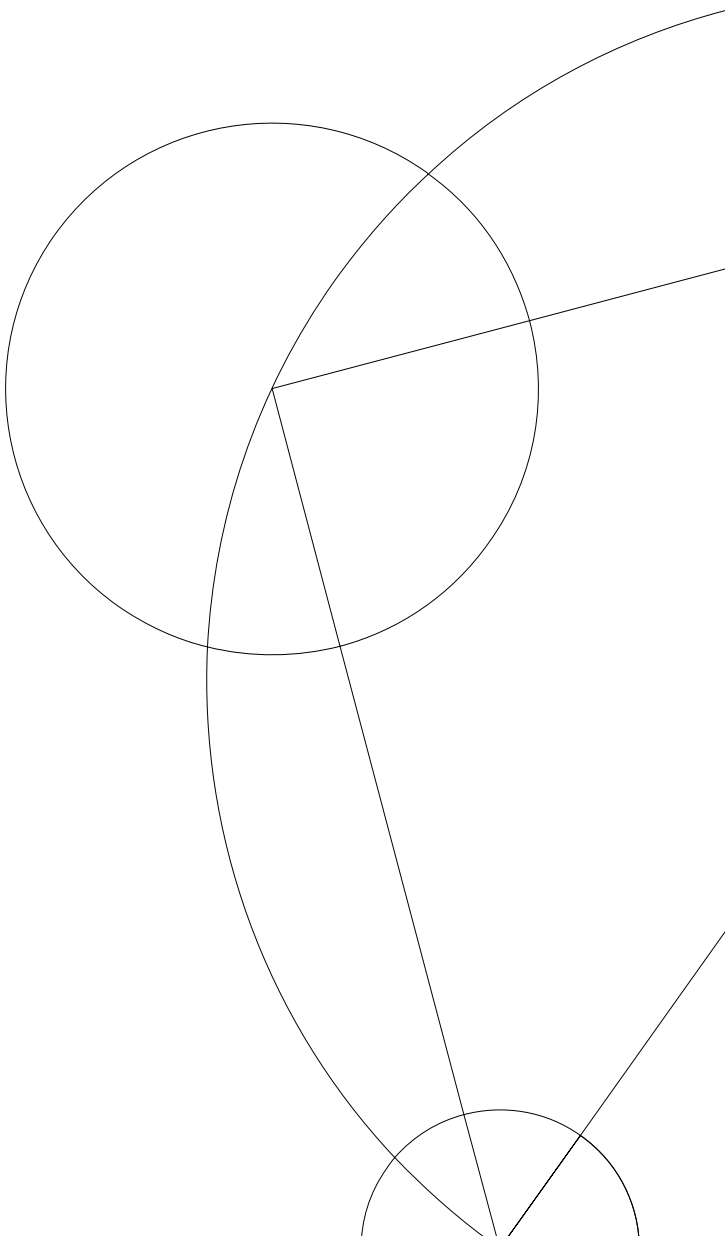
KØBENHAVNS UNIVERSITET

Bachelor thesis

Thor Steen Larsen (mvj665)

Phase Transitions In Word Embeddings

Department of Computer Science



Supervisor: Daniel Hershcovich

9. juli 2020

Abstract

Hershcovich et al. [2019] and Mckinney-Bock and Bedrick [2019] found anecdotally that word embedding of CBoW with context window size 1 judge similarity better than those trained with larger windows and performance improves gradually until it recovers. This bachelor thesis aims to research why this is happening, and what is changing in this phase transition. We reproduce Skip-gram and CBoW models with similar parameters on different context windows sizes and evaluate on the WS353 dataset. Furthermore, we look into relations between the most similar word pairs found by the word embedding and come up with hypothesis that semantically relations as sister terms can explain why model with narrow context windows perform better. We test our hypothesis on the SimLex dataset.
Models are created using Gensim's implementaion of the original Word2Vec model by Mikolov et al. [2013a], evaluated using Spearman's ρ and our hypothesis is tested using a hyper-geometric significance test.

