Word Embeddings, Analogies, and Machine Learning: Beyond $King$ - Man + $Woman$ = $Queen$

Aleksandr Drozd[†], **Anna Gladkova**[‡], **Satoshi Matsuoka**[†]
† Tokyo Institute of Technology, Meguro-ku, Tokyo 152-8550, Japan
alex@smg.is.titech.ac.jp, matsu@is.titech.ac.jp
‡ The University of Tokyo, Meguro-ku, Tokyo 153-8902 Japan
gladkova@phiz.c.u-tokyo.ac.jp

Abstract

Solving word analogies became one of the most popular benchmarks for word embeddings on the assumption that linear relations between word pairs (such as $king$:man :: $woman$:$queen$) are indicative of the quality of the embedding. We question this assumption by showing that the information not detected by linear offset may still be recoverable by a more sophisticated search method, and thus is actually encoded in the embedding.

The general problem with linear offset is its sensitivity to the idiosyncrasies of individual words. We show that simple averaging over multiple word pairs improves over the state-of-the-art. A further improvement in accuracy (up to 30% for some embeddings and relations) is achieved by combining cosine similarity with an estimation of the extent to which a candidate answer belongs to the correct word class. In addition to this practical contribution, this work highlights the problem of the interaction between word embeddings and analogy retrieval algorithms, and its implications for the evaluation of word embeddings and the use of analogies in extrinsic tasks.

1 Introduction

Discovering analogical relations is currently one of the most popular benchmarks for word embeddings. This trend started after (Mikolov et al., 2013b) showed that proportional analogies (a is to b as c is to d) can be solved by finding the vector closest to the hypothetical vector calculated as c - a + b (e.g. $king$ - man + $woman$ = $queen$). Many subsequent studies used this approach to evaluate the performance of word embeddings with the Google test set (Mikolov et al., 2013a); the top current result is over 80% accuracy (Pennington et al., 2014). The assumption is that a "good" word embedding encodes linguistic relations in such a way that they are identifiable via linear vector offset (see section 2).

Analogies are interesting not only as a benchmark, but also potentially as a method for discovering linguistic relations (Turney, 2008). They are already used for morphological analysis (Lavallée and Langlais, 2010), word sense disambiguation (Federici et al., 1997), semantic search (Cohen et al., 2015), and even for broad-range detection of both morphological and semantic features (Lepage and Goh, 2009). However, Mikolov's study was a demonstration of how word embeddings capture linguistic relations, rather than a proposal of linear vector offset as a method for their discovery. It was later shown to not work as well for a wider range of relations (Köper et al., 2015; Gladkova et al., 2016).

This study questions the underlying assumption that linguistic relations should translate to linear relations between vectors rather than a more complex correspondence pattern. We show that relations not detected by vector offset may be recoverable by other methods, and thus are actually encoded in the embedding. The method we propose is based on learning the target relation from multiple word pairs, since reliance on single word pair makes linear vector offset sensitive to word idiosyncrasies. A naive average-based baseline outperforms the state-of-the-art. A more sophisticated machine-learning algorithm achieves further improvement (up to 30% for some embeddings and linguistic relations) by combining similarity to a source word vector ($king$) with the estimate of whether a candidate answer ($queen$) belongs to the correct class of words ("woman").

This work is licensed under a Creative Commons Attribution 4.0 International Licence. Licence details: http://creativecommons.org/licenses/by/4.0/

2 State of the Art: Analogical Reasoning Based on the Offset of Word Vectors

The starting point for this study is (Mikolov et al., 2013a), the first work to demonstrate the possibility of capturing relations between words as the offset of their vectors. The answer to the question "a is to b as c is to ?" is represented by hidden vector d, calculated as $argmax_{d \in V}(sim(d, c - a + b))$. Here V is the vocabulary (excluding word vectors a, b and c), and sim is a similarity measure, for which Mikolov and most other researchers use angular distance between vectors u and v: $sim(u, v) = cos(u, v) = \frac{u \cdot v}{||u||||v||}$ We will refer to this method as **3CosAdd**. The intuition behind it is that the position of, e.g., vector man relative to $king$ should be roughly the same as the position of $woman$ relative to $queen$. Vylomova et al. (2016) use this method as a basis for learning lexical relations with spectral clustering and Support Vector Machines (SVM).

An alternative method was introduced by Levy and Goldberg (2014) who propose to calculate the hidden vector as $argmax_{d \in V}(cos(d - c, b - a))$. They report that this method produces more accurate results for some categories. Its essence is that it accounts for $d - c$ and $b - a$ to share the same direction and discards lengths of these vectors. We will refer to this method as **PairDistance**.

Linzen (2016) reports results of experiments with 6 more functions, including reversing the relation, returning simply the nearest neighbour of the c word, and the word most similar to both b and c. None of these functions outperformed 3CosAdd and PairDistance consistently. Reversal was beneficial for some relations, but it is only applicable to symmetrical one-on-one relations. Crucially, when the words a, b and c are not excluded from the set of possible candidates, the performance drops to zeroes, and for the singular-plural noun category the correct answers are obtained with 70% accuracy as simply the nearest neighbours of the c word.

3 The Alternative: Learning From Multiple Examples

3.1 Naive Approach

The vector offset approach relies on a single pair of words, which makes it sensitive to noise and word idiosyncrasies, such as differences in polysemy networks. Consider the above $king$:$queen$ example: depending on the corpus, there may be more differences in their vectors than just masculinity/femininity. $Queen$ is also a musical group, and therefore appears in many contexts in which $king$ does not appear.

The alternative is to learn the relation from a set of example pairs. The "naive" baseline would be a simple average of the offset between every pair of vectors in the training set: $argmax_{d \in V}(sim(d, c + avg\_offset))$, where $avg\_offset = \frac{\sum_{i=0}^{m} a_i}{m} - \frac{\sum_{i=0}^{n} b_i}{n}$ and a_i and b_i represents words from source and target classes. We refer to this method as **3CosAvg**. To the best of our knowledge, this has not been explored before - surprising as it is.

