**Karlsruhe Reports in Informatics 2017,3**
Edited by Karlsruhe Institute of Technology,
Faculty of Informatics
ISSN 2190-4782

# On the Various Semantics of Similarity in Word Embedding Models

Ábel Elekes, Martin Schäler, Klemens Böhm

2017

KIT – University of the State of Baden-Wuerttemberg and National Research Center of the Helmholtz Association

# On the Various Semantics of Similarity in Word Embedding Models

Ábel Elekes
Karlsruhe Institute of Technology
Karlsruhe, Germany
abel.elekes@kit.edu

Martin Schäler
Karlsruhe Institute of Technology
Karlsruhe, Germany
martin.schaeler@kit.edu

Klemens Böhm
Karlsruhe Institute of Technology
Karlsruhe, Germany
klemens.boehm@kit.edu

## ABSTRACT

Finding similar words with the help of word embedding models has yielded meaningful results in many cases. However, the notion of similarity has remained ambiguous. In this paper, we examine when exactly similarity values in word embedding models are meaningful. To do so, we analyze the statistical distribution of similarity values systematically, in two series of experiments. The first one examines how the distribution of similarity values depends on the different embedding-model algorithms and parameters. The second one starts by showing that intuitive similarity thresholds do not exist. We then propose a method stating which similarity values actually are meaningful for a given embedding model. In more abstract terms, our insights should give way to a better understanding of the notion of similarity in embedding models and to more reliable evaluations of such models.

## Keywords

Word embedding models; similarity values; semantic similarity.

## 1. INTRODUCTION

**Motivation.** One important objective of so-called *distributional models* [1] [2] is to capture the semantic similarity of words, based on their context in large corpora. If one is able to quantify their similarity, there will be a good understanding of the actual meaning of a word, by knowing which words are similar. Adopting the taxonomy of Baroni et al. [3], one can discern between count-based distributional models [4] [5] and training-based, predictive models, also called *embedding models* [6] [7] [8] [9] [10] [11] [12]. Embedding models use vectors to represent words in a low-dimensional space to quantify semantic similarities between words. All these models have in common that two words are *semantically similar* if the vectors representing them are close according to some distance function.

Embedding models have received renewed popularity after Mikolov et al. presented new neural network based models [11] [12]. In comparison to count-based models, the training of such models scales very well even to huge corpora, while learning high-quality word vector representations. With that, embedding models have become key tools in Natural Language Processing (NLP), showing impressive results in various semantic tasks such as *similarity detection* or *analogical reasoning* [3]. Despite the limited linguistic information distributional models contain, embedding models have proven to be successful not only in elementary tasks, but also in complex ones such as part-of-speech (POS) tagging [7], named entity recognition (NER) [13], dependency parsing [14], social media sentiment analysis [15], image annotation [16], and machine translation [17] [18] [19].

It currently is an open question whether embedding models are superior to traditional count-based models. Some research suggests that they indeed are, in various similarity and analogy detection tasks [3] [12]. But others have argued that this superiority is only a result of better parameter settings [20] [21] [22] [23]. However, these papers are only *using* the similarity attribute of the models, while the following questions remain open: What do similarity values from those models actually mean? For instance, are low values of similarity comparable to each other? To illustrate, if Word A is 0.2-similar to Word B and 0.1-similar to Word C on a [-1, 1] scale, should we say that A is more similar to B than to C, or does it not make any difference at these low levels of similarities? Are there 'natural' thresholds for similarity, such that values above (beneath) it represent a definite similarity (dissimilarity) of two words? For example, if A is more than 0.5-similar to B, then are A and B always semantically similar? How about the same questions with similarity lists, i.e., lists of words most similar to a certain words, sorted by similarity? For instance, can we say that the 100 words most similar to an arbitrary word are always similar to this one, or words not in the top 500 are always dissimilar? When exactly is it meaningful to stick to the natural idea of taking the top N most similar words for a certain word and deem them similar? In this paper we study and answer all these questions. These questions are not just academic in nature; any paper relying on comparisons of similarity values might lack validity if these questions remain open.

**Challenges.** Several issues arise when studying similarity in computer science: How does an evaluation dataset look like, what does it contain? How to create good baseline datasets, how do they measure similarity, and how to evaluate a model on them?

As for the first question there is a generally accepted simple structure how the similarity datasets should look like, and any widely used dataset such as WordSim353 [24] or MEN [25] is formatted like this. These datasets contain a set of word pairs and similarity scores for every pair, set by human annotators. To show the difficulty of how to create good such datasets, think of the following linguistic challenge pointed out by Hill et al. [26] in this context: What is the definition of similarity? Are *cup* and *coffee* similar words or only *associated*, i.e., dissimilar? In general, does relatedness or associatedness imply similarity or not? – They argue that word pairs which are only associated should not have high similarity scores, in contrast to datasets such as WordSim353 or MEN, where this is the case, i.e., associated pairs do have high similarity scores. Batchkarov et al. [27] also address the problem of creating good baseline datasets. They show that it is challenging even for human annotators to assign similarity scores to certain word pairs. For example, they show that the similarity scores for the *tiger-cat* pair range from 5 to 9 on a scale of ten in the WordSim353 dataset. They also provide example word pairs where the similarity scores differ significantly when the pairs are contained in different data sets. They argue that this is the result of the different notions of similarity these datasets use.

Next, Avraham et al. [28] identify problems regarding the evaluation of the models. They argue that the use of the same rating scale for different types of relations and for unassociated pairs of words makes the evaluation biased. For example, they say that it is meaningless to compare the similarity value of *cat-pet* to *winter-season*, because they are unassociated, and models which rank the wrong word pair higher should not be punished. If *cat-pet* has a similarity score of 0.7, and *winter-season* has one of 0.8 in a similarity dataset, an evaluation should not punish a model which ranks *cat-pet* higher. They also find it problematic how the conventional evaluation method measures the quality of a model. It calculates the Spearman correlation of the annotators ranking and the model ranking, without considering the similarity values further. To illustrate, such an evaluation penalizes a model that misranks two low-similarity, unassociated pairs (e.g.: *cat-door, smart-tree*) just as much as one that misranks two objectively distinguishable pairs (e.g.: *singer-performer, singer-person*).

Having said this, the concept of similarity remains ambiguous, and understanding similarity values remains difficult as well, affecting several NLP tasks, especially when it comes to evaluate embedding models on these tasks.

**Contributions.** To understand what similarity values in embedding models mean, we evaluate how different parameter settings (e.g.: size of the corpus they are trained on, vocabulary size) influence the similarity values of the models. We do so by systematically training various models with different settings and comparing the similarity value distributions. One intention behind these experiments also is to confirm that the meaning of similarity values of two terms is not sufficiently clear, and to reveal that this also holds for the relationship between model parameters and similarity values. We show that indeed it is not always meaningful to compare two word pairs by their similarity values.

A core contribution of ours then is the discovery that meaningful similarity threshold values do indeed exist, and we show that they can be found. We do so by calculating similarity value and similarity list aggregates based on WordNet [29] similarity as the baseline and evaluate the resulting similarity distributions of the models with statistical tests. It turns out that these thresholds are not general and should be calculated for every individual model using the method we present in this paper. At this point, our evaluation connects with the parameter evaluation of the models just mentioned: The evaluation shows that altering the parameters does not change our method; all similarity value distributions of the models are fundamentally similar. This is an important step both regarding the design of future word embedding models as well as the improvement of existing evaluation methods.

