# Limits of Deep Learning

Simon Jenni
University of Bern
simujenni@students.unibe.ch

Simon Brulhart
University of Fribourg
simon.brulhart@unifr.ch

***Abstract*—Current natural language models try to capture semantic and syntactic relationships between words. This allows them to answer questions in the form of analogies: "X is to Y as U is to ?". We study the performance of the word2vec tool on this task. To this end, we analyse the sensitivity to different hyperparameter choices. We generate new question sets and demonstrate the models (in)ability to learn and answer on domain-specific knowledge. To offer a more detailed view of the model's performance, we introduce a broader definition of accuracy which is taking into account how far off the model is in a failure case and study how the choice of similarity measure influence the model's accuracy. To increase the models accuracy on solving analogies, we propose a mean-shift procedure as preliminary step to computing the query vector and a combination of analogy interpretations for solving for the most similar word-vector. The effectiveness of these approaches are demonstrated on a generalised question set.**

## I. Introduction

Recent state-of-the-art natural language models represent words by embedding them as vectors in a continuous vector space [1] [2]. These embeddings are learned as weights of a recurrent neural network language model. It has been demonstrated that the distributed representation of words as vectors in a common vector-space captures not just syntactic similarities (*e.g.* $cat$ is close to $cats$) but semantic similarities as well. Concretely, the model's ability to perform "word-arithmetic" has been shown in [3]. To exemplify this, let $f$ be the word-embedding function. Ideally, we would then have an embedding where distances in vector-space give a measure of semantic-similarity between the words. For two word pairs such as $(king : man)$ and $(queen : woman)$ which have the same relation (or similarity), we would obtain $f(king) - f(man) \approx f(queen) - f(woman)$ or equivalently $f(king) - f(man) + f(woman) \approx f(queen)$. This of course is equivalent to the analogy "$king$ is to $man$ as $queen$ is to $woman$".

The goal of this work is to further evaluate the performance of the popular word2vec tool on these tasks. We evaluate the effect of different choices for model-architectures and hyperparameters on the performance and test the limits of the models semantic-analysis capabilities on new challenging test data. The ability to learn analogies has been demonstrated on rather general and hand-picked test-sets. We test the models ability to learn domain-specific knowledge by generating questions that test relationships such as $Brand \sim Country$ and $Movie \sim Director$. Previous work focuses on the nearest resulting vector in the embedding space when evaluating the accuracy of this word-arithmetic. We generalise the notion of accuracy by considering not only the nearest word-vector, but consider the set of the $k$ nearest vectors and test whether the target word is present therein.

The content of this paper is structured as follows: Section II gives an overview of relevant previous works. In Section III we present a brief overview of the underlying model and in Section IV we outline the experimental setup and testdata used. The results of the experiments are then reported in Section V before we conclude in VI-C.

## II. Related Work

The work by Tomas Mikolov [1], where the models were introduced, reports results on a test set they themselves created. The set contained five types of semantic questions where relations such as $Country \sim Currency$ or $Country \sim Capital$ were tested. In [3] the models performance on semantical similarity has been demonstrated on SemEval-2012 Task 2 [4]. The objective in this task is to decide how similar two target wordpairs such as $(glass : break)$ and $(soldier : fight)$ are with respect to a relation "an $X$ typically $Y$". Though the model is not directly trained on this task, it achieved better results than the previous best methods.

In [5] Levy *et. al.* provide a comprehensive study of several word embedding techniques on the two tasks *Word Similarity* and *Analogy*. They conclude that much of the performance gains of new embedding methods stem from design choices and parameter optimisation rather than the underlying algorithms. Performance estimation was based only on the testdata sets introduced in [1] and [3].

In a previous work [6] they show that generalising the Skip-Gram model's negative sampling to arbitrary contexts results in the model learning functional similarities. They demonstrate this by manually inspecting the top $k$ most similar words to a given query word. A query *e.g.* for *Hogwarts* returns a list of famous schools in their modified Skip-Gram implementation while CBOW returns a list of *Harry Potter* characters.