3.2 LRCos Method

We propose an alternative approach to discovering linguistic relations with analogies based on a set of word pairs that have the same relation, such as the country:capital relation shown in Table 1:

Source	Target
France	Paris
Japan	Tokyo
China	Beijing

Table 1: Example analogy pairs set: capitals

In this set the right-hand-side and left-hand-side elements represent coherent groups of words - in this example, "countries" and "capitals". We shall refer to the left-hand-side of such analogies as the "source class", and to the right-hand-side - as "target class". Given a set of such word pairs, the question "what is related to $France$ as $Tokyo$ is related to $Japan$?" can be reformulated as "what word belongs to the same class as $Tokyo$ and is the closest to $France$?"

We detect words belonging to the target class (e.g. "capitals") with logistic regression[1]. Given a set of word pairs (e.g. *Japan:Tokyo*), the available target words are used as positive samples, and source words, along with random words from the dictionary, as negative samples. The number of random words and other parameters of logistic regression such as regularization strength can affect the performance of the classifier, but in our pilot tests no set of parameters yielded significant gains over the default choices. In experiments reported below the number of random words was equal to the number of positive samples. We used logistic regression implementation from Python `linear_model.LogisticRegression` module from Python's sklearn module version 0.17.1 with default parameters[2].

The probability of a word being the correct answer for a given analogy is calculated by combining (in this study, multiplying) the probability of this word belonging to the target class, and its similarity with the vector a measured using angular distance. Theoretically this enables further optimization through different weighting schemes, although our test did not show significant gains over simple multiplication.

Both set-based methods (3CosAvg and LRCos) were evaluated in exclude-n scheme. Given a set of 50 word pairs, n of them are excluded, and remaining are used for obtaining the "rule" of transfer (this part differs by the method). Then each of the n pairs become the question, and the learned "rule" is used to try to derive the answer. Larger n speeds up the computation and can be used for larger datasets, while n=1 will maximize the number of training elements to obtain the "rule". In this study we used n=2.

3.3 Filtering Vector Dimensions

Performance of LRCos could theoretically benefit from forcing the similarity metric to ignore irrelevant features. Consider the task of identifying plural forms of nouns (e.g. *phone:phones*, *banana:bananas*). The two linguistic classes (in this case singular and plural nouns) necessarily introduce some dissimilarity between $phone$ and $phones$.

Assuming that this dissimilarity is shared by all word pairs, we can learn which features are responsible for it, and exclude them in the similarity estimation step. This should give an advantage to words from the target class. Ideally, when the "plurality" features are excluded, the $phones$ vector should be the most similar to the $phone$ vector. To implement this method we have additionally trained C-Support Vector Classifier (sklearn.svm.SVC) with a linear kernel to discriminate between "left" and "right" words and used complimentary values of the weights assigned to the features it learned to scale individual dimensions. We will refer to this "filtered" variant of LRCos method as LRCosF.

4 Corpora and Word Embeddings

Word embeddings represent words in the vocabulary as vectors that can be derived directly from co-occurrence counts ("explicit models") or learned implicitly by neural nets (see (Erk, 2012) for general overview of the field). It is currently debated whether explicit and implicit models are conceptually different (Levy and Goldberg, 2014), or whether the latter have an advantage over the former (Baroni et al., 2014). To contribute to the ongoing debate, this work explores both types of models.

The source corpus combines an English Wikipedia snapshot from July 2015 (1.8B tokens), Araneum Anglicum Maius (1.2B) (Benko, 2014) and ukWaC (2B) (Baroni et al., 2009) (uncased, words occurring less than 100 times were discarded). The resulting vocabulary size is 301,949 words.

The **SVD**-based explicit model is built upon co-occurrence matrix weighted by Positive Pointwise Mutual Information (PPMI, Church and Hanks (1990)). The co-occurrence matrix was computed using the co-occurrence extraction kernel by (Drozd et al., 2015) with a window size of 8. Singular Value Decomposition (SVD) transformation was used to obtain low-rank approximation of the sparse co-occurrence matrix. SVD factorizes $m \times n$ real or complex matrix M in a form $M = U\Sigma V^*$ (Golub and Van Loan, 1996), and embeddings can be obtained as $U\Sigma$. Σ is a diagonal matrix the elements of which reflect how much of a variance of original data is captured in a given dimension. We used the technique by (Caron, 2001) of rising Σ matrix element-wise to the power of a where $0 < a \leq 1$ to give a boost to dimensions

[1] We tried several other classification algorithms such as SVM with linear, polynomial and radial basis function kernels, but neither of them yielded higher classification accuracy, and they also were more computationally expensive. Building a classifier directly from the set of vector offsets of all word pairs was also not successful.

[2] Source code and additional materials are available at `http://vsm.blackbird.pw`

with smaller variance, with a =0.1 for 300-dimensional embeddings and a=0.6 for the rest. We have used embeddings of size 300 and 1000 for comparison with GloVe and Skip-Gram models, and sizes 100-1200 for studying the dimensionality effect. Finally, we have normalized each embedding vector individually, as we have found that it increases the performance of SVD-based embeddings.

As representatives of implicit models we used GloVe and Skip-Gram. The **GloVe** model was trained with the original software by (Pennington et al., 2014) with 300 dimensions, window size 8, 20 iterations, parameters x_{max}= 100, a = 3/4. **Skip-Gram** embeddings were trained with original software by Mikolov (Mikolov et al., 2013a) in skip-gram mode, with windows size 8, 25 negative samples, 5 iterations, "sample" parameter (for down-sampling of frequent words) equal to 1e-4. It is also worth noting that co-occurrences for the SVD model were collected with respect to sentence boundaries, while GloVe and Skip-Gram models disregard them.