# 2. Fundamentals and Notation

In the following, we first define embedding models and their parameters in general. We then introduce two relevant models which we rely on in the paper.

## 2.1 Background on Word Embedding Models

Word embedding models "embed" words into a low-dimensional space, representing them as dense vectors of real numbers. Vectors close to each other according to a distance function, often the cosine distance, represent words that are semantically related.

Formally, a word embedding model is a function *F* which takes a corpus *C* as input, such as a dump of the Wikipedia, generates a dictionary *D* based on the corpus and associates any word in the dictionary $w \in D$ with a *d*-dimensional vector $\boldsymbol{v} \in \mathbb{R}^d$. The *dimension size* parameter (*d*) sets the dimensionality of the vectors. It usually ranges between 50 and 1000 with embedding models. The training, i.e., iteratively associating vectors with words in the dictionary, is based on word-context pairs $w \times c \in D \times D^{2\times win}$ extracted from the corpus. *win* is the *window size* parameter, which determines the *context* of a word. For example, a window size of 5 means that the context of a word is any other word in its sentence, and their distance is at most 5 words. However, there are further parameters that affect the generation of the dictionary. One is the *minimum count* parameter (*min_cnt*). When creating the dictionary from the corpus, the model adds only words into the dictionary which appear at least *min_cnt* times in the corpus. An alternative is to set the *dictionary size* directly as a parameter (*dict_size*). This means that the model adds only those words to the dictionary which are in the *dict_size* most frequent words of the corpus. In this paper we rely on the *dict_size* parameter, because we find it easier to handle in our experiments. With this variant, the corpus does not influence the size of the dictionary.

Having said this, we define word embedding models as:

$$F(C, d, win, dict\_size) \in \mathbb{R}^{|D|\times d}.$$

*dict_size* is not necessarily equal to the size of the dictionary $|D|$. For example, it is unequal when the number of distinct words in the corpus is smaller than *dict_size*. *F* is not deterministic, as it may use random values when initializing the word vectors.

## 2.2 Word Embedding Model Realizations

In this paper, we work with two well researched embedding models, Mikolov et al.'s Word2Vec model [12] and Pennington et al.'s Glove model [10]. These models learn the vector representations differently. Word2Vec models use a neural-network based learning algorithm. It learns by maximizing the probability of predicting either the word given the context (Continuous Bag of Words model, CBOW), or the context given the current word (Skip-Gram model, SG) [11] [12]. Glove trains the word vectors by explicitly factorizing the log-count matrix of the underlying corpus, wrt. word-context pairs [10]. Levy et al. [30] have shown that the SG model is implicitly factorizing a word-context pointwise mutual information matrix. This means that the objectives of the two models and sources of information they use are not overly different, and, more important here, is that they share the same parameter space. See [31] for a further comparison.

When building models ourselves, we use the *gensim* software package [32] for the Word2Vec models and the Glove toolkit[1] for the Glove models. More specifically, we use the gensim toolkit in Python. It allows querying any word in the model dictionary for its similarity with any other word. This means that for any word there is an indexed list containing every other word in the dictionary, sorted by similarity. In this paper we use the terms *list index* and *position in the list* as synonyms. The similarity values are floating point numbers between -1 and 1, with 1 being the highest similarity. We will differentiate between the *similarity values* of models and *similarity lists*. In the first case we are only concerned with the similarity value of a word pair and not its position in those lists. In the second case our interest is the reverse.

# 3. Embedding Model Parameter Investigation

In this section, we investigate how the different parameters affect the similarity values of the models. We are particularly interested in identifying parameters that change the stochastic distribution of the similarity values significantly. These insights are generally relevant to understand word embedding models. We also require such insights in the next section for our threshold evaluation.

[1] http://nlp.stanford.edu/projects/glove/

## 3.1 Investigation Objectives

A core contribution of this paper is to find meaningful thresholds both for similarity values and for similarity lists for a given model. To this end, we evaluate how different models and their parameters affect the similarities. We will show that similarities in embedding models when trained with different parameters can differ significantly. So our first hypothesis is as follows:

**Hypothesis 1.** It is not possible to find general value and list thresholds that are reasonable for all embedding models, only for specific ones.

We plan to confirm this hypothesis by showing that the similarity value distributions have such statistical characteristics such as different mean values of different highest similarity values which makes uniform threshold values meaningless. We present two examples of such models in the following.

**Example 1.** *Think of two models, Model A with an average similarity between two words of 0.0, and Model B with an average of 0.1. This means that the similarity value is negative for roughly half of the pairs in Model A and for roughly 1% of the pairs in Model B. If one now assumed that a negative similarity value implied dissimilarity between the words of the pair, this assumption would have a highly different meaning for the two models.*

***Example 2.*** *Again think of two models. The highest similarity score of a word pair is 0.9 in Model A and 0.6 in B. Saying that a pair with a similarity above 0.7 is definitively similar could be meaningful in Model A, but makes less sense in B. This is because there is no word pair with this similarity value in this model.*

Although the similarity value distributions of the models can significantly differ in certain characteristics, we hypothesize that they are all similar in shape, with only their means and standard deviations depending on the parameters.

**Hypothesis 2.** While the learning algorithms and parameters influence the similarity value distributions of the models, these distributions are very similar in shape.

We plan to confirm this hypothesis as follows. First we normalize all distributions, so that they have 0 mean and 1 standard deviation. We then randomly draw 1000 values from all distributions and pairwise compare the samples by means of the two-sample Kolmogorov-Smirnov (K-S) test [33] with 99% confidence. This test checks if two samples are drawn from the same distribution.

For the overall understanding of the similarity values and lists, it is important to know how the model selection and the parameters affect the similarities. Our main contribution in this section is that we do the evaluation systematically for all the parameters and models already introduced. This means that we evaluate how the model selection (*F*), the corpus (*C*), the dimensionality (*d*), the window size (*win*) and the models dictionary size (*dict_size*) interact with the similarity values and lists.

## 3.2 Experiment Setup

In this paper, we work with Chelba et al.'s 1 Billion word dataset [34] as training corpus. It has shown to be a good benchmark dataset for language modelling, with its great size, large vocabulary and topical diversity. [34] The dataset is around 4 Gb in size as a text file and contains almost 1 billion words in approximately 30 million English sentences. The sentences are shuffled, and the data is split into 100 disjoint partitions. This means that one such partition is 1% of the overall data. We train all our models using this dataset as training corpus.

In the following, for every parameter, we present our results in the same way. In particular, we graph results in two figures. First there are similarity value distributions of the models. For these plots, we randomly select 10,000 words from the model dictionary and calculate the similarity values of *every other word* to them. Then we group the values in 0.01 intervals and count the number of values in each group. Thus, the x-axis represents the similarity values from [-1,1], the y-axis the share of the values per group.

The second figures contain the results from the similarity lists experiments. In these experiments, we randomly select 10,000 words $(w_1, w_2, \dots, w_{10000})$ from the dictionary of the model. For each of these words, we compute the most similar thousand words $w_{i,1}, w_{i,2}, \dots, w_{i,1000}$ for $i \in \{1, \dots, 10000\}$, together with their respective similarity values, $t_{i,1}, t_{i,2}, \dots, t_{i,1000}$, i.e., $t_{i,j}$ is the similarity value between words $w_i$ and $w_{i,j}$. $w_{i,1}, w_{i,2}, \dots, w_{i,1000}$ is sorted by the similarity values. Because of this sorting for every $i$, it holds that $t_{i,j_1} \geq t_{i,j_2}$, for any $j_1 < j_2$. We then calculate the average similarity value for every list index $avg\_sim(j) = average(t_{\cdot,j})$. Finally we plot the results with the x-axis being the list indices $(j)$ and the y-axis the average similarities $(avg\_sim(j))$. Although the $avg\_sim()$ function is only defined for arguments that are natural numbers, the plots connect the points to arrive at a smooth curve, for better visibility.