## III. The Models of word2vec

The word2vec tool is an implementation of the Continuous Bag-of-Words (CBOW) and Skip-Gram models introduced in [1]. Both of these models are based on two-layer neural networks with architectures depicted in Figure 1 and Figure 2. The two architectures reflect the objectives of both models: CBOW is trained to predict the centre word from its context in a sentence while Skip-Gram tries to predict the context from its centre. Both architectures take vectors of one-hot encoded

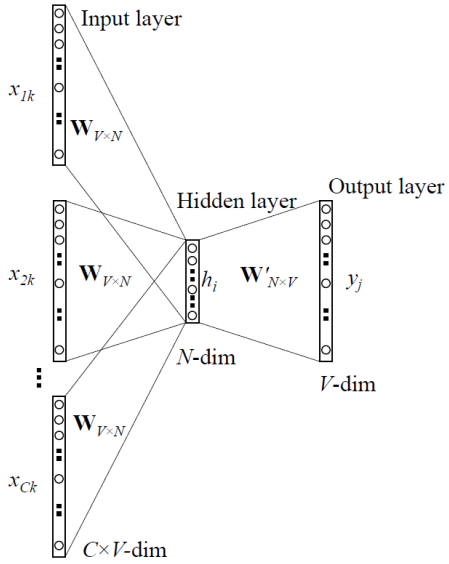



Fig. 1: Architecture of the CBOW model. The three one-hot input vectors represent the context of a centre word in a sentence and the output vector is a vector with probabilities over possible centre-words.

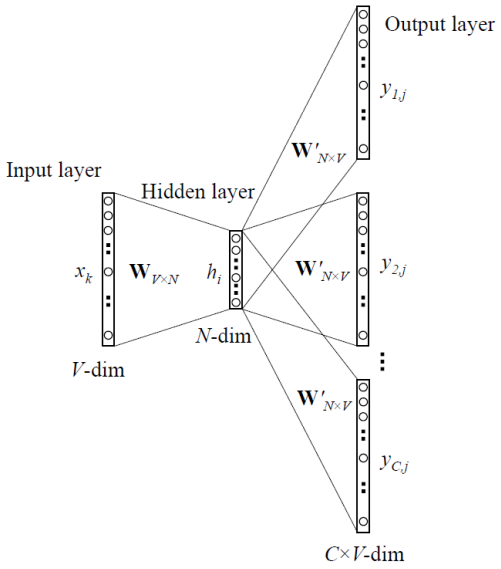



Fig. 2: Architecture of the Skip-Gram model. The output of this model are vectors with probabilities for the words occurring in the context of the input word.

words as inputs. Let $x_i$ denote an input-vector corresponding to word $i$ and let the input vocabulary be $V$, then the inputs to the model are given by the $|V|$-dimensional vectors:

$$w_a = \begin{pmatrix}1\\0\\0\\\vdots\\0\end{pmatrix}, w_{abandon} = \begin{pmatrix}0\\1\\0\\\vdots\\0\end{pmatrix}, \ldots, w_{zone} = \begin{pmatrix}0\\0\\\vdots\\0\\1\end{pmatrix} \tag{1}$$

These one-hot vectors get mapped into a $N$-dimensional vector via the $|V|\times N$ weight matrix $\boldsymbol{W}$. The hidden unit then accumulates the sum of these vectors for all input words in the case of CBOW and consists of only the vector associated with the single input in the Skip-Gram model. The output of the model is generated by multiplying the hidden-units by with a $N \times |V|$ weight matrix $\boldsymbol{W'}$ and applying softmax to the result in order to generate a valid probability distribution over words in the vocabulary. Concretely, given a sequence of training words $w_1, w_2, \ldots, w_T$ the hidden-states are computed as

$$h = \boldsymbol{W}w. \tag{2}$$

Given the values of the hidden units, the output can be computed as

$$y = P_t(\boldsymbol{W'}h) \tag{3}$$

with the softmax function $P_t$ of a $N$-dimensional vector $x$ given by

$$P_t(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{|V|} e^{x_j}}, \tag{4}$$

for $i \in \{1, \ldots, |V|\}$. The learnt word representations are then given by the columns of $\boldsymbol{W}$.

The models are learnt in an unsupervised fashion where either the centre-word of a series of words is predicted from its context (CBOW) or the context is predicted from the centre word (Skip-Gram). Both models try to maximise the log-likelihood of their output. Training is done via gradient descent using backpropagation [7].

At test time when looking for the word most similar to a given word $w$ we solve for

$$x = \text{argmax}_{v\in V} d(f(w), f(v)) \tag{5}$$

Where $f$ is the embedding and $d$ is a similarity measure. We therefore search for the word in the vocabulary which has the minimal distance in the embedding space. The model as proposed by [1] chose $d$ to be the cosine similarity, but we investigate different choices such as $L^p$-norm or Dice similarity.

