ISSN 1342-2812

Research Reports on Mathematical and Computing Sciences

Linguistic Regularities from Multiple Samples

Aleksandr Drozd and Satoshi Matsuoka

February 2016, C–283

Department of
Mathematical and
Computing Sciences
Tokyo Institute of Technology

SERIES C: Computer Science

Abstract

We present a simple method for better retrieval of analogical relations from word embeddings by learning from the multiple related pairs of words. We provide performance evaluation of the proposed method in comparison with the state-of-the-art approaches based on vector offset.

1 Introduction

The recent boom of research on analogies with word embedding models is largely due to the striking demonstration of "linguistic regularities" between vectors in [1]. In [2] the authors showed that certain relations between lexical units can be captured as linear offsets in the vector space of word embeddings. The typical task to evaluate this mechanism is solving an analogy problem of the *a:b :: c:d* type. The two pairs of words (*a:b* and *c:d*) are supposed to have the same linguistic relation; the word *d* is unknown, and the task is to find it as the vector closest to the result of vector calculation from words *a, b,* and *c* (a frequently cited example is "king" - "man" + "woman" = "queen"). A big analogy test set for evaluating performance of word embedding models was introduced in [1]. Many subsequent studies used the same test set for evaluating performance of their models; the top current result is over 80% accuracy [3].

This line of research is very promising, since reliable detection of analogies can help with detection of virtually any other semantic relation, as shown in [4]. Its results would also have practical value for many tasks, as analogical reasoning has long been used in morphological analysis, word sense disambiguation, improving search results, and other areas. However, the vector offset method has its limitations, as it is very sensitive to idiosyncrasies of individual words. This work proposes an alternative, machine-learning-based method for solving analogy problems of the *a:b :: c:d* type.

2 Vector-Space Models: general methodology

Words are textual data, but machine-learning algorithms such as regression models and clustering algorithms require numerical data. The "conversion" of words into vectors is made possible with the distributional view of language: *similar words appear in similar contexts* [5].For example, the word *cat* is more likely to be found in the context of words like *mouse*, *milk*, or *tree* than with words like *skyscraper*, *peptides* or *dictionary*. Some researchers suggest abandoning standard dictionary definitions in favor of distributional similarity-based description [6].

A vector space model (VSM) represents words as vectors in multi-dimensional space, where each dimension is a possible context where words may or may not occur. This allows for representing various relations between words through the spatial relations of the corresponding vectors. In the simplest case, two words with similar meaning should be close in vector space under a certain metric. One of the biggest success stories of VSMs is solving TOEFL synonymy tests with 100% accuracy; in these tests the task is to choose on of the four given options which is the synonym of the target word. This can be done simply by choosing the option which has the higher proximity score in the vector space [7].

2.1 Sparse vectors

Depending on the size and type of context, vector space models can be document-based, window-based or syntax-based. Each of these models have various additional parameters, including the size of the window, penalty for the larger distance to the target word inside the window (triangular, Gaussian etc [8]), and also whether the left and right contexts are counted together or independently. The choice of parameters depends on the goals of a particular project, as different parameters perform better for different kinds of tasks.

Each dimension of VSM is determined as probability p of a context word c with respect to the target word t: $p(c|t) = \frac{p(c,t)}{p(t)}$. But raw word frequency as a measure for "belongingness" to certain context can be biased by the total corpus-scale frequency, as frequent words will be more frequent in any context. TO cope with this effect it is possible to use the Pointwise Mutual Information (PMI) associated measure [9]. PMI quantifies the discrepancy between probability of the coincidence of two random variables, given their joint distribution and their individual distributions, and assuming their independence: $pmi(c,t) \equiv \log \frac{p(c,t)}{p(c)p(t)} = \log \frac{p(c|t)}{p(c)}$

In this project the VSM is built with window size 2 (both left and right contexts). Initially we build a matrix where each row is a feature-vector of a term, and each element of the vector is determined by the PMI-weighted frequency of this term in each context.