At this point we are not trying to answer why different parameters affect the similarity values as they do; we are investigating *how* they affect the values. This means that we are not making qualitative statements, i.e., we are not concerned how parameters affect the quality of the models on different semantic tasks. We are not making any statement that any model is better or worse than the other, but only how and to which extent they are different. In other words, we focus on the hypotheses from Section 3.1.

## 3.3 Model Selection

The first parameter whose effect we investigate is the model itself. We consider the three already introduced models, Word2Vec SG, Word2Vec CBOW and Glove. We build all three models on the full 1 billion words dataset with the same parameter settings. As we have noted in Subsection 2.2, these models share the same parameter space. This means that we can use the exact same parameter setting for the models. The parameters we use are $d = 100$, $win = 5$, $dict\_size = 100{,}000$, the default settings for the Word2Vec models. These values have shown to be a good baseline setting for different semantic tasks [35] [26].

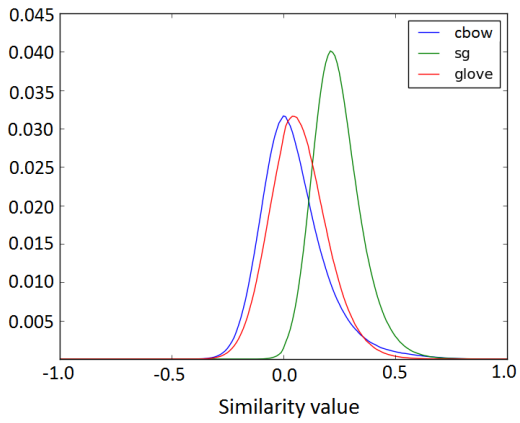


**Figure 1 Learning algorithms similarity value distributions**

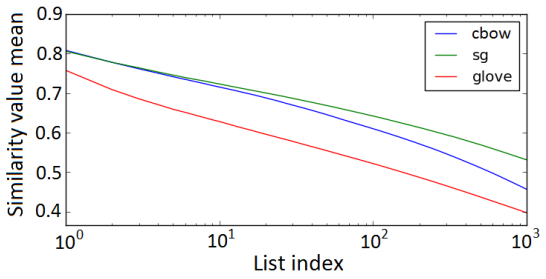


**Figure 2 Learning algorithms similarity values by list indices**

*Similarity Values*. Figure 1 shows the approaches to differ much in similarity values. The CBOW and Glove models are almost identical, although Glove has slightly higher values. But the SG algorithm generally produces higher similarity values than the other two, and only few pairs of words have negative similarities. This implies that, while words in the CBOW and Glove model fill almost the entire space, the SG model learns word vectors positioned at a high density area of the space, leaving the remainder of the space sparse. We test Hypothesis 2. by comparing the normalized distributions pairwise, cf. Section 3.1:

$K\_S\_p\_value(sim\_dist_i, sim\_dist_j) > 0.01$ for every $i, j \in \{cbow, sg, glove\}$

We conclude that the models are similar in their distributions.

Regarding Figure 2, although the Glove model generally produces higher similarity values than CBOW, the values by list position are smaller than with both Word2Vec models. At the end of the top 1000 list, the values with the SG model are the highest ones.

*Result interpretation.* Both results indicate that our hypotheses hold, i.e., the distributions of the similarity values are indeed very similar, although at the same time visibly different in certain characteristics. This is important: It indicates a certain robustness of embedding models and generalizability of empirical results. The differences also show that we cannot set general thresholds which apply to every model.

## 3.4 Corpus

Now we investigate how the size of the corpus affects similarity values and lists. We compare five different models, which are trained on differently sized parts of the 1 billion word benchmark dataset. Sampling is performed by retaining different percentages of the 1 billion words data used for the training. The models have every other parameter identical. We train them with the Word2Vec CBOW model, with $d = 100$, $win = 5$, $dict\_size = 100{,}000$.

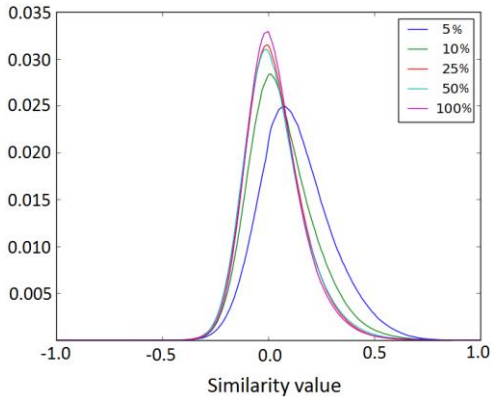


**Figure 3 Corpus size similarity value distributions**

*Similarity Values.* According to Figure 3, the bigger the corpus size, the narrower the distribution is. We can see that using 25% of the corpus is almost identical to using 50%, and very close to using the entire corpus for training. We test the normalized similarity distributions pairwise with the K-S test. Again every p-value is above 0.01. This means that the models are very similar.

Figure 4 shows that at the top 10 similar words there is almost no difference between the models. For higher indices, models trained on smaller corpora generally have higher similarity values, but the three models trained on bigger corpora are almost identical.

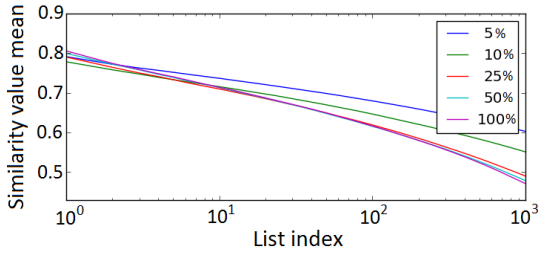


**Figure 4 Corpus size similarity values by list indices**

*Result interpretation.* We conclude that models trained on more than 1 Gb of text data or approximately 250 million words have almost identical similarity value distributions. All distributions are similar, but visibly different at the same time, especially for smaller corpus sizes. This confirms our hypotheses.

## 3.5 Dimensionality

When measuring similarity with the cosine distance, the dimensionality of the embedding model is a parameter that strongly affects its similarity values. In this section we train every model with the Word2Vec CBOW model with different dimensionalities on the full corpus, with $win = 5$, $dict\_size = 100{,}000$.

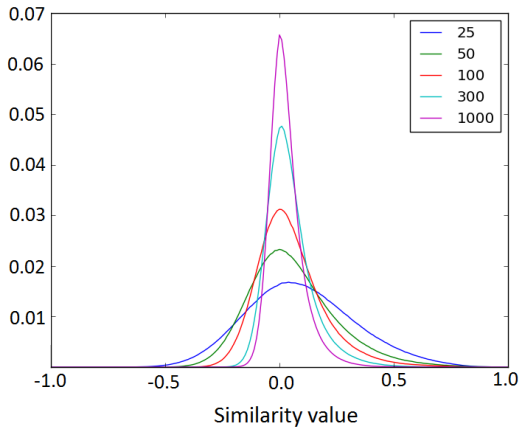


**Figure 5 Dimension size similarity value distributions**

*Similarity Values.* Figure 5 shows that the higher the dimensionality the model is built with, the narrower the similarity distributions are. We have expected this, as vector spaces with lower dimensionality are denser when filled with 100,000 words than ones with higher dimensionality. This leads to closer words and higher similarity values. In contrast to the visibly different distributions, we again see that the distributions are similar, as the K-S test did not distinguish the normalized distributions, with 99% confidence.

