence the results include context windows and similarity algorithms.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

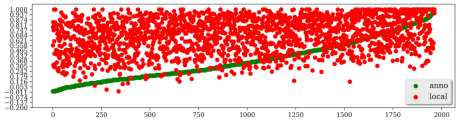
Fig. 2. Similarity of scores scatter with different cosine algorithms



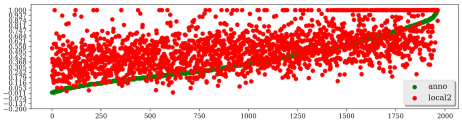
(a) annotated VS global



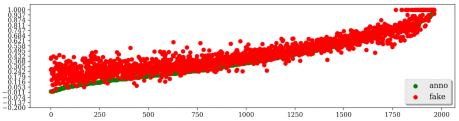
(b) annotated VS average



(c) annotated VS local



(d) annotated VS local2



(e) annotated VS fake

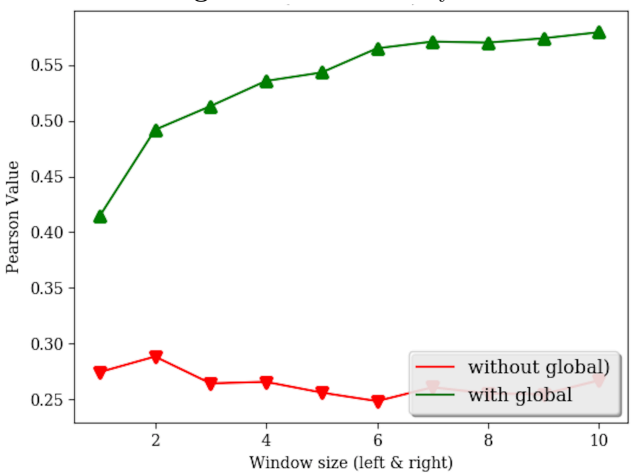
Table 2. Correlation coefficient and MSE on MSSG word vectors

-	**GlobalSim**	**AvgSim**	**LocalSim**	**LocalSim2**	**FakeSim**
ρ_1	0.634	0.481	0.256	0.543	0.946
ρ_2	0.543	0.388	0.248	0.456	0.943
e_1	16.767	16.493	15.981	16.935	-
e_2	12.077	11.333	10.882	11.998	-
$e_1^{'}$	15.807	15.585	15.633	16.294	-
$e_2^{'}$	10.968	10.540	10.585	11.303	-

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

We also analyzed the influence of different context window size indicated in 4. Because MSSG word vectors are trained with context window 5, so we choose the context window size in range [1,10]. The results showed that sense vector didn't play ideal role, but the global vector perform better. We guess that global vector catch more semantic information and sense vectors didn't get fully trained and even get incorrect updated. Therefore, single sense word embedding or global vector would be enough in most NLP tasks.

Fig. 3. Window size analysis



5 Conclusion

Multiple sense word embedding is a better way for word disambiguation, it considers specific meaning of a word in context environment. But models may not get better performance if it can not determine sense information. Sometimes, single sense or global word vector can achieve same results with multiple sense vectors and more flexible during serialization process. As most used intrinsic evaluation, word similarity is an efficient measuring for word vectors. But the gap between estimated annotated scores and cosine similarity-based scores influence the performance of similarity evaluation. In this paper, we designed a new similarity distance $fakeSim$ and used mean square error to evaluate the performance of word similarity and found that word similarity fails in multiple sense word embedding evaluation, there are still a long way to reach the desired goal.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

Acknowledgements

This work is supported by the National Natural Science Foundation of China No. 91546201, No. 71331005 and No. 71501175, Shandong Independent Innovation and Achievement Transformation Special Fund of China (2014ZZCX03302), and the Open Project of Key Laboratory of Big Data Mining and Knowledge Management, Chinese Academy of Sciences.

References

1. Harris, Z.S.: Distributional structure. Word **10**(2-3) (1954) 146–162
2. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. Journal of machine learning research **3**(Feb) (2003) 1137–1155
3. Mnih, A., Hinton, G.: Three new graphical models for statistical language modelling. In: Proceedings of the 24th international conference on Machine learning, ACM (2007) 641–648
4. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech. Volume 2. (2010) 3
5. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
6. Collobert, R., Weston, J.: A unified architecture for natural language processing: Deep neural networks with multitask learning. In: Proceedings of the 25th international conference on Machine learning, ACM (2008) 160–167
7. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). (2014) 1532–1543
8. Landauer, T.K., Foltz, P.W., Laham, D.: An introduction to latent semantic analysis. Discourse processes **25**(2-3) (1998) 259–284
9. Joulin, A., Grave, E., Bojanowski, P., Mikolov, T.: Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759 (2016)
10. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. arXiv preprint arXiv:1607.04606 (2016)
11. Huang, E.H., Socher, R., Manning, C.D., Ng, A.Y.: Improving word representations via global context and multiple word prototypes. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, Association for Computational Linguistics (2012) 873–882
12. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient non-parametric estimation of multiple embeddings per word in vector space. arXiv preprint arXiv:1504.06654 (2015)
13. Zheng, Y., Shi, Y., Guo, K., Li, W., Zhu, L.: Enhanced word embedding with multiple prototypes. In: Industrial Economics System and Industrial Security Engineering (IEIS'2017), 2017 4th International Conference on, IEEE (2017) 1–5
14. Tian, F., Dai, H., Bian, J., Gao, B., Zhang, R., Chen, E., Liu, T.Y.: A probabilistic model for learning multi-prototype word embeddings. In: COLING. (2014) 151–160
15. Li, J., Jurafsky, D.: Do Multi-Sense Embeddings Improve Natural Language Understanding? CoRR (2015)
16. Gladkova, A., Drozd, A.: Intrinsic Evaluations of Word Embeddings: What Can We Do Better? In: Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP, Stroudsburg, PA, USA, Association for Computational Linguistics (2016) 36–42

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38

17. Chiu, B., Korhonen, A., Pyysalo, S.: Intrinsic evaluation of word vectors fails to predict extrinsic performance. In: Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP. (2016) 1–6
18. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppin, E.: Placing search in context: The concept revisited. In: Proceedings of the 10th international conference on World Wide Web, ACM (2001) 406–414
19. Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Paşca, M., Soroa, A.: A study on similarity and relatedness using distributional and wordnet-based approaches. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Association for Computational Linguistics (2009) 19–27
20. Bruni, E., Boleda, G., Baroni, M., Tran, N.K.: Distributional semantics in technicolor. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1, Association for Computational Linguistics (2012) 136–145

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93701-4_38