2.2 Dimensionality reduction in "explicit" word embeddings

The procedure described in section 2.1 creates the so-called sparse vectors: they are very big, since the number of dimensions is defined by the number of possible contexts. For example, Araneum Russicum Maius corpus [10] (1.2 billion tokens), discarding all words with frequency less then 5, yields about 2 million possible contexts. However, the resulting co-occurrence matrix had only 0.5 billion non-zero elements (0.02% sparsity) and could be stored in about 2 GiB of memory space in compressed sparse row format.

Sparse vectors can be used as is for some tasks, such as computing vector-to-vector angular distance, but their size makes them prohibitive for using with artificial neural networks. Also, sparse vectors are more sensitive to noise in data which creates random co-occurrences. Dimensionality reduction of co-occurrence matrices can improve the performance of machine-learning techniques both in terms of accuracy and computation time. Its positive effect has been reported in many VSM applications [11, 12, 7].

Two popular techniques for dimensionality reduction are Principal Component Analysis (PCA) [13] and Singular Values Decomposition (SVD) transformations on the co-occurrence matrix.In this work we use SVD to construct the low-tank approximation of the co-occurrence matrix [14].

SVD is a factorization of an $m \times n$ real or complex matrix M in a form $M = U\Sigma V^*$ [15], where U is $m \times m$ real or complex unitary matrix, Σ is $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal, and V^* (the conjugate transpose of V, or simply the transpose of V if V is real) is $n \times n$ real or complex unitary matrix. The diagonal entries σ_i, of Σ are known as the singular values of M and are typically sorted in descending order. The matrices U and V contain left-singular vectors and right-singular vectors of M respectively.

2.3 Other approaches

Techniques for constructing VSMs on the basis of raw frequency counts are referred to as "explicit" or count-based VSMs. In addition to SVD and PCM, there are other ways to perform dimensionality reduction. It is also possible to map words (and possibly phrases) to vectors in a low dimensional space directly, without counting the co-occurrences [16].

A prominent class of word embeddings is based on artificial neural networks. A neural net takes a word in a straight-forward "one hot" representation, and it is trained to predict words in a given context. This approach recently attracted much attention after Mikolov et al showed that they capture certain "linguistic regularities", which will be discussed below. However, conceptually this approach is also rooted in the distributional hypothesis, and Levy and Goldberg (2014) showed that neural embeddings essentially perform implicit co-occurrence matrix factorization [17], and they share many properties with the "explicit" models [18]. In this work we use the "explicit" method for obtaining dense word vectors as it proved to be faster in our pilot study and provides more intuitive control over various model parameters.

3 Linguistic regularities

While the initial application of vector space models was to capture word similarity, measured as angular distance between vectors, it has been shown that word embeddings are capable of capturing more complex relations between words. "Linguistic regularities" is a term that denotes any morphological or semantic relation between groups of word pairs, such as relations between infinitives and gerunds (*play : playing :: walk : walking :: sing : singing*) or between countries and their capitals (*France : Paris :: Spain : Madrid :: Japan : Tokyo*).

Linguistic regularities have been used mostly as demonstration of the possibility of analogical reasoning with the help of word embeddings. The currently largest and widely used test set, known as "Google" dataset, comprises 5 semantic and 9 syntactic categories. Each category has 20-70 unique example pairs (*a:b :: c:d*), from which random combinations of pairs are generated. Each possible pair of pairs becomes a question in the test, such as "Tokyo:Japan, Paris:?", "Tokyo:Japan, Madrid:?", and so on . This set contains 8,869 semantic and 10,675 syntactic questions in total.

4 Existing methods for retrieving regularities

Mikolov et al.[2] suggested that linguistic regularities can be captured as the offset of their vector embeddings. The answer to the question "a is to b as c is to ?" is represented by hidden vector d, and the answer is calculated as $argmax_{b^* \in V}(sim(d, c - a + b))$. Here V is the vocabulary excluding words a,b and c. sim is a similarity measure, for which Mikolov and most practitioners use angular distance: $sim(u, v) = cos(u, v) = \frac{u \cdot v}{||u||||v||}$ In the remainder of this work this method will be referred to as **PairDistance**.

Levy at all [18] propose an alternative optimization objective: $argmax_{d \in V}(cos(d - c, b - a))$ They report that this method produces more accurate results for some categories. Essentially this method estimates the extent to which vectors $d - c$ and $b - a$ share direction, and discards the lengths of these vectors. This method will be referred to as **3CosAdd**.