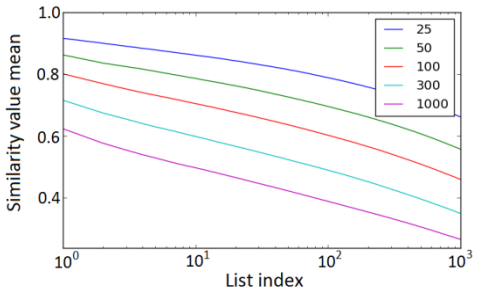


**Figure 6 Dimension size similarity values by list indices**

Figure 6 is even more straightforward – the higher the dimensionality, the lower the similarity values in the similarity lists are.

*Result interpretation.* The dimensionality parameter confirms our hypotheses in a manner that we deem clearer than the previous experiments. Namely, the models are fundamentally very similar and at the same time different. We cannot set any general threshold values, because average and highest similarity values are very different. But the distributions only differ in their standard deviations, which means they are fundamentally very similar.

## 3.6 Window Size

In this section we train every model with the Word2Vec CBOW model on the full 1 billion word corpus, with $d = 100$, $dict\_size = 100{,}000$ and five different window size settings.

*Similarity Values.* Figure 7 shows that there is only a slight difference of similarity values between models trained with different window sizes. It is noteworthy that, when the window size is 1, the distribution has a higher mean. This implies that the model has an area of higher density in the word vector space. The distributions are very similar without even normalizing them. The pairwise K-S test confirms this, as again every p-value is above 0.01. So the normalized distributions are almost identical.

The similarities corresponding to different positions in the similarity lists on Figure 8 tell us that the differences between the models are very small. Still we can see that the smaller the window size, the higher the similarity values are.

*Result interpretation.* These results are very similar to the ones for dimensionality, with both figures consistently changing with the parameters, only on a smaller scale in this current case. Only the smallest window size parameter, i.e., $win = 1$, interferes with the similarity distribution in an inconsistent manner, but it also changes the mean of the distribution.

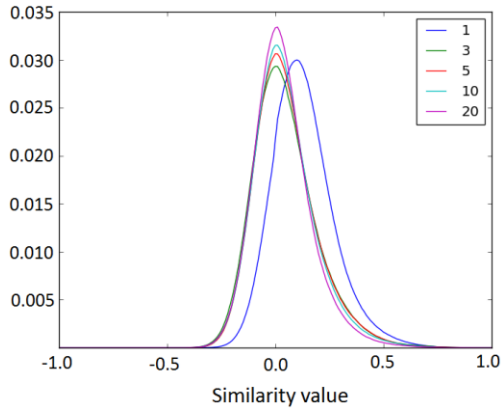


**Figure 7 Window size similarity value distributions**

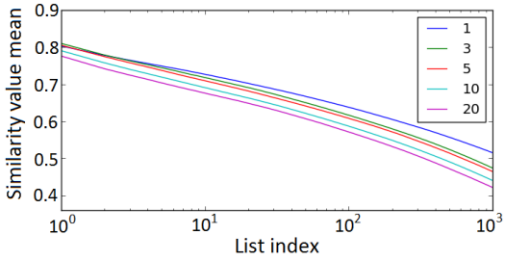


**Figure 8 Window size similarity values by list indices**

## 3.7 Dictionary Size

In this section we evaluate how the dictionary size of the models affects their similarity values and lists. We train five models with different dictionary sizes with the Word2Vec CBOW model on the full corpus, with $d = 100$, $win = 5$.

*Similarity Values.* Figures 9 show that the dictionary size does not affect the similarity value distribution of the models up to a certain size. With very large dictionaries however, the numerous noise words (typos, unmeaningful words, contraction, etc.) have a very strong effect on the distribution. The same effect is visible in the dimensionality experiment, i.e., when considering many words in the dictionary, the 100 dimensional space is not large enough for the models to distribute them sufficiently. This leads to wider similarity value distributions and even to an asymmetric distribution with the largest dictionary.

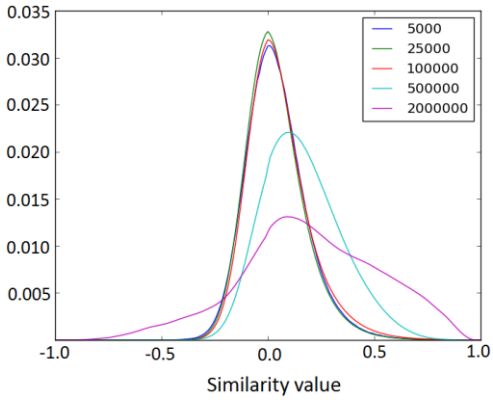


**Figure 9 Dictionary size similarity value distributions**

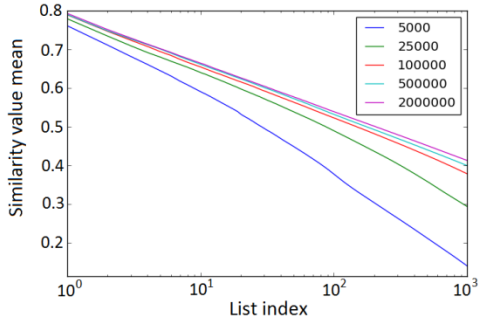


**Figure 10 Dictionary size similarity values by list indices**

The K-S test confirms the similarity distribution of the 2 million word dictionary model to significantly differ from the others, as

$K\_S\_p\_value(sim\_dist_{2M}, sim\_dist_i) < 0.01$ for every $i \in \{5k, 25k, 100k, 500k\}$.

Let us now look at the similarities of items with the same position in the different similarity lists in Figure 10. We find it interesting that the big dictionary models are almost identical to the baseline 100,000 words dictionary model. The smaller dictionary models naturally have lower similarity values. This is because there are fewer words which are close to each other.

*Result interpretation.* This is the only evaluation where one distribution does not have the bell shape observable in all other experiments. This is a consequence of an unreasonably large dictionary. Apart from this, even in a 500 thousand word dictionary the hypotheses stand, as the distributions are similar.

## 3.8 Summarizing Parameter Effects

Our evaluations in this section have confirmed the two hypotheses. We have shown that different algorithms and parameter settings indeed affect the value distributions of embedding models significantly, but at the same time they have the same abstract shape. All value distributions of the models are Gaussian-like, except for one unrealistic setup. This remarkable robustness implies that one can now work with one specific model and adjust the thresholds calculated to other models later if necessary.

To our knowledge, such systematic experiments have not been done before for embedding models. For systematic evaluations of the effect of parameters on the quality of word embedding models see Hill et al. [22], Altszyler et al. [36], Chiu et al. [35] and Lin et al. [37]. These studies evaluate how the corpus size, window size and dimensionality affect the results of the models on similarity and analogy tasks. We will show in the next section that all these evaluations suffer from one thread of validity: They do not take the cardinality of the similarity values into further consideration when comparing the similarity of two word pairs.