### A. Word Arithmetic for Solving Analogies

Consider the case of an analogy question "$X$ is to $Y$ as $U$ is to $V$" where we are interested in solving for the word $V$. Let $x = f(X)$, $y = f(Y)$ and $u = f(U)$ be the embedded words. The straightforward way of computing $v = f(V)$ is then to use Equation 5 and solving for

$$v = \underset{v\in f(V)}{\text{argmax}}\ d(v, u - x + y) \tag{6}$$

As it turns out however depending on the similarity measure $d(\cdot|\cdot)$ it can be a good idea to consider different approaches such as

$$v = \underset{v\in f(V)}{\text{argmax}}\ d(v, u) - d(v, x) + d(v, y) \tag{7}$$

The intuition behind this approach is less one of performing word-vector arithmetic and more one of finding the word $V$ most similar to $X$ and $U$ and dissimilar to $Y$. This approach turns out to work considerably better when using the cosine-similarity. Note that Equations 6 and 7 are equivalent for unit vectors when using the cosine similarity.

### B. Considered Similarity Measures

In our experiments we considered the following similarity measures:

$$L^2(x,y) = -||x-y||_2 = -\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{8}$$

$$L^1(x,y) = -\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2} \tag{9}$$

$$Dice(x,y) = \frac{xTy}{||x||+||y||} \tag{10}$$

$$Cos(x,y) = \frac{x^Ty}{||x||\cdot||y||} \tag{11}$$

We denote the approach in Equation 7 with cosine similarity by $cosAdd$ in the results.

### C. Mean-Shift for Better Discrimination

Solving for the closest word in Equation 5 based on the cosine similarity can be interpreted as projecting on an $N$-dimensional unit sphere and looking for the projected vector with smallest angle to the given query vector. This projection can lead to similar vectors being very close on the sphere and therefore hard to distinguish based on angular distance. This observation is also backed up by the plots shown Figure 7 where we observe that the correct word is often very close by. Being able to better distribute out the points on the sphere would therefore likely improve the predictions.

We therefore propose the following mean-shift procedure:

1) Normalise all vectors (*i.e.* project on the unit sphere)
2) Compute the mean of the normalised vectors
3) Shift by mean
4) Normalise again (*i.e.* re-project on unit sphere)
5) Compute query vector with respect to the new normalised word vectors
6) Compute dot-products with query vector to get cosine-similarities

The intuition behind this procedure is depicted in Figure 3 which demonstrates how points very close by get distributed out. Note that the word arithmetic to compute the query vector is performed after the normalisation (otherwise no change in the results would be expected). We denote this approach with $CosMS$ in our experiments.

### D. Combining Analogy Interpretations

We have already touched upon the idea that there are different ways of interpreting analogy exercises with respect to some similarity measure in Subsection III-A. The idea for Equation 7 for example is that we are looking for the vector $v$ most similar to $x$ and $u$ but dissimilar to $y$. We could however also pose it as looking for

$$v = \underset{v\in f(V)}{\operatorname{argmin}} |d(x,y) - d(u,v)|, \tag{12}$$

that is $x$ and $y$ should have the same similarity as $u$ and $v$. Yet another option is to consider

$$v = \underset{v\in f(V)}{\operatorname{argmin}} |d(x,u) - d(y,v)|. \tag{13}$$

Note that these objectives all capture different aspects of the analogy, *i.e.* they are not implying each other. We propose to combine these objectives in a weighted sum of these terms. Let $\phi$ denote the term in Equation 7, *i.e.*

$$\phi(v) = d(v,u) - d(v,x) + d(v,y). \tag{14}$$

We then want to maximise the following expression:

$$\phi(v) - \lambda\cdot(|d(x,y)-d(u,v)| + |d(x,u)-d(y,v)|) \tag{15}$$

We set the weight parameter $\lambda = 0.1$ in our experiments and denote models using this similarity measure (including mean-shift) with $CosMS+$.

## IV. Experiments

In this section we first outline our experimental setup, introduce novel test data and then report results obtained for different parameter choices and test scenarios.