5 Learning from multiple examples

Learning a relation based on just one pair of words has obvious caveats due to linguistic confounds - additional characteristics of words that are not shared by the two words in an analogical pair. For example, *king* and *queen* are supposed to be examples of the *masculine:feminine* relation, but these two words have different polysemy networks, which results in additional differences in their word embeddings. *Queen* is also a musical group, and so the word *queen* appears in a lot of contexts in which the word *king* does not appear.

The alternative approach is to learn linguistic relations from a set of example pairs. The naive solution is to average the vector offset between every pair in a training set; however, in our tests, this method did not cause significant improvement in relation retrieval accuracy. We propose an alternative interpretation of linguistic relations, given that they are to be deduced from a set of pairs. Let all the right-hand-side elements represent a class. Then the question "what is related to c as b is related to a" can be expressed in a form "which word belongs to the same class as b and at the same time is closest to c".

Under this interpretation the following method is proposed for analogy retrieval. First, we train a binary classifier, for example logistic regression, to predict if the word belongs to the right-hand-side class. We use left-hand-side along with random samples from the dictionary as negative training samples. Then the probability for the target word d to be the correct answer in a given analogical problem is obtained by combination of the probability of this word belonging to the target class and its similarity with c, measured as an angular distance. The simplest way to combine the two measures is a product of the probability with which the evaluated word d belongs to the target linguistic class, and its similarity with its pair c. In the following section we evaluate the performance of this method.

6 Performance evaluation

We used the Google[1] test set for the benchmark. The word embeddings were obtained by the PMI-weighted coinsurance matrix factorization with SVD algorithm, using our tool-chain described in [19]. For this test we produced 1000-dimensional vectors from the English corpus which combined the English Wikipedia snapshot from July 2015 (1.8B tokens), Araneum Anglicum Maius (1.2B) [10] and ukWaC (2B) [20]. We uncased all words and discarded those occurring less than 100 times, which resulted in the vocabulary of 301,949 words.

To learn the target relations we used logistic regression - a classification algorithm in "exclude one" scheme, i.e. the classifier was re-trained each time on all right-hand-side words excluding the one from the pair in question. This method is referred to as **RegressionCos**. Figure 1 shows method accuracy across all test categories. PairDistance method produced inferior results on all categories even in comparison with 3CosAdd, but RegressionCos yields significant gains over it in 5 categories out of 9, although in 1 category (formation of opposites, such as *clear : unclear*) its performance is slightly inferior to 3CosAdd. It remains to be investigated why certain categories work better.

Within semantic part of the test RegressionCos method outperforms 3CodAdd on 4 categories with significant gap and marginally on "currency" questions. While the overall pattern is that RegressionCos yielding better results on all categories, this test also shows that different categories respond differently to different methods, and that it is necessary to conduct a larger evaluation of linguistic properties that lead to such discrepancies.