Indhold

1 Theory

1.1 Context-predicting Language Models

By looking at natural language, we in general have a stream of words in sentences where a certain number of words fall into a certain. By calculating the probability that a word falls into a bin, we can from a frequentist point of view begin to understand word relations and derive a good statistical estimator of word context. That is the context of word relations for every word in the stream. Contextual information provides a good approximation to word meaning, since semantically similar words tend to have similar contextual distributions (ref).
In linguistics, this is formulated as the distributional hypothesis, that is words are used and occur in the same contexts tend to infer similar meanings [Harris, 1954]. In our coverage of the background of language model, we will not go further into this.
Distributional semantic modelling vector spaces make language model work efficiently on computers and make us able to use architectures such as Word2Vec. In our case we want to experiment with the predictive models using Word2Vec which given semantic, word vectors assigns weights to differently distributed sentences and tries to optimize the weights of these.
An alternative is a count-based models. using word n-gram, a sequence of words in a sentence $w_1, \dots w_n$, we are interested in the prediction task:

$$P(w_n|w_1 \dots w_{n-1}) = \frac{P(w_1 \dots w_n)}{P(w_1 \dots w_{n-1})}$$

where P is the probability function of w conditioned on the word n-gram w_{n-1}. We then calculate P and find the Maximum Likelihood Estimator. The parameters of the MLE will be the words with the highest probability of being in the context of the word of interest. Here we assume the Markov property that the probability of one word affects the probability of the next word.
Predictive models have though shown to be highly superior in deciphering semantic context than count-based methods due to the fact that simply counting the amount of word does not tell you much about semantic relations between words [Baroni et al., 2014].

1.2 CBoW

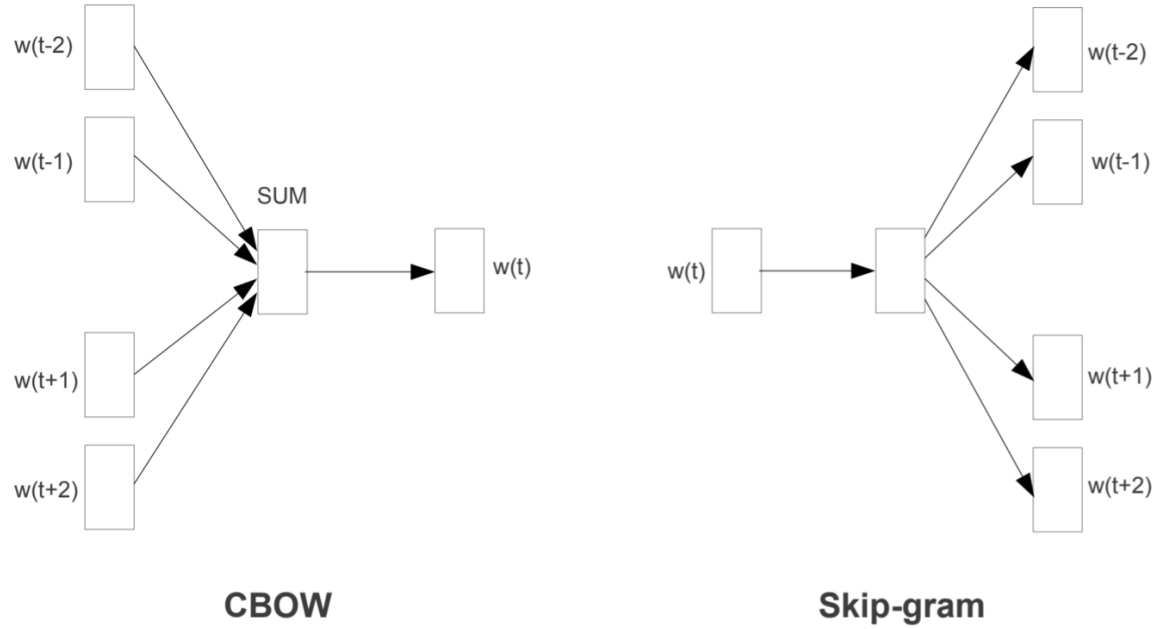


Fig 1: CBoW and Skip-gram architecture [Mikolov et al., 2013a]