The performance of word embeddings can be drastically affected by their parameters (Levy et al., 2015; Lai et al., 2015), which prompts parameter searching for different tasks. However, accuracy of solving word analogies also varies immensely for different linguistic relations (Gladkova et al., 2016). Optimizing for "average accuracy" on a diverse set of relations may not be meaningful, as it does not necessarily guarantee better performance on a particular relation. Therefore we did not attempt such parameter search for our models. However, in section 5.1 we will test our embeddings on the widely used Google analogy test to show that they are generally on the par with the previously reported results (Levy et al., 2015; Pennington et al., 2014), and not victims of some particularly unfortunate configuration.

5 Evaluation

5.1 The Google Test Set

3CosAdd is widely used for benchmarking word embeddings on the test known as the Google test set (Mikolov et al., 2013a). It contains 14 categories with 20-70 unique example pairs per category, which are combined in all possible ways to yield 8,869 semantic and 10,675 syntactic questions. The state-of-the-art on this test has over 65% average accuracy: 67.8% for DRRS (Garten et al., 2015), 70.64% for GCeWE (Zhou et al., 2015), and 75.6% for GloVe (Pennington et al., 2014).

The average accuracy for 3 models with 4 analogy detection methods is presented in table 2. We used logistic regression as a classification algorithm in the "exclude one" scheme, where the classifier is re-trained each time on all target class words excluding the one from the pair in question. Table 2 shows that LRCos clearly outperforms 3CosAdd and 3CosAvg, although for all methods accuracy varies between relations and models.

We compute both Mean_{all} (the number of correct answers divided by the total number of questions in the whole dataset) and Mean_{rel} (the average accuracy scores for all categories), and, for the latter, also SD (standard deviation) between categories. It is Mean_{all} that is typically reported (Mikolov et al., 2013a; Pennington et al., 2014), but table 2 suggests that Mean_{all} tends to be higher than Mean_{rel}. We attribute this to the fact that the Google test set is not balanced (20-70 unique pairs per category), and the more successful country:capital relations constitute the bulk of the semantic questions. Mean_{all} also can not represent the variation between categories, which in our experiments is between 17-28%.

Method	3CosAdd			PairDistance			3CosAvg			LRCos		
	Mean_{all}	Mean_{rel}	SD	Mean_{all}	Mean_{rel}	SD	Mean_{all}	Mean_{rel}	SD	Mean_{all}	Mean_{rel}	SD
SVD300	50.6%	45.1%	24%	22.7%	16.1%	17%	54.8%	51.2%	26%	68.2%	68.1%	23%
SVD1000	58.1%	49.4%	25%	23.6%	22.3%	17%	59.7%	54.0%	27%	74.6%	72.6%	21%
GloVe	**79.6%**	67.8%	26%	**33.5%**	**26.9%**	22%	79.1%	74.2%	28%	73.7%	70.9%	24%
Skip-Gram	75.1%	**66.6%**	23%	28.6%	24.2%	21.6%	**80.3%**	**78.3%**	17%	**79.8%**	**78.0%**	17%

Table 2: Average accuracy in the total dataset (Mean_{all}), between 14 categories (Mean_{rel}), and the standard deviation (SD) between categories in the Google test set.

Table 2 highlights the interaction between the method of discovering analogies and the word embedding itself. The results of GloVe and Skip-Gram improve with LRCos as compared to 3CosAdd, but the simple average 3CosAvg works even slightly better for them. However, SVD gets an over 15% boost

from LRCos, but not from 3CosAvg. This suggests that (a) the information not detected by 3CosAdd was actually contained in the SVD model, and (b) evaluating different embeddings with the same method might not be as meaningful as is commonly believed. Perhaps a better question to ask is why these embeddings behave so differently, and what does it tell us about them and the methods used.

5.2 The Bigger Analogy Test Set

The above results suggest that performance on the Google test set varies significantly between categories, and it was shown that some relations are in principle not detected successfully with 3CosAdd (Köper et al., 2015). Gladkova et al. (2016) developed the BATS analogy test set that is balanced across 4 types of linguistic relations (grammatical inflections, word-formation, lexicographical and world-knowledge relations), with 50 word pairs per category and 10 categories of each type (overall 2000 unique word pairs)[3]. This test presents a bigger challenge: the performance of GloVe is reported to drop from 80.4% on the Google test set to 28.5% on BATS due to difficulties with derivational and lexicographic categories.

Table 3 and figure 1 show that LRCos follows this overall trend, achieving only 47.7% average accuracy on BATS with Skip-Gram, the best-performing embedding. But it still outperforms the others: the best average for 3CosAdd is 28.1% (Glove), 7.5% for PairDistance (GloVe), 34.4% for 3CosAvg (GloVe). LRCosF is only slightly behind (47.2% for LRCosF on Skip-Gram).

Compared to 3CosAdd LRCos achieves up to 25% boost on encyclopedic relations (SVD model with 1000 dimensions), up to 8% boost on lexicographic relations (SVD), and, most significantly, up to 34% boost on the difficult derivational relations (for Skip-Gram). For inflectional morphology LRCosF yielded even better results (up to 28% for SVD).

Method	Encyclopedia			Lexicography			Inflectional Morphology			Derivational Morphology		
	SVD	GloVe	Skip-Gram	SVD	GloVe	Skip-Gram	SVD	GloVe	Skip-Gram	SVD	GloVe	Skip-Gram
PairDistance	11.8%	13.6%	12.4%	1.1%	1.0%	0.8%	12.8%	14.5%	14.9%	1.9%	0.8%	0.8%
3CosAdd	18.5%	31.5%	26.5%	10.1%	10.9%	9.1%	44.0%	59.9%	61.0%	9.8%	10.2%	11.2%
3CosAvg	30.0%	**44.8**%	34.6%	12.2%	13.0%	9.6%	51.2%	68.8%	69.8	13.0%	11.2%	15.2%
LRCos	39.3%	40.6%	**43.6**%	**18.0**%	16.8%	**15.4**%	65.2%	74.6%	87.2%	**30.4**%	17.0%	**45.6**%
LRCosF	**43.7**%	40.8%	42.6%	16.4%	**17.6**%	14.4%	**72.2**%	**75.0**%	**87.4**%	30.0%	**17.1**%	44.2%

Table 3: Average accuracy per relation type in BATS per method for SVD1000, GloVe and w2v models.