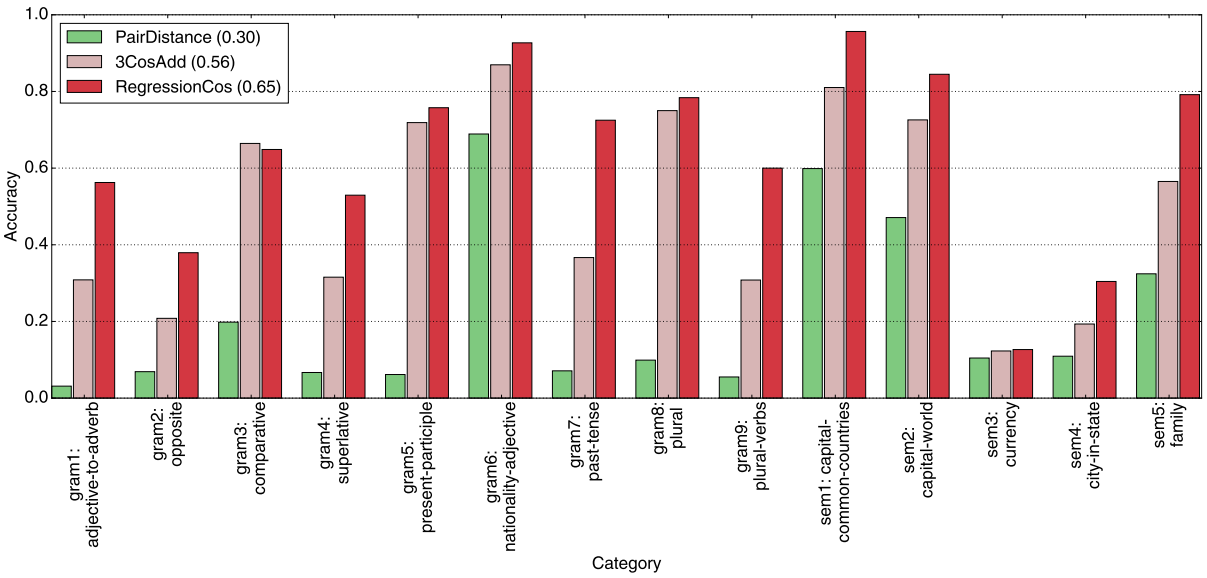

Figure 1: Accuracy comparison across categories.

6.1 Mitigating the effect of vector size

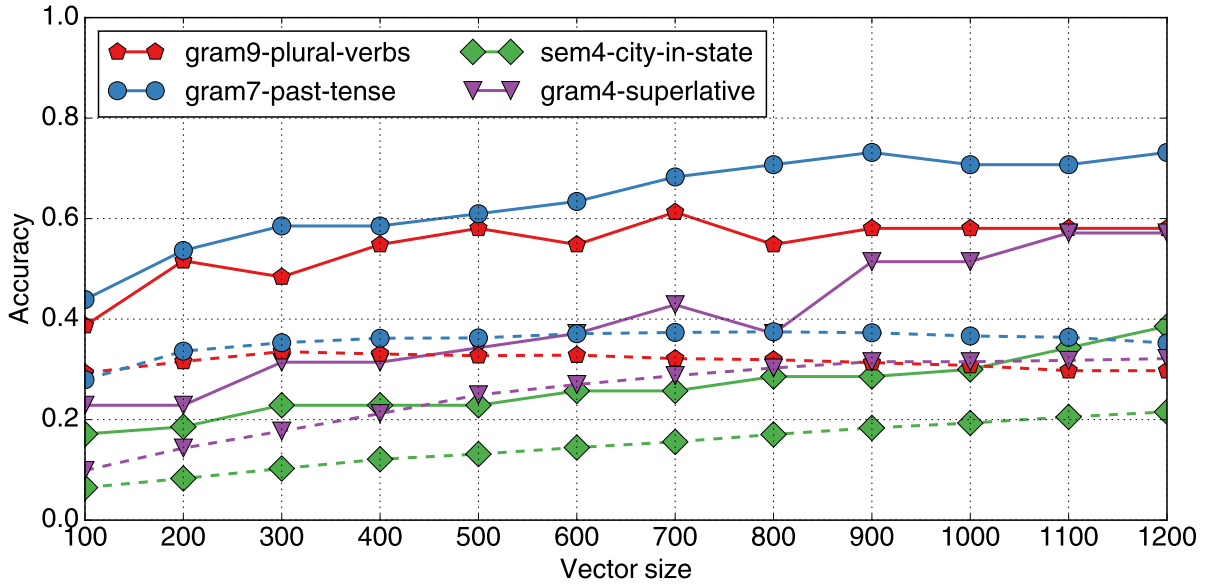

Figure 2: Effect of vector size. Solid lines represent RegressionCos method, dashed lines represent 3CosAdd method.

Vector embedding models come with multiple parameters which can affect the performance of certain linguistic tasks on them. For example, it has been reported that different sorts of tasks involving distributional semantics models benefit from different widow sizes [7]. Our findings indicate that vector size also has an effect on accuracy in detection of linguistic regularities.

Figure 2 shows that on certain categories higher dimensionality has negative effect on accuracy, as measured with the 3CosAdd method. One possible explanation is that once the dimensions corresponding to the core aspects of a particular analogical relation are included in the vectors, adding more dimensions increases noise. However, with the proposed method this effect is mitigated by the fact that regression can assign near-zero weights to the dimension which are not responsible for the target analogical relation and thus algorithm performance continues to grow with larger vector sizes.