### A. Setup

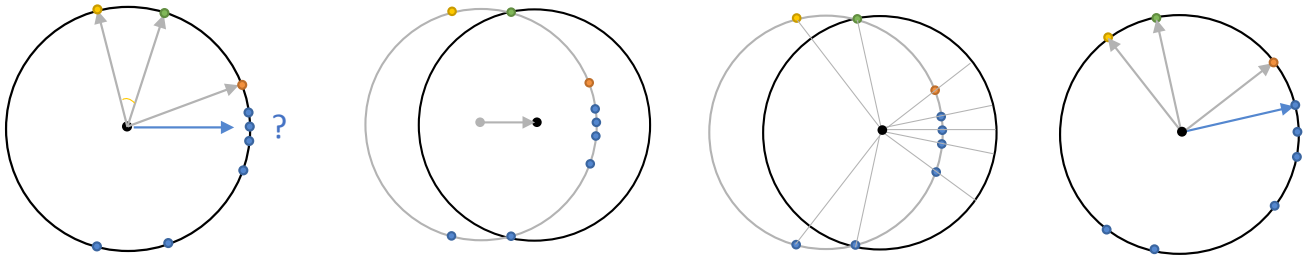
We used the Python re-implementation of word2vec contained in the Gensim library [8]. This implementation is functionally identical to the original C implementation provided by [1], but is easier to extend and has shown to perform just as well.

To compare the influence of parameters on the performance we train several models with different values for one of the parameters and keep all other parameters fixed. We identified the following set of parameters for our experiments:

1) Context-window size
2) Vector size (*i.e.* dimension of hidden-layer)
3) Learning rate $\alpha$
4) Training corpus
5) Hierarchical softmax vs. negative sampling for Skip-Gram

### B. Training Text Corpora

We trained models using multiple text corpora to check if hyperparameters influence performance differently depending on the corpus. These corpora include:




(a) Nearby points in the right are difficult to discriminate

(b) Shifting by the mean.

(c) Reprojecting on the unit sphere.

(d) Better discrimination of points

Fig. 3: Sketch of the proposed mean-shift procedure in 2D. Consider the analogy "*Yellow* is to *Green* as *Orange* is to ?". *a)*: Unit sphere with projected vectors in respective colours. The points on the right are relatively close together and difficult to distinguish. *b)*: The mean of all the points is computed and the centre of the sphere shifted. *c)*: The re-projections of the points on the shifted sphere. *d)*: The points are now farther apart and better distinguishable.

*1) Wiki:* The first 900 megabytes of the English Wikipedia dump on 03/03/2006. [1]

*2) Text8:* The first 100 megabytes of the English Wikipedia dump. Facilities for downloading and using this set are provided with word2vec. [2]

*3) Brown:* English corpus of one million words assembled from 500 sources of different genres, created in 1961 at Brown University. It is considerably smaller than our other corpora. We used this set and the sets below through the NLTK python library. [3]

*4) Treebank:* Fragment of the *Penn Treebank* created in 1995 at the University of Pennsylvania. This contains material from the Wall Street Journal of 1989.

*5) Movie_reviews:* Corpus of movie reviews up to 2004 from the website Rotten Tomatoes.

### C. New Test Data

To test the models performance on domain-specific questions we constructed several questions sets. The questions are all of the form "X is to Y as U is to ?", *i.e.* they take the form of analogies. We constructed the following sets of questions:

*1) Brand-Origin:* A set of questions testing for the relationship between well-known brands and their countries of origin. To construct this questions set we gathered names and country of origin for the top 500 brands according to brandfinance.com. We excluded all the brands from countries with names composed of multiple words (*e.g.* United States of America, ...) due to the model working on single words. The result is a set of 38,220 questions.

*2) Movie-Director:* A set of analogy-questions for the relationship between famous movies and their respective directors. The set was constructed by gathering movie titles and last name of the director for some of the top rated movies according to IMDb. Similar to the Brand-Origin questions we excluded movies with titles composed of more than one word with the exception of titles starting with "The ..." or similar. The result is a set of 4,557 questions.

[1]Instructions to download and build the test data on this website: http://mattmahoney.net/dc/textdata.html

[2]Ready-to-use corpus: http://mattmahoney.net/dc/text8.zip

[3]See http://www.nltk.org/book/ch02.html

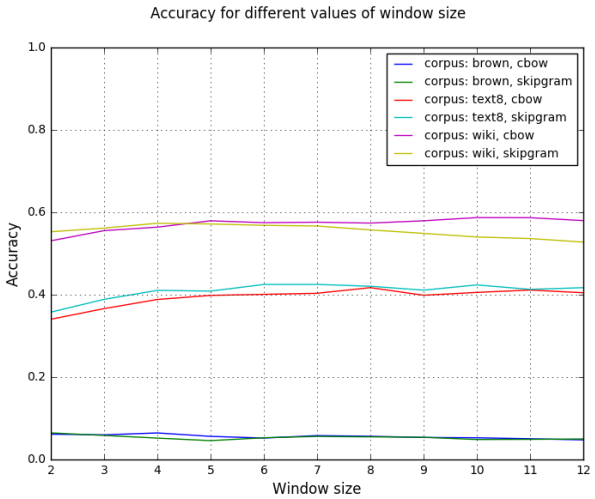



Fig. 4: Accuracy for different values of the context-window size.