In CBoW, we take some the ideas from the previous sub-section and use in a similar model to a neural network language model to learn the embedding of each word given its context. More precisely, we from the word's context $ww_{\text{window size}}$ want to predict the middle word. The basic CBoW architecture is as following.
As the input layer we have an one-hot encoded input context words represented as the vectors $\{\mathbf{x}_1, \ldots \mathbf{x}_C\}$ for a word of size C and vocabulary V depending on the size of the window and vocabulary.
The hidden layer is an N-dimensional vector $\mathbf{h}$ connected to the hidden layer via a $V \times N$ weight matrix $\mathbf{W}$ and the hidden layer is connected to the output layer via a $N \times N$ weight matrix $\mathbf{W'}$.
During forward propagation in the neural network $\mathbf{h}$ is computed by

$$\mathbf{h} = \frac{1}{C}\mathbf{W} \cdot (\sum_{i=1}^{C} \mathbf{x}_i)$$

which is the average of the input vectors weighted by the matrix $\mathbf{W}$. To compute the inputs to each node in the output layer

$$u_j = {v'_{w_j}}^T \cdot \mathbf{h}$$

where v'_{w_j} is the j'th of the output matrix $\mathbf{W}$'.The output y is calculated by passing the input $\mathbf{u}_j$ through the soft max function

$$y_j = p(w_{y_j}|w_1 \ldots, w_C) = \frac{\exp(u_j)}{\sum_{j'=1}^{V} \exp(u'_j)}$$

When we have learned the weight matrices $\mathbf{W}$ and $\mathbf{W}$', we back-propagate in the neural network by using negative logarithmic in an error function Mikolov et al. [2013a]. Similar to the MLE, the objective is to maximise the conditional probability of a

output word given a input context, therefore our loss function will be

$$\begin{aligned} E &= -\log p(w_O|w_I) \\ &= -u_{*j} - \log \sum_{j'=1}^{V} \exp(U_{j'}) \\ &= -\mathbf{v}_{w_O}^T \cdot \mathbf{h} - \log \sum_{j'=1}^{V} \exp(\mathbf{v}_{w_{j'}}^T \cdot \mathbf{h}) \end{aligned} \tag{1}$$

Where $*j$ is the index of the the actual output word, w_O is the middle output word and w_I is the previous/next input word. The next step is to derive the update equation for the hidden-output layer weights **W**', then derive the weights for the input-hidden layer weights **W** We then update the weights for the input and output hidden layer by using stochastic gradient descent or a similar method to optimize the weights. We can use the final output of a word vector $y \times \mathbf{W}$ through the softmax function to learn the probability of randomly picking a word x nearby any word in our vocabulary V.
In the predictive model, the words are averaged and the projection layer is shared for all the words. This is the reason why CBoW is a bag-of-words model as the order of words in the history does not influence the projection to the output layer and all the word vectors are averaged [Mikolov et al., 2013a]. This computation which happen in the hidden layer together with the objective is the main difference compared to the Skip-gram model.

1.3 Skip-gram

As in CBoW, we use a predictive model to obtain the word embedding. This time using a middle input word to infer the word's context.
The basic Skip-gram architecture is a mirror of the one in CBoW. Given a random word from the context of the window, we built up a data set of word pairs. We then calculate the probabilities of each word pair by feeding it to the hidden layer of a simple neural network in the form of a log-linear classifier with one layer per word in the vocabulary as features. From the hidden layer, we then obtain the weights which times our vector and through a softmax give us the same result as CBoW.
Or as put by the authors of Word2Vec, Mikolov et al. [2013a], we use each current word as an input to a log-linear classifier with continuous projection layer, and predict words within a certain range before and after the current word. We then use an error function and stocastic gradient decent as in the architecture of CBoW.
In addition to the general softmax function Mikolov et al. [2013b] implements a hierarchical softmax function to optimise the computation. This implementation uses a binary tree representation of the output layer with V words as its leaves and for each node the relative probabilities of its child nodes. This gives a random walk that assigns probabilities to the words. Furthermore, negative sampling can be used to decrease the computational cost [Mikolov et al., 2013b].