Figure 1 demonstrates variation in performance of 3CosAdd, 3CosAvg and LRCos on GloVe and SVD models by individual categories of BATS. LRCos almost always performs the best, but the pattern of results for GloVe and SVD is a little different. SVD did worse on inflectional morphology on SVD than on GloVe, so it benefitted more from LRCos - but it is interesting that (a) the benefit for the overall better-performing GloVe was overall smaller, and (b) LRCos almost never improves the results for categories where 3CosAdd already achieved near 80% accuracy. This suggests that there might be a certain limit on how accurate we can get on the test, at least for a given corpus.

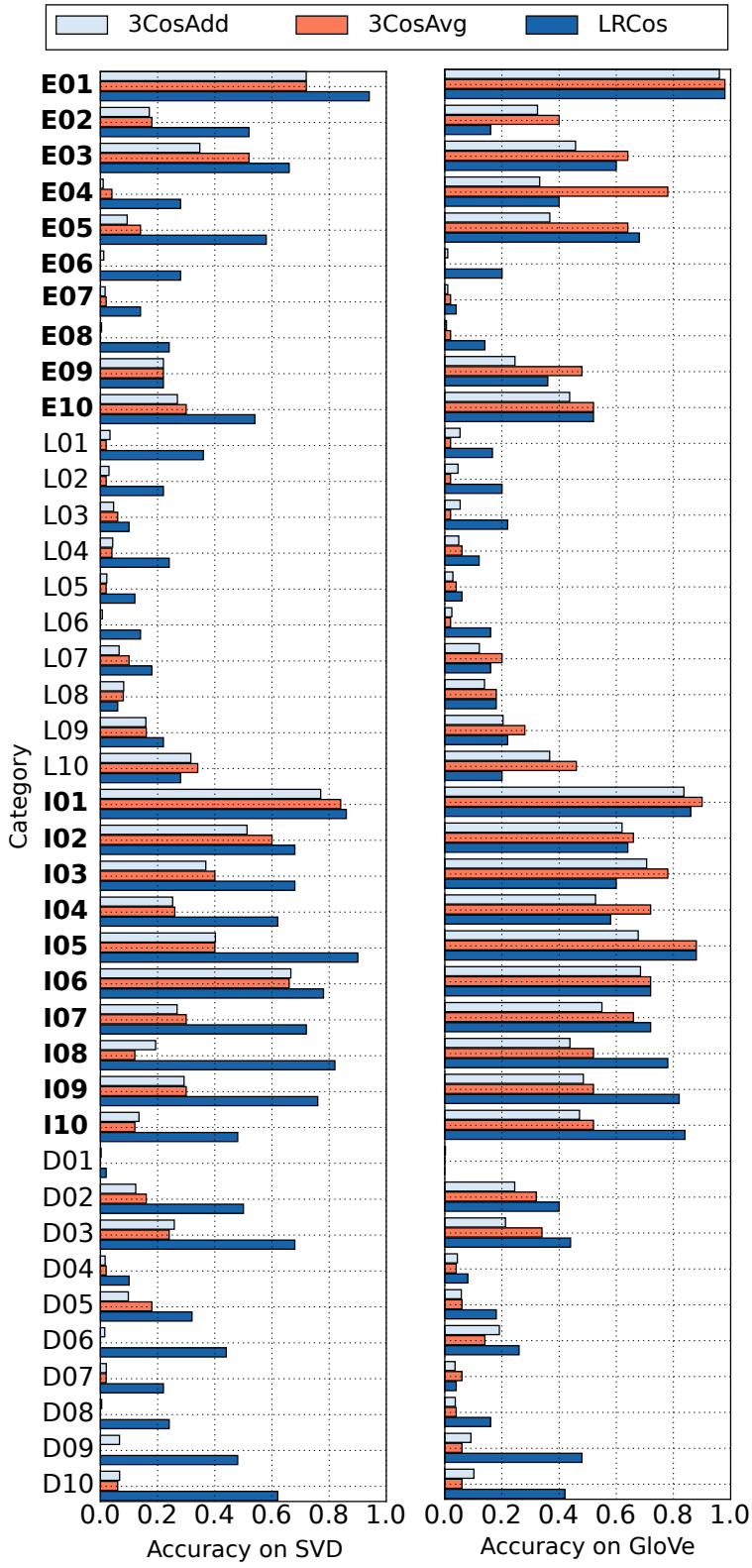
Table 3 shows that different methods for discovering analogies do not perform uniformly across the whole set or different embeddings. LRCos and LRCosF are "preferred" by different types of relations (although the gap between them is not so large), and in one case the baseline actually performs better.

One of the possible explanations for why LRCos yields significant improvement for derivational morphology, but not for lexicographic relations, is that LRCos relies on the notion of "target word class". In case of suffixes and prefixes such a target class is relatively coherent ("all words with the suffix *-ness*"), but for, e.g., synonyms, BATS includes different parts of speech (e.g. *scream:cry*, *car:automobile*, *loyal:faithful*). In this case there is no clear target class, and LRCos should actually be at a disadvantage compared to 3CosAdd (although it still improves results for some of the more coherent categories).