## V. Results

### A. Influence of Parameters on Model Performance

We first evaluate the influence of parameter choices using Google's generalised analogy dataset described in [1]. For this purpose we plotted the accuracy obtained for different values of a chosen parameter, while keeping all other parameters fixed. The baseline model for the comparison (with default parameter values) is shown in Table I . Here, the accuracy is defined as the ratio of the number of right answers over the total number of questions.

In Figure 4 we see that the window size has only a limited influence on the accuracy. It is also difficult to pinpoint a common best value but window sizes between 5 and 8 seem to

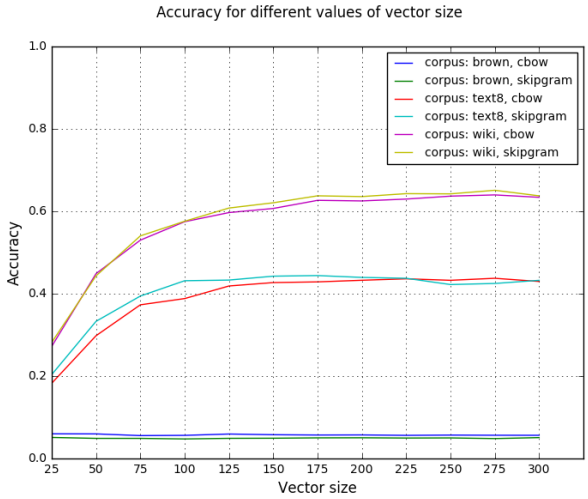



Fig. 5: Accuracy for different values of vector-size (*i.e.* dimension of embedding space).

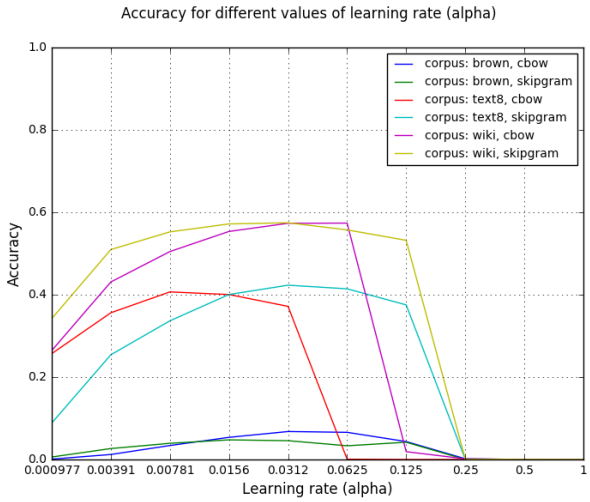



Fig. 6: Accuracy for different values of the learning rate $\alpha$.

work well on all corpora. Generally the model is quite robust with respect to this parameter. Interestingly, a bigger window size seems beneficial when using CBOW on the wiki corpus, but detrimental when using Skip-Gram.

Conversely, we see in Figure 5 that a large enough vector size is essential for good performance. A larger value generally improves accuracy, although we get diminishing returns above a certain point. This is not surprising as the vector size defines the complexity of the model and therefore also matters more for larger corpora. Obviously the vector size also has some

| Model-Architecture | Learning Rate $\alpha$ | Window-Size | Vector-Size |
|---|---|---|---|
| Skip-Gram | 0.025 | 5 | 100 |

TABLE I: The parameter values for the baseline model used in our experiments.

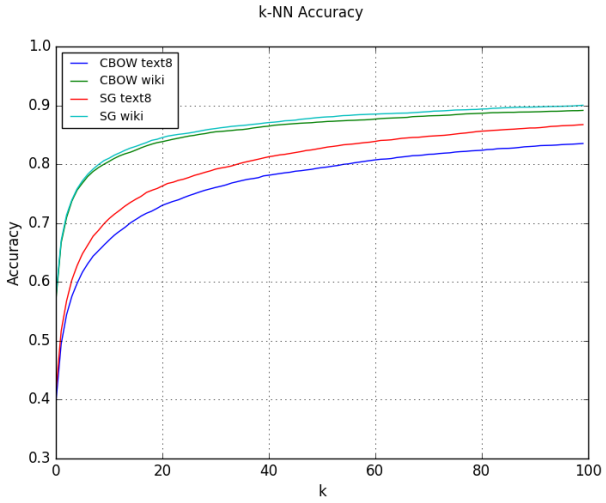



Fig. 7: k-NN accuracy according to k.

effect on training time, so it is wise to restrain the vector size.