# 4. Finding Meaningful Similarity Values

In this section, we contribute to the question when exactly similarity values are meaningful in word embedding models. First, we show that intuitive similarity thresholds do not exist. Then we propose a general method to find meaningful similarity value thresholds for a given model and baseline (e.g., WordNet) and examine the validity of this method with various models.

## 4.1 Investigation Objectives

Reviewing various approaches [24] [25] has revealed that their evaluations compare similarity values and list indices without taking their size into account. This means that they deem, say, two word pairs with similarity values 0.8 and 0.7 just as different as ones with values -0.2 and -0.1. But there is no examination of the distribution of the similarity values of word vectors indicating that this is reasonable. In fact, it might turn out that a more differentiated perspective is required. From Section 3, we already know characteristics of the distributions of the similarity values of the word vectors, for example their average and highest similarities. But we do not yet know how vector similarity corresponds to word similarity, such as similarity measures in WordNet.

## 4.2 Intuitive Similarity Thresholds

We now examine experimentally whether meaningful intuitive thresholds for similarity values exist. Many approaches using similarity values or lists implicitly presume this, as they for example only work with the top k most similar words. Our results indicate that respective results may be misleading.

### *4.2.1 Experiment setting*

Our procedure is similar to the one in Section 3.2. The main difference is that we compare the results to a baseline, WordNet in this case. We conduct two series of experiments, one for similarity values and one for lists. In both cases, we calculate word pair similarity aggregates, one grouped by values, the other one grouped by list indices, based on WordNet similarity scores. We do so in order to understand at which extent similarity values are meaningful in embedding models. We use the Leacock and Chodorow (LCH) [38] similarity measure in WordNet for the evaluation. We have chosen this measure because it is knowledge-based. This means that it does not use any external resource or corpus, but only the WordNet ontology itself. It also is a popular, highly researched measure and has proven to be a useful baseline for semantic similarity [39] [40] [41]. We have implemented our experiments with WordNet using the NLTK python toolkit [42]. For more information on similarity measures in WordNet see Meng et al. [43]. In all our experiments in this section, the baseline similarity measure (LCH) is replaceable. This means that one simply can rerun any experiment with a more specific, say, corpus based similarity measure, as well as with another model.

The model we use in this section is trained with the CBOW algorithm on the full 1 billion word corpus, with $d = 100$, $win = 5$, $dict\_size = 100{,}000$, the default model and parameter settings in the gensim Word2Vec toolkit.

### *4.2.2 Similarity value and list experiments*

For the first experiment, we compute the similarity values of every word to *any* other word in the dictionary: $wp_{i,j}$ is a word pair containing words $w_i$ and $w_j$ for $i, j \in \{1, \dots, 100000\}$, $t_{i,j}$ is their similarity. We now group these word pairs by their similarity value in 0.01 intervals: $G_{-1.0}$, $G_{-0.99}, \dots, G_{0.0}$, $G_{0.01}, \dots, G_{1.0}$ are these groups. To illustrate, $G_{0.05}$ contains all $wp_{i,j}$ word pairs where $0.04 < t_{i,j} \leq 0.05$ holds. Then we calculate the average similarity with the LCH measure in each group:

$$avg\_sim(G_k) = average(LCH\_dist(wp_{i,j})), \text{ where } wp_{i,j} \in G_k.$$

In the second experiment, we create the full similarity lists for every word in the dictionary, $w_{i,1}$, $w_{i,2}, \dots, w_{i,100000}$, i.e., for every $i \in \{1, \dots, 100000\}$. We create groups of word pairs $(G_1, \dots, G_{100000})$. $G_k$ contains the pair $(w_i, w_{i,k})$ for every $i \in \{1, \dots, 100000\}$. We then calculate the average similarity for every group with the LCH measure:

$$avg\_sim(G_k) = average(LCH\_dist(wp_{i,j})), \text{ where } wp_{i,j} \in G_k.$$

For both experiments, if a word is not in the WordNet dictionary, we remove all word pairs including it from the groups, in order to make the aggregation unbiased. We observe that the standard deviations are relatively high in the groups: In the similarity value groups, it is between 0.25 and 0.55, in the similarity list groups between 0.25 and 0.6. We will return to this observation when discussing the outcomes of the experiments.

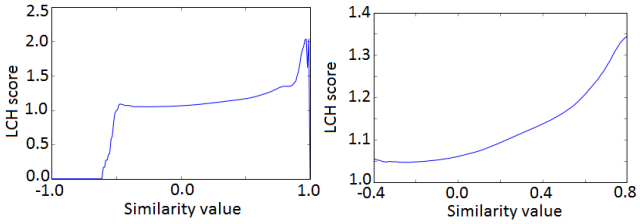


**Figure 11-12 LCH scores aggregates by similarity values**

In the similarity value distribution experiments, we evaluate the results only for values between -0.4 and 0.8. This is because the small number of word pairs with similarity values outside of this interval makes the data in these ranges noisy. This is in line with

our parameter evaluation results, as we can see from the graphs in Section 3 that the vast majority of word pairs have similarity values in this range for the model used in this section.

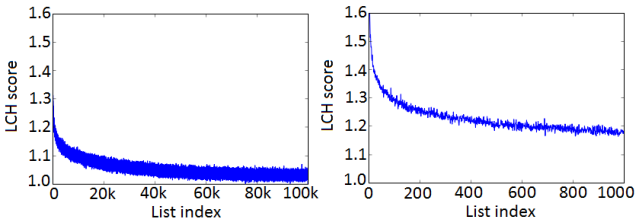


**Figure 13-14 LCH score aggregates by list indices**

To find meaningful threshold values, we check the plots of the averages of the similarity value distributions for patterns that could imply meaningful values. We do so in two steps. Our first step is an intuitive inspection of the figures; the second step is a statistical analysis of the graphs. We now discuss these steps.

## 4.3 On the Existence of Intuitive Similarity Thresholds

When analyzing the results visually, we hope to find horizontal segments in the result graph or other phenomena such as breaks, i.e., flat segments of the graph followed by a steep incline or decline, which might stand for certain properties of the models. A horizontal segment, for example, would mean that there is no difference in similarity between the values forming this line. To illustrate further, if Figure 14 was horizontal between list indices 800 and 1000, we could interpret this as follows: There is no general difference in similarity between a word being the 800th or the 1000th most similar word to a given word. Thus, it is meaningless to differentiate between words at these similarities. The same would follow for the similarity value distribution if there was a horizontal segment there. Other phenomena such as a break in the figure would imply a general change in similarity. For example, if there was a break, we could interpret it as a threshold between relevant and irrelevant similarity values at first sight. However, as observable in Figures 11-14, such an intuitive approach does not yield any useful result in our case. This is because there are no obvious horizontal segments or breaks in the graphs.

## 4.4 Towards Meaningful Threshold Values Based on Unequal Mean Values

The previous step has not identified any patterns pointing to intuitive threshold values for similarity. Hence, we now aim for a statistically sound derivation of meaningful threshold values, in contrast to a mere visual inspection.

### *4.4.1 Confidence-based threshold identification*

The general idea is examining the results of our experiments with statistical tests. We test the hypothesis that two populations have equal means, without assuming that they have equal variance. Here, these populations are the LCH scores of two groups of word pairs. Formally, such a group ($LCH\_G_k$) is as follows: $LCH\_G_k = \{LCH\_dist(wp_{i,j}) : wp_{i,j} \in G_k\}$, where $G_k$ is either a similarity value or a similarity list group, as introduced in Section 4.2.2. We use Welch's unequal variances t-test [44] for our experiments, a widely used two-sample statistical test for this problem. So the answers to the research questions from the introduction are statistical in nature, i.e., we will give answers with a certain confidence such as 99%, based on Welch tests.