1.4 Windows size and context

One of the main ways distributional semantic models differ is in terms of the context window. The window size parameter in Word2Vec or size of the context window defines

the word pairs we use to calculate the probability of each word's context. This means that it has a critical role in determining how the model predicts the next word.
If we have a sentence: "The capital of Denmark is Copenhagen and the its second largest city is Aarhus". A W2V network with a window size 1 is going to learn that there is high probability between (Capital, of, Denmark, is, Copenhagen) and less probability between (Capital, of, Denmark, is, Aarhus). A model using the Skip-gram algorithm would use word pairs to learn this while a model with the CBoW algorithm would take the whole sentence / context to learn this.
This means that the size of the context window has a great effect on how our model evaluates related and similar words. Hershcovich et al. [2019] and Mckinney-Bock and Bedrick [2019] have shown in their experiments that there is significant difference between similarity scores of CBoW models with a window size of 1 and 2.
Mikolov et al. [2013a] found that increasing the windows size improves quality of the resulting word vectors, but it also increases the computational complexity, and that the most efficient context window was of size 10 in their case.

1.5 Cosine similarity

To calculate the similarity score between words, we use cosine similarity.
The cosine similarity between two vectors $\mathbf{A}$ and $\mathbf{B}$ can be calculated by

$$\begin{aligned} \mathbf{A} \cdot \mathbf{B} &= \| \mathbf{A} \| \| \mathbf{B} \| \cos\theta \\ \cos(\theta) &= \frac{\mathbf{A} \cdot \mathbf{B}}{\| \mathbf{A} \| \cdot \| \mathbf{B} \|} \end{aligned} \tag{2}$$

Cosine similarity captures the angle of the word vectors which mean that a high similarity score of 1 is equivalent to a 0 degree difference between the two vectors and none-similarity score of 0 means a 90 degree difference between the vectors. This make us able to avoid using the magnitude as by euclidean distance to compare the word vectors.

1.6 Spearman's rank correlation coefficient

To evaluate our models on a test data-set in AWS SageMaker and Gensim, we furthermore calculate Spearman's ρ of the word rankings. In our case the words are ranked by cosine similarity from high to low where the highest cosine similarity will be of the most similar word in the n-gram.
Spearman ρ can be used as a statistical test to determine if there exists a relation between two random variables X and Y. The test can be a bilateral test or a unilateral test [Dodge, 2008].
Spearman's ρ, where ρ is Pearson correlation coefficient, and $rank_X, rank_Y$ are the ranking of the two random variables can be written as

$$\rho_{rank_X, rank_Y} = \frac{cov(rank_X, rank_Y)}{\sigma_{rank_X}, \sigma_{rank_Y}}$$

Identical values are usually each assigned fractional ranks equal to the average of their positions in the ascending order of the values, which is equivalent to averaging over all possible permutations [Dodge, 2008].

1.7 Similarity

In linguistics similarity both refers to what we humans associate with being similar and similar words in terms of attributes such as hypernyms. Following is short description of semantics more generally and how the data set, we will use for reference are put together.

1.7.1 Semantic relations

There are several different kinds of semantic relations where the most common for similarity are Synonyms, Antonyms, Hypernyms.

1. Synonyms refer to words that have exactly the same meaning such as sick and ill.

2. Antonyms refer to words that have the opposite same meaning such as man and woman.

3. Hypernyms refer to words that the word refer to where root hypernym refers to the most basic hypernym such as entity or plant.

The latter is compromises of hyponyms which can be described as a type-of-relation of the word. For example, duck, crow and seagull are all hyponyms of bird while bird is a hypernym of duck, crow and seagull.
Words pairs with the same hypernyms can be described as sister terms. These words will naturally also be very similar, but may not be associated by humans. Examples of this is given in the following reference datasets.