5.3 Russian Morphological Categories

As an additional task we compiled a small set consisting of 6 Russian noun forms: nominative case paired with instrumental, dative and prepositional cases in singular and plural form, such as *yaponets : yapontsem* ("a Japanese": "by a Japanese"). As in BATS, each category contains 50 unique word pairs.

[3]BATS dataset can be downloaded from `http://vsm.blackbird.pw/bats`



Encyclopedic relations

E01: geography: capitals (*Athens:Greece*)
E02: geography: languages (Peru:*Spanish*)
E03: geography: UK counties (*York:Yorkshire*)
E04: people: nationality (*Lincoln:American*)
E05: people: occupation (*Lincoln:president*)
E06: animals: the young (*cat:kitten*)
E07: animals: sounds (*dog:bark*)
E08: animals: shelter (*fox:den*)
E09: thing:color (*blood:red*)
E10: male:female (*actor:actress*)

Lexicographic relations

L01: hypernyms: animals (*turtle:reptile*)
L02: hypernyms: miscellaneous (peach:*fruit*)
L03: hyponyms: miscellaneous (*color:white*)
L04: meronyms: substance (*sea:water*)
L05: meronyms: member (*player:team*)
L06: meronyms: part-whole (*car:engine*)
L07: synonyms: intensity (*cry:scream*)
L08: synonyms: exact (*sofa:couch*)
L09: antonyms: gradable (*clean:dirty*)
L10: antonyms: opposites (*up:down*)

Inflectional Morphology

I01: noun sg:pl (regular) (*student:students*)
I02: noun sg:pl (irregular) (*wife:wives*)
I03: adjective: comparative (*strong:stronger*)
I04: adjective: superlative (*strong:strongest*)
I05: infinitive: 3Ps.Sg (*follow:follows*)
I06: infinitive: participle (*follow:following*)
I07: infinitive: past (*follow:followed*)
I08: participle: 3Ps.Sg (*following:follows*)
I09: participle: past (*following:followed*)
I10: 3Ps.Sg : past (*follows:followed*)

Derivational Morphology

D01: noun+ness (home:*homeness*)
D02: un+adjective (*able:unable*)
D03: adjective+ly (*usual:usually*)
D04: over+adjective (*used:overused*)
D05: adjective+ness (*mad:madness*)
D06: re+verb (*create:recreate*)
D07: verb+able (*edit:editable*)
D08: verb+er (*bake:baker*)
D09: verb+tion (*continue:continuation*)
D10: verb+ment (*argue:argument*)

Figure 1: Performance of 3CosAdd, 3CosAvg and LRCos methods on BATS categories.

The overall accuracy on the SVD embedding with 1000 dimensions is **18.0%** with 3CosAdd method, **19.2%** with 3CosAvg and **43.1%** with LRCos. For GloVe the results are **28.1%**, **34.4%** and **39.4%**, respectively.

While this is not a comprehensive test like BATS, it is sufficient to see if the different methods of discovering analogical relations are equally successful on morphologically complex and simple languages. The problem with the former is that there are many more word forms per lemma (e.g., the English text of *Pride and Prejudice* contains 7266 tokens for 6576 lemmas, and its Russian translation – 17903 tokens for 7715 lemmas, i.e. almost three times more). For word-level word embeddings this means that there are more vectors from the same volume of text, that they are built with less information, and that there are more vectors that are very similar (which complicates the task for a method relying on cosine similarity).

For this test we built an SVD-based model from Araneum Russicum Maximum (Benko and Zakharov, 2016) - a web-corpus containing 13.4B tokens. The parameters are as follows: 1000 dimensions, window size 3, with PMI-weighted co-occurrence matrix and Σ raised to the power $a = 1$.

Fig. 2 shows that morphological complexity does increase difficulty for analogical reasoning with word embeddings. In English 3CosAdd scores over 50% on many morphological categories, but in Russian cases its performance is in the 20% range. LRCos performs significantly better, although not always on the par with English. Further research is needed to tell why, e.g., Russian prepositional case appears to be more difficult than instrumental.

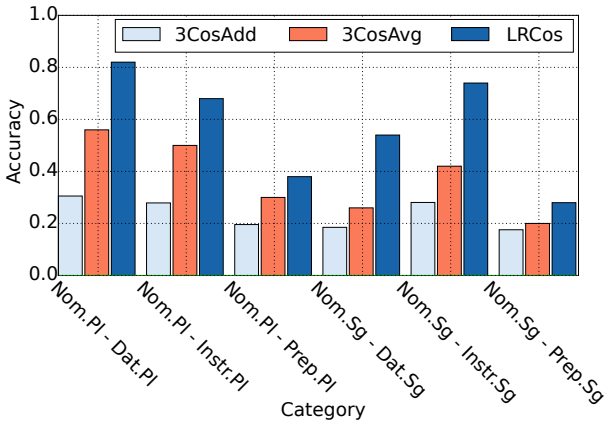


Figure 2: Accuracy of LRCos, 3CosAdd and 3CosAvg on Russian noun case forms.

6 Exploring LRCos

6.1 Effect of Training Set Size

Any method relying on supervised learning is only useful when there is sufficient data for learning. To use a method such as LRCos for practical analogical reasoning tasks we need to know how much data we would have to provide for each relation.

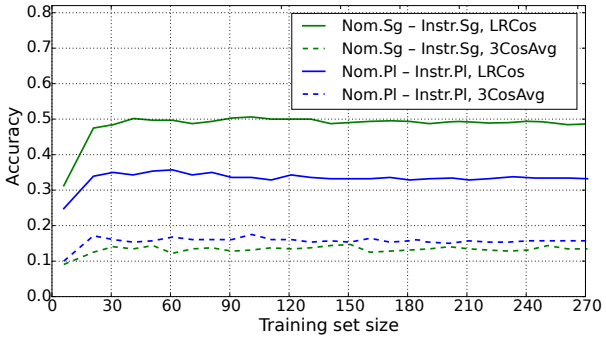


Figure 3: Effect of training set size.