Our tests are as follows: We compare two groups ($LCH\_G_k$, $LCH\_G_l$), as introduced above, with the Welch test. The groups are obtained by similarity values (Experiment 1) or by similarity list indices (Experiment 2). The null hypothesis in a Welch test is that the two groups have equal means. One either rejects the null hypothesis at a confidence level chosen apriori (99% in our case), or there is not enough evidence to do so. In case of a rejection, we conclude that there is a significant difference between the two groups in terms of similarity. I.e., the group with the higher LCH mean contains significantly more similar word pairs.

### *4.4.2 Experimental results for similarity values*

For the similarity value group evaluation, we first test the exemplary questions asked in the introduction. We then investigate generally at which extent similarity value groups are different.

***Q1.*** *Are low values of similarity comparable to each other?*

***Q2.*** *If Words A and B have a higher similarity value than A and C (say 0.2 and 0.1), is A more similar to B than to C?*

For Q2, we test the following null hypothesis: The aggregated LCH scores have the same mean values for the word pairs with a 0.10 and with a 0.20 similarity value. – The number computed on our corpus is as follows:

$$welch\_test\_p\_value(LCH\_G_{0.10}, LCH\_G_{0.20}) = 5.19e^{-9} < 0.01$$

So we conclude with 99% confidence that the hypothesis is false. We infer that the word pairs with 0.20 similarity values in general are more similar to each other than the pairs with 0.10 similarity values. In other words, to answer Q1, even at these low levels of similarity, differences in value have a meaning.

We now turn to the systematic experiment. For every group, we search the next group with higher index which significantly differs in LCH scores with 99% confidence. See Figure 15. The values can be understood as follows: For the -0.30 similarity value group (X axis), to give an example, the next successive group which significantly differs in similarity is the -0.17 similarity value group (Y axis). Starting from the -0.18 (X axis) group every successive group has a significantly higher LCH score mean than the previous one. On the other hand, there is a bend in the figure at -0.18. It means that at low values of similarity, i.e., below -0.18, there is no significant evidence that the higher similarity value group implies higher LCH similarity scores. We conclude that below the -0.18 similarity value there is no significant difference between the groups.

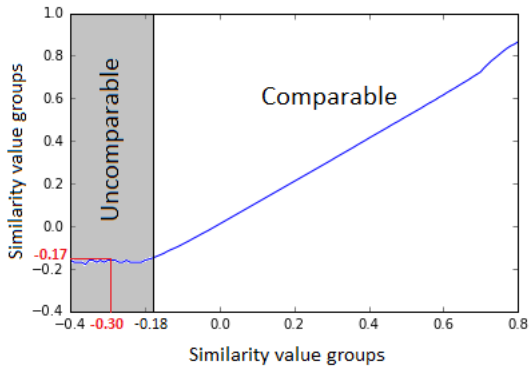


**Figure 15 Groups with significant differences in LCH mean scores by similarity values**

Another way to understand these values is as follows: Somewhat naturally, we assume that the -0.40 similarity value group contains dissimilar word pairs. This is because it is the group with the pairs with the smallest similarity values. For this group we calculate the

next group with significantly higher LCH mean score, the -0.18 similarity value group. This means that between -0.40 and -0.18 there is no significant difference in LCH scores between the groups. Based on our assumption that the -0.40 group contains dissimilar word pairs, we conclude that the word pairs with similarity values between -0.40 and -0.18 are dissimilar.

Because of the relatively high standard deviation in the groups, we cannot conclude that all word pairs in these groups are dissimilar, but we can say that the groups do not differ significantly. For higher similarity values, i.e., above -0.18, every group is significantly different, as we have seen. This means that any increase in similarity, even if it is only 0.01, implies a higher similarity of the word pairs. Again, we cannot say this for every specific word pairs, because of the high deviation, but only in general terms, for the groups as a whole.

Overall, we conclude that the similarity value groups are significantly different from each other above -0.18 and not different below this value. This also can be seen visually, as there is a characteristic bend in Figure 15 at -0.18 on the x-axis.

### *4.4.3 Experimental results for similarity lists*

We now investigate the same exemplary questions asked in the introduction with similarity lists.

***Q3.*** *Can we say that being in the top 100 list of most similar words always implies similarity, or not being in the top 500 list always implies dissimilarity?*

***Q4.*** *What are meaningful cutoff values, and how to find them?*

We answer these questions with the following experiments. Our experiments with similarity lists actually are the same as just before, but with the word pairs being grouped by list indices. Figure 13 shows that there is a long almost horizontal noisy stripe of LCH averages. We are making the same tests for the index groups ($G_k, k \in \{1, \dots, 100000\}$) as we have with the similarity value groups, again with 99% confidence.

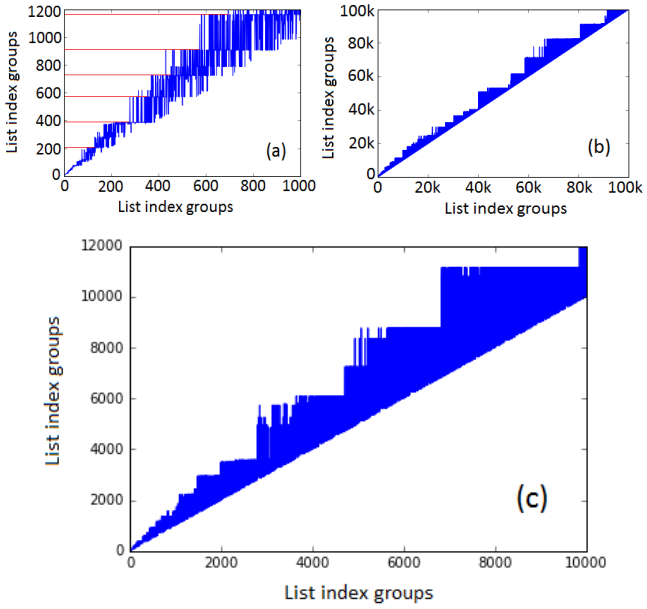


**Figure 16 a, b, c Groups with significant differences in LCH mean scores by similarity indices**

For every index group, we search the next group with higher index with a significantly different LCH similarity mean using that test. The figure shows the following: At the smallest indices even small differences in the indices imply significantly different mean score. But as the indices increase, the bigger the differences have to be between groups to yield a significant difference in the mean.

Figures 16a-c show that there are certain indices which generally identify the significant differences. These indices correspond to groups with particularly high LCH mean scores, and because of that, they are significantly different from many lower index groups. The horizontal lines in Figure 16a identify them.