7 Conclusion

This work proposed a method for discovering linguistic regularities, based on learning the target relation from a group of examples. Our tests show improved accuracy on most categories in Google test set, with up to 20-30% gain on 8 out of 14 categories in the set.

While the proposed method showed better performance than state-of-the art 3CosAdd approach, there is still much room for improvement. One interesting observation is that learning from multiple examples helps to determine which dimensions are relevant for capturing certain linguistic relations and as the result the performance of the method continues to improve with the increase of vector size.

References

[1] T. Mikolov, W. tau Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*. Association for Computational Linguistics, May 2013. [Online]. Available: http://research.microsoft.com/apps/pubs/default.aspx?id=189726

[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013. [Online]. Available: http://arxiv.org/abs/1301.3781

[3] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," vol. 12, pp. 1532–1543, 2014. [Online]. Available: http://llcao.net/cu-deeplearning15/presentation/nn-pres.pdf

[4] P. D. Turney, "A uniform approach to analogies, synonyms, antonyms, and associations," in *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, 2008, pp. 905–912.

[5] Z. Harris, "Distributional structure," *Word*, vol. 10, no. 23, pp. 146–162, 1954.

[6] K. Erk, "What is word meaning, really? (and how can distributional models help us describe it?)," in *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, ser. GEMS '10. Stroudsburg, PA, USA: Association for Computational Linguistics, 2010, pp. 17–26.

[7] J. A. Bullinaria and J. P. Levy, "Extracting semantic representations from word co-occurrence statistics: stop-lists, stemming, and SVD," *Behavior Research Methods*, vol. 44, no. 3, pp. 890–907, 2012.

[8] K. Lund and C. Burgess, "Producing high-dimensional semantic spaces from lexical co-occurrence," *Behavior Research Methods, Instruments, & Computers*, vol. 28, no. 2, pp. 203–208, 1996.

[9] K. W. Church and P. Hanks, "Word association norms, mutual information, and lexicography," *Comput. Linguist.*, vol. 16, no. 1, pp. 22–29, Mar 1990.

[10] V. Benko, "Aranea: Yet another family of (comparable) web corpora," in *Text, speech, and dialogue: 17th international conference, TSD 2014, Brno, Czech Republic, September 8-12, 2014. Proceedings*, ser. LNCS 8655, P. Sojka, A. Horák, I. Kopeček, and K. Pala, Eds. Springer, 2014, pp. 257–264.

[11] T. K. Landauer and S. T. Dutnais, "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge," *Psychological review*, pp. 211–240, 1997.

[12] R. Rapp, "Word sense discovery based on sense descriptor dissimilarity," in *Proceedings of the Ninth Machine Translation Summit*, New Orleans, LA., 2003, pp. 315–322.

[13] R. Lebret and R. Collobert, "Word embeddings through Hellinger PCA," in *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden: Association for Computational Linguistics, April 2014, pp. 482–490.

[14] D. Achlioptas and F. McSherry, "Fast computation of low rank matrix approximations," in *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, ser. STOC '01. New York, NY, USA: ACM, 2001, pp. 611–618.

[15] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[16] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of EMNLP*, 2014.

[17] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in Neural Information Processing Systems*, 2014, pp. 2177–2185.

[18] ——, "Linguistic regularities in sparse and explicit word representations," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. Ann Arbor, Michigan: Association for Computational Linguistics, June 2014, pp. 171–180.

[19] A. Drozd, A. Gladkova, and S. Matsuoka, "Python, performance, and natural language processing," in *Proceedings of the 5th Workshop on Python for High-Performance and Scientific Computing*, ser. PyHPC '15. New York, NY, USA: ACM, 2015, pp. 1:1–1:10. [Online]. Available: http://doi.acm.org/10.1145/2835857.2835858

[20] M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta, "The wacky wide web: a collection of very large linguistically processed web-crawled corpora," *Language Resources and Evaluation*, vol. 43, no. 3, pp. 209–226, 2009. [Online]. Available: http://dx.doi.org/10.1007/s10579-009-9081-4