We performed such an evaluation with our Russian morphology data, creating sets that contained up to a thousand word pairs. To estimate the optimal number of training samples we repeated the experiment multiple times, each time randomly selecting a subset of word pairs and observing how its size affects performance.

Two sample categories are shown in Figure 3 (LRCos and 3CosAvg methods). Our experiments suggest that accuracy for Russian morphological categories for both methods saturates at 50 pairs on average. While more tests are needed to determine if this number may be different for other types of relations or for other languages, such a range is feasible for LRCos to be used in practical tasks such as morphological parsing.

6.2 Effect of Vector Size

Our experiments suggest that although higher dimensionality implies more information about words being captured, it does not necessarily leads to better accuracy with the 3CosAdd method (a similar effect was observed by Cai et al. (2015) for similarity task). Some categories benefit slightly from higher dimensions, but for many there is no improvement, or there is even a slight degradation, while for

3CosAdd performance continues to rise. Data for four example categories are shown in Figure 4. One possible explanation for this phenomenon 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.

LRCos is not completely free from this negative effect, but it suffers less as the effect is mitigated by the fact that regression can assign near-zero weights to the dimensions which are not responsible for the target analogical relation. Thus algorithm performance continues to grow with larger vector sizes.

This result suggests that there is potential for LRCos to achieve even better results with combined models (Garten et al., 2015; Fried and Duh, 2014). For example, different window sizes are believed to be more beneficial for different tasks: larger windows for topical relations, smaller windows for morphological relations, as shown in (Lebret and Collobert, 2015). This would prevent any one model from achieving top performance on all tasks. However, we can have, e.g., a model that combines window 10 and window 2, and the extra dimensions will not become noise for LRCos method.

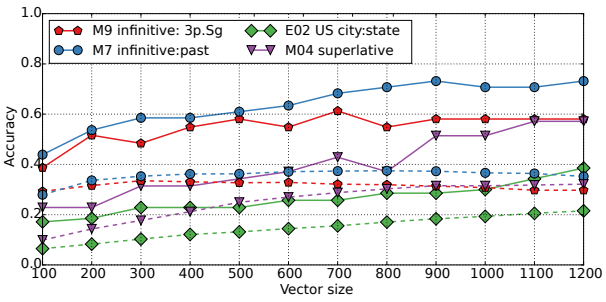


Figure 4: Effect of vector dimensionality on 3CosAdd (dashed lines) and LRCos (solid lines) methods

6.3 Effect of a Parameter

As described in section 4, we raise the elements of Σ matrix of factorization to the power of a to control the contribution of different singular vectors. If a is equal to 1, then each column in the transformed matrix is scaled proportionally to the variance in the original data it represents. Smaller values of a essentially boost the features which were less pronounced in the corpus in terms of co-occurrence patterns.

Figure 5 illustrates the impact of changing a for several example relations. Similarly to other model parameters, there is no value that provides the best results for all cases. The average accuracy is affected slightly, but certain relations may experience up to a two times decrease or improvement. This suggests that treating individual dimensions of embeddings independently could yield better results. While experimenting with the size of embeddings suggests that the LRCos method is better at selectively choosing useful dimensions for the embeddings, there is still a lot of room for improvement.

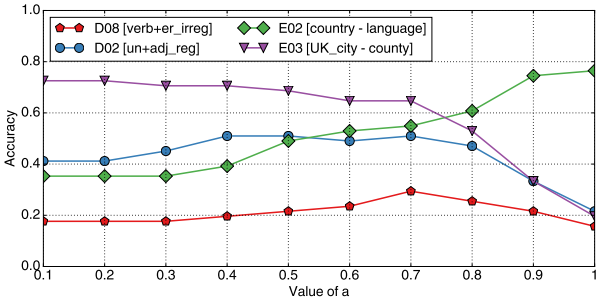


Figure 5: Effect of changing the value of α

7 Further Ways to Improve: Mistakes of 3CosAdd and LRCos

Since LRCos relies on both cosine similarity and degree to which the hypothetical answer belongs to the target class of words, it could be expected to yield a different pattern of mispredictions than 3CosAdd. To investigate the differences in the output of the two methods we manually annotated the incorrect answers by 3CosAdd and LRCos on 4 BATS categories: E05 (*Lincoln:president*), L05 (*parishioner:parish*), M05 (*follow:follows*) and WF05 (*create:recreate*). The evaluation was done with the SVD model at 1000 dimensions, window size 3. The results of this evaluation are summarized in table 4.

First of all, for both methods the ratio of "random" answers is insignificant; most of the wrong answers are morphologically, derivationally, collocationally, or semantically related to one or more of the source words – as can be expected for methods relying on cosine similarity. The problem, traditionally, is distinguishing between different types of relations.

When proposing the 3CosAdd method, Mikolov et al. (2013b) exclude the three source words from

Type of answer	Example	E05 name:occupation		L05 member: group		I05 infinitive: 3p.Sg		D05 re+verb	
		3CosAdd	LRCos	3CosAdd	LRCos	3CosAdd	LRCos	3CosAdd	LRCos
Correct answer	*hear: hears :: seem: seems*	10.62	52.00	0.97	4.08	35.1	78.00	26.24	48.00
Acceptable answer	*plato:philosopher :: hegel:?theorist*	1.42	6.00	1.75	6.12	-	-	-	-
Morphological relation	*define:redefine :: imagine: *imagining*	3.84	-	44.28	56.00	43.39	-	23.19	-
Misspelling of a source word	*confirm:reconfirm :: acquire:*aquire*	-	-	-	2.04	0.08	-	2.20	2.00
Derivational relation	*hear: hears :: seem: *seemingly*	0.94	-	1.25	-	0.82	-	1.55	2.00
Lexicographical relation	*include: includes:: appear: *seems*	61.6	22.00	27.69	4.08	4.53	4.00	14.04	8.00
Frame-semantic relation	*sit: resit :: learn: *coursework*	15.03	14.00	20.13	24.49	9.63	10.00	13.11	36.00
Collocate of a source word	*protect: protects:: learn: *basics*	-	-	1.99	4.08	3.35	2.00	0.49	-
Mistake due to polysemy	*parishioner: parish :: relative: *density*	1.67	-	8.49	4.08	0.94	-	-	-
Partially correct answer	*protect: protects :: maintain: *ensures*	-	-	0.97	10.20	6.82	18.00	13.71	28.00
Unrelated word	*send: resend :: engage: *categorise*	2.37	4.00	2.63	-	1.39	4.00	8.28	2.00

* Several relations may be applicable to each case, so the sum for each column does not necessarily add up to 100%.