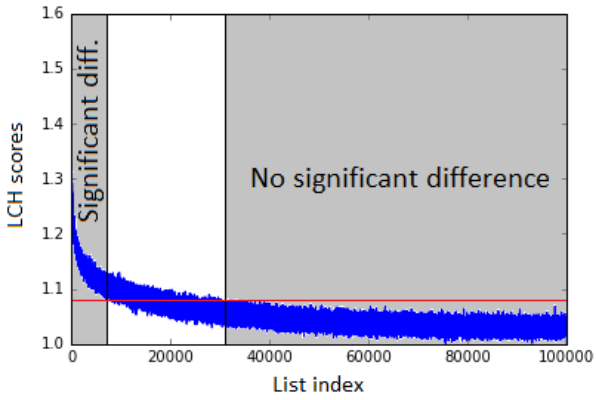


**Figure 17 Meaningful list indices**

Just as we have done with the similarity values, we assume that the last group of word pairs, i.e., pairs consisting of a word and its least similar word, are dissimilar. We test two items:

- What is the last group with an LCH similarity score significantly different from the last group overall? Formally, what is the highest index $(i)$ so that, for every $j > i$, $welch\_test\_p\_value(LCH\_G_{100000}, LCH\_G_j) > 0.01$ holds?
- What is the first group that is not significantly different from the last group? Formally, what is the lowest index $(i)$ so that $welch\_test\_p\_value(LCH\_G_{100000}, LCH\_G_j) < 0.01$ holds, for every $j < i$?

The answers to these questions are the 31584th group and the 6094th group, respectively. Namely, the horizontal line in Figure 17 is the one separating the groups whose LCH mean scores are significantly higher than the one of the last group from the rest. We conclude that indices higher than 31584 are statistically not different from the last group. Based on our assumption, our interpretation is that they contain dissimilar word pairs. On the other side, all groups with indices below 6094 have a higher LCH mean than the last group. This means that they all contain significantly more similar word pairs. Again this does not mean that all the word pairs in these groups are dissimilar or similar, respectively, but that the groups differ significantly.

Figure 18 contains our experiments in pseudo-code.

### *4.4.4 Implications and external validity*

The experimental results indicate that a confidence-based comparison based on statistical tests identifies large ranges of steady similarity values as well as large ranges of list positions where the similarity of word pairs is meaningful. However, the results so far are exemplary to the model and text corpus used. In the next section we generalize our insights with further models trained on different corpora to find meaningful similarity values.

```
# 1. Step: Similarity value experiments.
Next_sig_diff_group_index := empty list;
Sim_value_group_indices := list of the indices //
                        // of the similarity value groups;
For i in Sim_value_group_indices:
    For j in Sim_value_group_indices:
        If  welch_test_p_value(LCH_G_i,LCH_G_j)<0.01:
            Next_sig_diff_group_index.append(j)
Plot.Next_sig_diff_group_index
# 2. Step: Similarity list experiments.
Next_sig_diff_group_index := empty list;
Sim_list_group_indices := list of the indices //
                         // of the similarity list groups;
For i in Sim_list_group_indices:
    For j in Sim_list_group_indices:
        If  welch_test_p_value(LCH_G_i,LCH_G_j)<0.01:
            Next_sig_diff_group_index.append(j)
Plot.Next_sig_diff_group_index
# 3. Step: Similarity list thresholds:
reverse() := list reverse function;
Sim_list_group_indices := list of the indices //
                         // of the similarity list groups;
Sig_higher_indices := empty list;
Not_sig_higher_indices := empty list;
For i in reverse(Sim_list_group_indices):
    If welch_test_p_value(LCH_G_100000,LCH_G_i)<0.01:
        Sig_higher_indices.append(i)
    If welch_test_p_value(LCH_G_100000,LCH_G_i)>0.01:
        Not_sig_higher_indices.append(i)
Threshold_below_no_sig_diff := first(Sig_higher_indices)
Threshold_above_sig_diff := last(Not_sig_higher_indices)
```

**Figure 18 Experiment procedure**

## 4.5 Generalization with Additional Corpora

The results from the prior subsection indicate that our approach to identify meaningful similarity values with a statistical test is promising. The results in Section 4.4.2 and 4.4.3 are already interesting for practitioners, as the corpus, embedding model (with these parameters), and the baseline are widely used. We now show that our approach yields meaningful results with other corpora as well.

### *4.5.1 Rationale behind the experiments*

With the model algorithm (e.g., SG or Glove model) and the parameters changing, the similarity values and lists change as well, cf. Section 3. This means that one must adjust the specific numbers that identify ranges where similarity is meaningful for any other model. To show that the procedure we propose is generally relevant we train two other models with different underlying corpora, but with the same model and parameter setting. To make the results of the experiments comparable we use corpora of the same size as before. If the results from this section (i.e., the plots) will be highly similar to those from Section 4.4, we will claim that our method to find meaningful similarity thresholds or list sizes is valid in general.

### *4.5.2 Experimental results*

The first dataset we train a model on is a Wikipedia dump with the articles shuffled and trimmed to contain approximately 1 billion words[2]. The second one is a 5-gram corpus extracted from the Google Books n-gram dataset [45]. We have extracted the 5-grams, shuffled them, and trimmed the data to have the same size as our original 1 billion word dataset. We note that working with 5-grams as the underlying corpus is slightly different from working with full text corpora. This is because of the limited size of the 5-grams, i.e., all the sentences considered by the learning algorithm only have a length of 5. We conduct the same experiments with the models trained on these corpora as in Section 4.4.

[2] Available at http://download.wikimedia.org/enwiki/

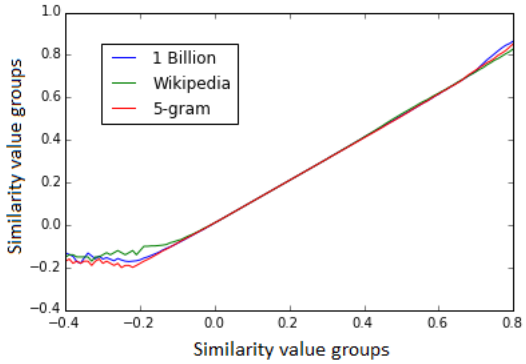


**Figure 19 Significantly different groups by similarity value**

Figure 19 shows that the results are almost identical to the ones in Section 4.4. The structure of the figures and even the values are very similar. For all three models, the similarity values which are not meaningful are between -0.4 and approximately -0.2.

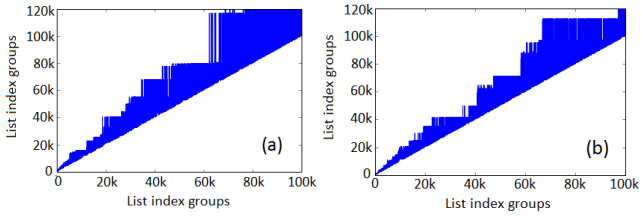


**Figure 20 a, b Significantly different groups by list indices for the models trained on 5-grams (Fig. 20a), Wikipedia (Fig. 20b)**

As for the similarity lists, we again see that the figures are very similar, but they naturally differ in the actual values. We also test the two models regarding the same questions we have asked earlier, namely: What is the last group which is significantly different in LCH similarity score from the last group overall? What is the first group that is not significantly different from the last group? The results are 28570 and 5889, respectively, for the model trained on the Wikipedia corpus and 35402 and 6408, respectively, for the model trained on the 5-grams. These numbers also are very much like the ones calculated before.

All this shows that our approach to derive those threshold values is fairly independent of the underlying corpus. The approach is applicable on any kind of corpus, and only the model selection and their parameters influence the resulting numbers.

## 4.6 Robustness of Evaluations Methods

The results of Section 4 so far indicate that meaningful ranges of similarity values exist. More specifically, for these values it is meaningful to compare two word pairs with different similarity values and to conclude that higher values imply greater semantic similarity. In contrast, the values outside of these regions are either very noisy, because of the lack of word pairs with the respective values, or indistinguishable in terms of similarity.