The learning rate ($\alpha$) greatly affects performance as can be seen in Figure 6. There appears to be a critical spot for $\alpha$ around which the accuracy drops dramatically (values around 0.03 - 0.15). However, a learning rate around 0.007 - 0.03 yields good results for every model we tried.

From a broader perspective, we observe that the corpus size is by far what influences performance the most. This is not surprising given that neural networks are dependent on large amounts of training data. This is especially apparent when looking at the models trained on the $Brown$ corpus, which all fail to get reasonable accuracies.

Considering the choice of model architecture we also notice that Skip-Gram slightly outperforms CBOW in most cases. It should be noted however that the Skip-Gram model is slightly more complex to train.

### B. K-NN Accuracy

In cases when the model gives the wrong answer, the common definition of accuracy ignores how far off the model's prediction was from the correct answer. To get an idea of how wrong the model really is, we tried a different definition of accuracy where we consider the $k$ nearest neighbours of the prediction. We count the answer as a success if this set of neighbours contains the correct answer.

We plotted this k-NN accuracy for different values of $k$ in Figure 7. It is clearly observable that the accuracy increases dramatically when changing from $k = 1$ to 2 or 3 and only slowly starts saturating at around $k > 10$. This suggests that the prediction is in fact not far off most of the time and suggests that small modifications to the similarity computations (such as the mean-shift) might have a positive effect on the performance. We also notice how models trained on larger corpora saturate faster and that the difference in performance between CBOW and Skip-Gram is more apparent in models trained on small corpora.

| Method | Capital-Country | Currency-Country | City-State | Family | Adjective-Adverb | Opposite | Superlative | Nationality-Adjective | Past-tense | Plural | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $CosAdd$ | 47.4% | 21.3% | 22.2% | 63.7% | 15.1% | 15.4% | 40.1% | 77.8% | 32.9% | 48.3% | 43.0% |
| $Dice$ | 42.8% | 20.5% | 19.5% | 65.0% | 13.5% | 15.4% | 37.7% | **79.2%** | 32.1% | 44.8% | 41.1% |
| $L^2$ | 44.0% | 14.9% | 20.6% | 60.1% | 15.2% | 11.8% | 27.7% | 74.9% | 30.2% | 42.8% | 39.9% |
| $L^1$ | 39.5% | 17.5% | 17.4% | 56.9% | 12.8% | 8.8% | 25.7% | 72.8% | 24.8% | 36.2% | 36.1% |
| $CosMul$ | 43.4% | 20.5% | 20.6% | 56.5% | 13.2% | 11.8% | 39.7% | 73.5% | 28.6% | 39.9% | 39.5% |
| $CosMS$ | 47.8% | **21.6%** | 23.1% | 63.4% | 15.1% | 15.0% | 42.7% | 77.9% | 33.0% | **49.3%** | 43.3% |
| $CosMS+$ | **49.2%** | 21.3% | **24.1%** | **65.4%** | **15.5%** | **16.0%** | **43.9%** | 78.4% | **33.9%** | 47.7% | **44.0%** |

TABLE II: Comparison of different similarity-measures. Results obtained on the generalised question set of [1] with a Skip-Gram model trained on the *Text8* corpus. We can observe that cosine-similarity overall performs better compared to the other measures. Although not a similarity-measure by its own we list the results of our proposed mean-shift approach for comparison and note a slight increase in accuracy. Note that the total is computed as the micro-average (mean over all questions).

### C. Similarity Measures

To compare the different similarity measures introduced in Subsections III-B and III-C we show results on selected categories of the generalised analogy dataset [1] obtained with the baseline Skip-Gram model with parameters shown in Table I. The resulting accuracies are listed in Table II.

We observe that overall cosine similarities perform significantly better than the other similarity measures (with a few exceptions). Possible explanations for this circumstance are difficult to provide. For one we note that the Euclidean distance between weight-vectors are never explicitly used in the training of the model. In fact, the weight vectors only occur in dot-products with the columns of the hidden-to-out weight matrix $\boldsymbol{W'}$. Another explanation is that the norm of a word vector is related to the word-frequency as found by [9] and is therefore not directly helpful in the analogy task.