Table 4: Types of mistakes for 3CosAdd and LRCos methods in different linguistic relations.

the set of possible answers, because otherwise one of them is too likely to turn up to be the closest to the hypothetical vector. But even if they are excluded, these source vectors can still "misdirect" the method. The fact that in L05, I05, and D05 the most frequent type of mistakes are the wrong morphological forms of a source word is consistent with the finding of Linzen (2016) that in the noun plural category of the Google test set the nearest neighbor of the c word provides the correct answer in 70% of cases. The plurals-as-nearest-neighbors were the pitfall for our L05 *member:group* category, where the analogy *student:class :: bird:?(flock)* would yield the answer *birds*. Likewise, with the verbs in I05 and D05 we were getting many participles with *-ing* ending: *arrange:rearrange::grow:*growing* (expected: *regrow*), *create:creates::accept:*accepting* (expected: *accepts*).

Consider now the E05 category that seems to break the pattern of morphologically-related nearest-neighbor: here the most mistakes are "lexicographic'. E05 category has analogies such as *aristotle:philosopher::bach:?composer*. The typical mistake is a co-hyponym of the a or (usually) c word, i.e. another composer in this case. This is also explained by the fact that for the names of famous people their nearest neighbors frequently happen to be co-hyponyms: in our SVD model the nearest neighbors of *Bach* are *Haydn* and *Schubert*, and in GloVe - *Handel* and *Beethoven*. This means that in all the categories the basic source of mistakes is the same indiscriminateness of cosine similarity.

Unfortunately, this means that word analogies fail to provide sufficient "context" to words: ideally, *king:queen :: man:woman* and *king:kings :: queen:queens* should profile sufficiently different aspects of the $king$ vector to avoid the nearest-neighbor trap. However, it does not seem to work this way. This is particularly clear in mistakes resulting from polysemy of one of the source words. For example, in L05 we had: *crow:murder::fish:*killing* (expected: *school*), *lion:pride::bird:*ambition* (expected: *flock*).

In E05, D05 and I05 LRCos significantly improves over 3CosAdd by reducing this nearest-neighbor kind of mistake, but it is telling that this improvement comes with the increase of partially-correct answers: the model comes up with the correct target feature in an incorrect word, e.g. ask $asks$ + $happen$ = $realizes$ (expected: *$happens$). Such mistakes suggest that the contribution of classifier and cosine similarity could differ for different words, although it is not clear how to determine them dynamically.

Another observation from our data is that for both methods the margin of error is very thin. For example, in the E05 category LRCos gives *Depp:screenwriter* a total score of 0.36, and this incorrect answer beats the correct answer *Depp:actor* that is ranked 0.35. Average accuracy would be much higher if we allowed the answers to be in the top five nearest neighbors, although this, of course, brings up the problem of where analogies could be used in practice, and what level of precision that would require.

8 Discussion: Embeddings vs Methods

LRCos offers a significant boost in accuracy for detecting analogical relations over the most widely-used 3CosAdd method, including derivational relations where the latter does not perform well. However, LRCos is by no means perfect, and there is room for further improvement, especially with respect to lexicographic relations. This includes algorithms aimed at searching for complex patterns of correspondences between vectors rather than simple similarity. A different (and potentially more fruitful) approach is to investigate whether the target relations are at all reflected in the distributional properties of words.

Aside from the practical result for non-lexicographic relations, this work also brings up a theoretical question. We have shown that different methods of detecting analogies provide different results on different embeddings, and this means that low performance of a word embedding with, e.g., 3CosAdd method, does not prove that the embedding does not capture certain linguistic relations - only that they are not detectable with this particular method. This brings into question the validity of analogy detection with 3CosAdd as a benchmark for word embeddings, as it is frequently used (Pennington et al., 2014; Garten et al., 2015; Cui et al., 2014).

It could be argued that embeddings could be judged "good" as in "easy to work with"; in this sense a "good" embedding is an embedding that yields correct answers with simple rather than complex extraction methods. However, what is good for practical applications is not necessarily the same as what is good for a benchmark. In this case with analogies, 3CosAdd is at disadvantage with embeddings that encode a lot of extra (but useful) information in dimensions that are irrelevant to a particular relation, and thus misleads 3CosAdd. On the other hand, it could be argued that machine-learning-based methods should not be used for benchmarking because they could learn to ignore noise too well.

9 Conclusion

We presented LRCos, a method of analogical reasoning that is based on supervised learning from a group of examples. LRCos significantly outperforms the popular 3CosAdd method (based on offset for individual word pairs) on both the Google and BATS test sets, although the gain varies between embeddings and relation types. Importantly, LRCos achieves high accuracy in two areas where 3CosAdd mostly failed: word-formation in English and grammar in Russian, a morphologically rich language. Unlike 3CosAdd, LRCos is less sensitive to idiosyncrasies of individual word pairs, and does not suffer from higher vector dimensionality.

We compared 5 analogical reasoning methods on 40 types of linguistic relations with two word embeddings: explicit SVD model and neural-net-based GloVe. Both models yielded overall similar patterns of performance with different methods, offering further evidence for conceptual similarity of explicit and implicit word embeddings.