As the introduction has pointed out, evaluation methods compare word pair similarities on the full scale of similarity values and lists. Based on our results so far, we propose that the comparison should only be done at certain ranges of similarities. One can determine these ranges using the method proposed in Section 4.4. In particular, we propose that only those values should be compared which significantly differ in mean similarity scores, cf. Figure 15. For example, when evaluating the model in this section one should only compare word pair similarity values when the values are above -0.20. It is also noteworthy that every 0.01

difference in this range implies a significantly different similarity. For the list indices, similar conclusions are feasible. For example, with the model of this section we recommend to compare only indices below approximately 31500.

With other models, these values and indices could be different, but the method how to calculate them and the implications are the same. This means that for any embedding model we propose to calculate these values first, to improve any evaluation.

## 5. Conclusions

Word embedding models allow to quantify similarities of words. However, the notion of similarity and the meaning of similarity values has remained ambiguous. In this paper we have studied when exactly such values are meaningful in word embedding models. To this end, we have designed and conducted two series of experiments. With the first experiments we have shown how the distribution of similarity values change when changing the embedding-model algorithms or their parameters. As a result, we see that similarity values highly depend on the algorithms and parameters, i.e., the same value can represent different grades of similarity in different models. The second set of experiments has resulted in an evaluation method based on statistical tests, in order to find meaningful similarity values in embedding models. An important insight is that meaningful intervals of similarity values do exist, and one can actually find them for a specific embedding model. We have shown that these results are corpus-independent; they only depend on the learning algorithms and parameters already evaluated. Finally, we have proposed amendments to any evaluation method of word embedding models.

## 6. REFERENCES


[1] K. Erk, "Vector space models of word meaning and phrase meaning: A survey," *Language and Linguistics Compass*, vol. 6, pp. 635-653, 2012.

[2] S. Clark, "Vector space models of lexical meaning," *Handbook of Contemporary Semantic Theory, The*, pp. 493-522, 2013.

[3] M. Baroni, G. Dinu and G. Kruszewski, "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors.," in *ACL (1)*, 2014.

[4] S. Deerwester et al., "Indexing by latent semantic analysis," *Journal of the American society for information science*, vol. 41, p. 391, 1990.

[5] D. M. Blei, A. Y. Ng and M. I. Jordan, "Latent dirichlet allocation," *Journal of machine Learning research*, vol. 3, pp. 993-1022, 2003.

[6] Y. Bengioet al., "A neural probabilistic language model," *journal of machine learning research*, vol. 3, pp. 1137-1155, 2003.

[7] R. Collobertet al., "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493-2537, 2011.

[8] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008.

[9] E. H. Huanget al., "Improving word representations via global context and multiple word prototypes," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, 2012.

[10] J. Pennington, R. Socher and C. D. Manning, "Glove: Global Vectors for Word Representation.," in *EMNLP*, 2014.

[11] T. Mikolov et al., "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[12] T. Mikolov et al., "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013.

[13] A. Passos, V. Kumar and A. McCallum, "Lexicon infused phrase embeddings for named entity resolution," *arXiv preprint arXiv:1404.5367*, 2014.

[14] H. Komatsu, R. Tian, N. Okazaki and K. Inui, "Reducing Lexical Features in Parsing by Word Embeddings," 2015.

[15] W. Y. Wang and D. Yang, "That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using petpeeve tweets," in *EMNLP 2015, 2015*.

[16] B. Klein et al., "Fisher vectors derived from hybrid Gaussian-Laplacian mixture models for image annotation," *arXiv preprint arXiv:1411.7399*, 2014.

[17] J. Devlin et al., "Fast and Robust Neural Network Joint Models for Statistical Machine Translation.," in *ACL (1)*, 2014.

[18] I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014.

[19] S. Liu et al., "A Recursive Recurrent Neural Network for Statistical Machine Translation.," in *ACL (1)*, 2014.

[20] 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.

[21] R. Lebret and R. Collobert, "Rehabilitation of count-based models for word vector representations," in *International Conference on Intelligent Text Processing and Computational Linguistics*, 2015.

[22] F. Hill et al., "Not all neural embeddings are born equal," *arXiv preprint arXiv:1410.0718*, 2014.

[23] T. Schnabel et al., "Evaluation methods for unsupervised word embeddings," in *Proc. of EMNLP*, 2015.

[24] L. Finkelstein et al., "Placing search in context: The concept revisited," in *Proceedings of the 10th international conference on World Wide Web*, 2001.

[25] E. Bruni et al., "Distributional semantics in technicolor," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*,

2012.

[26] F. Hill, R. Reichart and A. Korhonen, "Simlex-999: Evaluating semantic models with (genuine) similarity estimation," *Computational Linguistics*, 2016.

[27] M. Batchkarov et al., "A critique of word similarity as a method for evaluating distributional semantic models," 2016.

[28] O. Avraham and Y. Goldberg, "Improving Reliability of Word Similarity Evaluation by Redesigning Annotation Task and Performance Measure," *arXiv preprint arXiv:1611.03641*, 2016.

[29] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, pp. 39-41, 1995.

[30] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in neural information processing systems*, 2014.

[31] T. Shi and Z. Liu, "Linking GloVe with word2vec," *arXiv preprint arXiv:1411.5595*, 2014.

[32] R. Rehurek and P. Sojka, "Software framework for topic modelling with large corpora," in *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 2010.

[33] Massey Jr, J. Frank, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, pp. 68-78, 1951.

[34] C. Chelba et al., "One billion word benchmark for measuring progress in statistical language modeling," *arXiv preprint arXiv:1312.3005*, 2013.

[35] B. Chiu, A. Korhonen and S. Pyysalo, "Intrinsic evaluation of word vectors fails to predict extrinsic performance," *ACL 2016*, p. 1, 2016.

[36] E. Altszyler, M. Sigman and D. F. Slezak, "Comparative study of LSA vs Word2vec embeddings in small corpora: a case study in dreams database," *arXiv preprint arXiv:1610.01520*, 2016.

[37] C.-C. Lin et al., "Unsupervised POS induction with word embeddings," *arXiv preprint arXiv:1503.06760*, 2015.

[38] C. Leacock and M. Chodorow, "Combining local context and WordNet similarity for word sense identification," *WordNet: An electronic lexical database*, vol. 49, pp. 265-283, 1998.

[39] A. Budanitsky and G. Hirst, "Evaluating wordnet-based measures of lexical semantic relatedness," *Computational Linguistics*, vol. 32, pp. 13-47, 2006.

[40] R. Mihalcea, C. Corley and C. Strapparava, "Corpus-based and knowledge-based measures of text semantic similarity," in *AAAI*, 2006.

[41] A. Budanitsky and G. Hirst, "Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures," in *Workshop on WordNet and Other Lexical Resources*, 2001.

[42] S. Bird, "NLTK: the natural language toolkit," in *Proceedings of the COLING/ACL on Interactive presentation sessions*, 2006.

[43] L. Meng, R. Huang and J. Gu, "A review of semantic similarity measures in wordnet," *International Journal of Hybrid Information Technology*, vol. 6, pp. 1-12, 2013.

[44] B. L. Welch, "The generalization ofstudent's' problem when several different population variances are involved," *Biometrika*, vol. 34, pp. 28-35, 1947.

[45] J.-B. Michel et al., "Quantitative analysis of culture using millions of digitized books," *science*, vol. 331, pp. 176-182, 2011.