### D. Mean Shift and Combination of Analogy Interpretations

Tables V and VI show results comparing models with and without the proposed mean-shift and combination of analogy interpretations. Results for the models with mean-shift are indicated with *CosMS*. We observe a slight overall improvement of models with mean-shift compared to the standard model for all the models.

Models with the additional term of alternative analogy interpretations from Equation 15 are depicted with *CosMS+*. This has shown to further slightly improve the accuracy of the model.

### E. Domain Specific Questions

We evaluated how well word2vec models built from a large-scope corpus could answer domain specific questions. For this purpose we used the baseline model with parameters shown in Table I, using either the *wiki* or *text8* corpus. We then measured the mean-shift accuracy of these models against the Brand-Origin and Movie-Director questions sets.

We also tried updating the existing models by training them using corpora closer related to the questions domains. We re-trained the models using the *treebank* and *movie_reviews* corpora to answer Brand-Origin and Movie-Director questions, respectively.

Table III and IV show that the baseline large-scope model performs much worse on our domain-specific questions than on the generalised analogy dataset [1], in particular for movie related questions. An explanation for this result is that many movie titles are made of common words, cited in numerous different context. On the other hand, a movie's title and director may be mentioned too far from each other in the corpus to fit in the window size.

| Training iterations | wiki | | text8 | |
|---|---|---|---|---|
| | General questions | Specific questions | General questions | Specific questions |
| 0 | **57.80%** | **9.62%** | **43.28%** | **5.65%** |
| 1 | 50.72% | 9.51% | 36.97% | 5.55% |
| 2 | 40.43% | 9.44% | 31.84% | 5.53% |
| 3 | 37.11% | 9.33% | 28.50% | 5.42% |
| 10 | 30.21% | 9.39% | 22.45% | 5.45% |

TABLE III: Accuracy of models trained again with *treebank* corpus against Brand-Origin questions

| Training iterations | wiki | | text8 | |
|---|---|---|---|---|
| | General questions | Specific questions | General questions | Specific questions |
| 0 | **57.80%** | **2.32%** | **43.27%** | 1.18% |
| 1 | 32.96% | 1.52% | 24.75% | **2.26%** |
| 2 | 20.15% | 1.07% | 14.97% | 1.51% |
| 3 | 14.42% | 0.89% | 11.26% | 1.40% |
| 10 | 6.28% | 0.80% | 5.28% | 0.54% |

TABLE IV: Accuracy of models trained again with *movie_reviews* corpus against Movie-Director questions

Note that for both questions sets, the accuracy increases significantly with a larger model. This improvement is more important for specialized questions than for general ones.

Updating existing models with more specific corpora worsened their accuracy in most cases. Applying this training multiple times amplified the effect. It also harmed the models accuracy against the generalised analogy dataset. However the *treebank* and *movie_reviews* corpora are smaller than the original ones by orders of magnitude. Therefore we cannot generalise the counter-productivity of such a re-training based on our results.

## VI. Discussion

### A. Limitations

All of our experiments were based on small word2vec models, relatively to the size of models often used in practice. We can observe the importance of the model size on our

| Model | Capital Country | Currency Country | City State | Family | Adjective Adverb | Opposite | Superlative | Nationality Adjective | Past-tense | Plural | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CBOW | 63.6% | 19.1% | 42.8% | **79.7%** | 28.4% | 36.0% | 62.3% | 84.3% | 45.3% | 67.8% | 58.7% |
| CBOW $CosMS$ | 63.9% | 19.7% | 43.9% | **79.7%** | 28.4% | 35.7% | **63.4%** | 83.9% | 45.3% | 67.8% | 59.1% |
| CBOW $CosMS+$ | 64.9% | 20.2% | **44.5%** | 79.2% | 28.8% | 34.2% | **63.4%** | 83.7% | 44.4% | 67.8% | 59.1% |
| Skip-Gram | 70.4% | 21.3% | 34.6% | 70.5% | **29.2%** | **38.6%** | 58.7% | 89.4% | 49.2% | 69.3% | 59.2% |
| Skip-Gram $CosMS$ | 70.5% | 21.3% | 35.9% | 70.8% | 28.3% | **38.6%** | 58.7% | **89.5%** | **49.6%** | 70.0% | 59.4% |
| Skip-Gram $CosMS+$ | **71.6%** | **21.9%** | 37.4% | 73.7% | 28.0% | 37.7% | 60.1% | 88.9% | 48.6% | **70.1%** | **60.0%** |