This work also makes a theoretical contribution in demonstrating the interaction between word embeddings, types of analogies, and different types of search algorithms: with LRCos our SVD-based model approaches the state-of-the-art performance for GloVe and Skip-Gram. This suggests that the information about linguistic relations from the test set was actually encoded in the SVD-based embedding, possibly in a different way. In that case we need to decide whether failure to detect a relation with 3CosAdd method actually indicates inferiority of a word embedding, and whether a "good" embedding should encode different kinds of relations in the same way - as the Google test set in conjunction with 3CosAdd is still one of the most popular benchmarks.

Acknowledgements

This paper was partially supported by JST, CREST (Research Area: Advanced Core Technologies for Big Data Integration).

References

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Marco Baroni, Georgiana Dinu, and Germn Kruszewski. 2014. Dont count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 238–247.

Vladimr Benko and V.P. Zakharov. 2016. Very large Russian corpora: New opportunities and new challenges. In *Kompjuternaja Lingvistika I Intellektuanyje Technologii: Po Materialam Medunarodnoj konferencii "Dialog" (2016)*, volume 15(22), pages 79–93. Moskva: Rossijskij gosudarstvennyj gumanitarnyj universitet.

Vladimír Benko. 2014. Aranea: Yet another family of (comparable) web corpora. In Petr Sojka, Aleš Horák, Ivan Kopeček, and Karel Pala, editors, *Text, speech, and dialogue: 17th international conference, TSD 2014, Brno, Czech Republic, September 8-12, 2014. Proceedings*, LNCS 8655, pages 257–264. Springer.

Yuanyuan Cai, Wei Lu, Xiaoping Che, and Kailun Shi. 2015. Differential Evolutionary Algorithm Based on Multiple Vector Metrics for Semantic Similarity Assessment in Continuous Vector Space. In *Proceedings of DMS 2015*, pages 241–249. [doi:10.18293/DMS2015-001].

John Caron. 2001. Computational information retrieval. chapter Experiments with LSA Scoring: Optimal Rank and Basis, pages 157–169. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29, Mar.

Trevor Cohen, Dominic Widdows, and Thomas Rindflesch. 2015. Expansion-by-analogy: A vector symbolic approach to semantic search. In *Quantum Interaction*, pages 54–66. Springer.

Qing Cui, Bin Gao, Jiang Bian, Siyu Qiu, and Tie-Yan Liu. 2014. Learning effective word embedding using morphological word similarity. *arXiv preprint arXiv:1407.1687*.

Aleksandr Drozd, Anna Gladkova, and Satoshi Matsuoka. 2015. Python, performance, and natural language processing. In *Proceedings of the 5th Workshop on Python for High-Performance and Scientific Computing*, PyHPC '15, pages 1:1–1:10, New York, NY, USA. ACM.

Katrin Erk. 2012. Vector space models of word meaning and phrase meaning: A survey. *Language and Linguistics Compass*, 6(10):635–653.

Stefano Federici, Simonetta Montemagni, and Vito Pirrelli. 1997. Inferring semantic similarity from distributional evidence: an analogy-based approach to word sense disambiguation. In *Proceedings of the ACL/EACL Workshop on Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, pages 90–97.

Daniel Fried and Kevin Duh. 2014. Incorporating both distributional and relational semantics in word representations. *arXiv preprint arXiv:1412.4369*.

Justin Garten, Kenji Sagae, Volkan Ustun, and Morteza Dehghani. 2015. Combining distributed vector representations for words. In *Proceedings of NAACL-HLT 2015*, pages 95–101.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proceedings of NAACL-HLT 2016*, pages 47–54. Association for Computational Linguistics.

Gene H. Golub and Charles F. Van Loan. 1996. *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, Baltimore, MD, USA.

Maximilian Köper, Christian Scheible, and Sabine Schulte im Walde. 2015. Multilingual reliability and semantic structure of continuous word spaces. In *Proceedings of the 11th International Conference on Computational Semantics 2015*, pages 40–45. Association for Computational Linguistics.

Siwei Lai, Kang Liu, Liheng Xu, and Jun Zhao. 2015. How to generate a good word embedding? *arXiv preprint arXiv:1507.05523*.

Jean-Franois Lavallée and Philippe Langlais. 2010. Unsupervised morphological analysis by formal analogy. In *Multilingual Information Access Evaluation I. Text Retrieval Experiments*, pages 617–624. Springer.

Rmi Lebret and Ronan Collobert. 2015. Rehabilitation of count-based models for word vector representations. In *Computational Linguistics and Intelligent Text Processing*, pages 417–429. Springer.

Yves Lepage and Chooi-ling Goh. 2009. Towards automatic acquisition of linguistic features. In *Proceedings of the 17th Nordic Conference on Computational Linguistics (NODALIDA 2009), eds., Kristiina Jokinen and Eckard Bick*, pages 118–125.

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 171–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. In *Transactions of the Association for Computational Linguistics*, volume 3, pages 211–225.

Tal Linzen. 2016. Issues in evaluating semantic spaces using word analogies. In *Proceedings of the 1st Workshop on Evaluating Vector Space Representations for NLP*, pages 13–18. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *Proceedings of International Conference on Learning Representations (ICLR)*.

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. 2013b. 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.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: global vectors for word representation. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, volume 12, pages 1532–1543.

Peter D. Turney. 2008. A uniform approach to analogies, synonyms, antonyms, and associations. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 905–912.

Ekaterina Vylomova, Laura Rimmel, Trevor Cohn, and Timothy Baldwin. 2016. Take and took, gaggle and goose, book and read: evaluating the utility of vector differences for lexical relation learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1671–1682. Association for Computational Linguistics.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2015. Category enhanced word embedding. *arXiv preprint arXiv:1511.08629*.