TABLE V: Performance comparison of models with and without our proposed mean-shift approach. Results obtained on the generalised question set of [1] with models trained on the *Wiki* corpus.

| Model | Capital Country | Currency Country | City State | Family | Adjective Adverb | Opposite | Superlative | Nationality Adjective | Past-tense | Plural | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CBOW | 33.5% | 16.4% | 14.7% | **79.7%** | 13.9% | **20.3%** | 42.3% | 65.4% | 30.3% | 44.7% | 39.5% |
| CBOW $CosMS$ | 35.3% | 17.5% | 15.8% | 78.4% | 14.9% | 19.9% | 42.3% | 64.9% | 31.0% | 45.3% | 39.8% |
| CBOW $CosMS+$ | 35.5% | 16.4% | 16.3% | 79.4% | 14.4% | 19.0% | 44.1% | 64.9% | 30.3% | 46.1% | 40.0% |
| Skip-Gram | 47.4% | 21.3% | 22.2% | 63.7% | 15.1% | 15.4% | 40.1% | 77.8% | 32.9% | 48.3% | 43.0% |
| Skip-Gram+RC | 47.8% | **21.6%** | 23.1% | 63.4% | 15.1% | 15.0% | 42.7% | 77.9% | 33.0% | **49.3%** | 43.3% |
| Skip-Gram $CosMS+$ | **49.2%** | 21.3% | **24.1%** | 65.4% | **15.5%** | 16.0% | **43.9%** | **78.4%** | **33.9%** | 47.7% | **44.0%** |

TABLE VI: Performance comparison of models with and without our proposed mean-shift approach. Results obtained on the generalised question set of [1] with models trained on the *Text8* corpus.

results, suggesting this would also be the case with larger models. Likewise we used corpora of multiple kinds through this work, but the only large corpora (*text8* and *wiki*) are based on the same material.

Moreover, experimenting with two novel sets of questions seems insufficient to conclude on models performance on specific domains. We also observe how counter-productive re-training can be, but this could be anecdotal. This would be clearer if we tried updating the original model with a model of similar size. We could compare the accuracy of both models individually and mixed together. Then we would know if bad results are caused by the re-training itself, or simply one of the model performing significantly worse than the other.

Lastly, we could not find a convincing theoretical explanation for the performance difference between similarity measures. Understanding the cause of such a difference would help find better measures.

### B. Future Work

As explained in the previous section, it would be interesting to train and evaluate models on much larger corpora to confirm the tendencies we exposed. In particular, we need to check if improvements from mean-shift and the combination of interpretations translate to bigger models.

We want to work on ways to explicitly train models with the goal of querying for analogies and similarities in mind.

Finally we want to compare the performance of word2vec against other tools such as GloVe[4] and recurrent neural networks.

### C. Conclusion

In this work we studied the influence of several parameter choices on the performance of word2vec models. We found that the learning rate $\alpha$, the size of the hidden layer and the choice of architecture are critical while the window size has not shown to affect the performance significantly. Not surprisingly however, the size of the training corpus has shown to be the most important factor on the performance of the model. By analysing the k-NN accuracy we observed that the model's guesses are usually very close to the correct answer.

In a comparison of different similarity measures we verified the superiority of the cosine similarity over other alternatives. We proposed to use mean-shift and a combination of analogy interpretations and showed how they lead to improvements in accuracy.

[4]Global Vectors for Word Representation

## References


[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[2] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." in *EMNLP*, vol. 14, 2014, pp. 1532–1543.

[3] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations." in *HLT-NAACL*, 2013, pp. 746–751.

[4] D. A. Jurgens, P. D. Turney, S. M. Mohammad, and K. J. Holyoak, "Semeval-2012 task 2: Measuring degrees of relational similarity," in *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2012, pp. 356–364.

[5] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015.

[6] O. Levy and Y. Goldberg, "Dependency-based word embeddings." in *ACL (2)*, 2014, pp. 302–308.

[7] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, no. 3, p. 1, 1988.

[8] R. Řehůřek and P. Sojka, "Software Framework for Topic Modelling with Large Corpora," in *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50, http://is.muni.cz/publication/884893/en.

[9] A. M. J. Schakel and B. J. Wilson, "Measuring word significance using distributed representations of words," 08 2015. [Online]. Available: http://arxiv.org/abs/1508.02297