1.7.2 WS353

WordSim3535 contains 353 word pairs, each associated with an average of 13 to 16 human judgements. In this case, both similarity and relatedness are annotated without any distinction on a scalar from 0 to 10. The Annotators were given the task to 'Assign a numerical similarity score between 0 and 10 (0 = words totally unrelated, 10 = words VERY closely related) ... when estimating similarity of antonyms, consider them "similar" i.e., belonging to the same domain or representing features of the same concept, not "dissimilar".' (ref).
As ... points out this results in many dissimilar word pairs receive a high rating and no associated but dissimilar concepts receive low ratings such as (coffee, cup) and (train, car) where the first pair is dissimilar but ? and the second pair is highly associated but ?. Furthermore, they point out there was a low annotator agreement (ibid.)
WS353 is tough the most used gold standard data set for similarity measures (ref).

1.7.3 Simlex

Simlex was made to correct some of the shortcomings for WS353. The data set is made of word pairs which conceptually are highly similar or related with a clear distinction between relatedness and association in rating by the annotator.
To create a test of the ability of models to capture similarity as opposed to association the dataset is based on the USF dataset for high word pairs with high similarity and a random selection of word pairs with high similarity put together. Such as words pairs

with same hypernyms and atonyms. In the Simlex dataset anntotators were introduced to similarity via the well-understood idea of synonymy, and in contrast to association: "In each case the participant was required to identify the most similar pair from a set of three options, all of which were associated, but only one of which was clearly similar (e.g. [bread, butter] [bread, toast] [stale, bread])"(ref). Furthermore, word pairs were not put in groups of context due to the fact that it introduces a high degree of subjectivity into the design process of the word pairs (ibid.).

1.8 Geometric distribution

The geometric distribution gives the probability that the first occurrence of success requires k independent trials, each with success probability p. If the probability of success on each trial is p, then the probability that the k'th trial (out of k trials) is the first success is

$$\Pr(X = k) = (1-p)^{k-1}p$$

The geometric distribution is an appropriate model if the phenomenon being modelled is a sequence of independent trials with only two possible outcomes for each trial, designated success or failure, the probability of success, p, is the same for every trial (ref).
If these conditions are true, then the geometric random variable X is the count of the number of failures before the first success. The possible number of failures before the first success is 0, 1, 2, 3, and so on.
The expected value of the geometric distribution can be derived as follows.

$$\begin{aligned} E(Y) &= \sum_{k=0}^{\infty}(1-p)^k p \cdot k \\ &= p\sum_{k=0}^{\infty}(1-p)^k k \\ &= p(1-p)\sum_{k=0}^{\infty}(1-p)^{k-1}\cdot k \\ &= p(1-p)\left[\frac{d}{dp}\left(-\sum_{k=0}^{\infty}(1-p)^k\right)\right] \\ &= p(1-p)\frac{d}{dp}\left(-\frac{1}{p}\right) = \frac{1-p}{p} \end{aligned} \tag{3}$$

1.8.1 Hyper-geometric distribution

To test our hypothesis, we use a special formulation of the geometric distribution called the hyper-geometric distribution which is also closely related to the binomial distribution. The probability mass function can be formulated as

$$p(k, M, n, N) = \frac{\binom{n}{k}\binom{M-n}{N-k}}{\binom{M}{N}} k \in [\max(0, N-M+n), \min(n, N)]$$

where M is the population size, N is the sample size, n is the number of successes in the population, X is number of drawn successes and the binomial coefficients are

defined as $\binom{n}{k} \equiv \frac{n!}{k!(n-k)!}$.
We find the variables in our Word2Vec model output which is specified in the method section.
Using a the survival function which is the inverse of the cumulative distribution function, we can then calculate the p value and determine if our findings are significant given a significance level α.
We assume that our samples are normal distributed and so that a standard normal distribution function Φ can describe our hyper-geometric distribution:

$$P(X \leq k) \approx \Phi\left(\frac{k - np}{\sqrt{np(1-p)}}\right)$$

2 Experimental setup

2.1 Description

We take a data driven inductive approach to finding a hypthesis that can explain the high performance of W2V using CBoW with a context window of size of 1 in relation to models with larger context windows, that is the phase transition from the smallest window size. We therefore want to recreate the setting where the phase transitions were found by Hershcovich et al. [2019] and Mckinney-Bock and Bedrick [2019]. To reproduce the experiment by Hershcovich et al. [2019], we need to create a series of W2V models with increasingly large context windows and evaluate their performance on the Ws353 dataset. To find a hypothesis about the observed, we can then examine the word embedding of the model and the semantics of the most similar word pairs rated by the model. This will hopefully give us an hypothesis that can explain the phase transitions.
To test our hypothesis, we can use the SimLex dataset and as with Ws353 evaluate the model and look into the word embedding to see if the same phenomena occurs.
We furthermore make sure our results are significant by a hyper-geometric test.
In the following, I will go the practical setup and method of the experiment.

2.2 Word2Vec Implementation

We use the BlazingText algorithm from Amazon SageMaker and a implementation of Word2Vec from Gensim.
BlazingText is a version optimised version of the Word2Vec Algorithm and can be used for embedding as well as text classification [Amazon, 2020].
Gensim's Word2Vec is based on the original C implementation by Mikolov et al. [2013a] of Word2Vec [Rehurek, 2019].
As described before, Word2vec models maps a corpus of words to high-quality distributed vectors. The resulting vector representation of a word is the word embedding. Using these embedding of a general corpus of words, we can find which words that are semantically similar in natural language put in sentences or other formats such as word pairs. The BlazingText and Gensim.Word2Vec are highly optimized for multi-core CPU architectures which makes it fast when given several million of words. Furthermore, they are both is able to provide word embedding from using either the Skip-gram (SGNS) or continuous bag-of-words (CBoW) algorithms (ibid.).

2.3 Dataset and pre-processing

Wikipedia is a excellent webpage for natural language in digital text format. Similarly to Hershcovich et al. [2019], we use fraction of all this text called text8 to create a corpus in our W2V models [Mahoney, 2011]. Pre-processing before Word2Vec-word embedding is very important when it comes to model performance. Babanejad et al (make ref) show in their analysis that model performance of word vectors models is improved by almost all pre-preprocessing techniques.
In text8 the following pre-processing steps are taken: The wiki dump which has been cleaned to words of the 27 character English alphabet containing only the letters a-z and nonconsecutive spaces. The goal of the authors of the data-set was to only retain text that normally would be visible when displayed on a Wikipedia web page and

read by a human. Therefore, only regular article text was retained, image captions are retained, but tables and links to foreign language versions were removed [Mahoney, 2011]. Citations, footnotes, and markup were removed. Hypertext links were converted to ordinary text, retaining only the visible anchor text. Numbers are spelled out so ”20”becomes ”two zero”etc.. Upper case letters are converted to lower case. Finally, all sequences of characters not in the range a-z are converted to a single space. The Perl script which was used to clean the text can be found here (url). Last but no least all words a tokenized for the w2V model.

2.4 Size of Context Window

We define the windows size to go from 1 - 25 in our models using either CBoW or Skip-gram. This give us a total of 50 different models.
The windows size determines what we feed into our model and model complexity due to the vectors direct relationsship with the size of the input word vectors.

2.5 Word2Vec parameters and execution

Beyond window size, we all need to decide on the dimension of the feature vectors and the minimum threshold of the words in the model.
Feature Vector Dimension is set to 500 and minimum frequency for words are set to 500 to get a similar results to Mckinney-Bock and Bedrick [2019].
Using BlazingText parameters were set similarly and in some instances lower.
The models are trained and evaluated in AWS SageMaker's cloud environment (url) and on a personal multi-core computer. Models, data and the used code written in Python can be found on GitHub-url.

2.6 Model evaluation

Each model is evaluated on WordSim353 for similarity and relatedness which is a data set of word which is human annotated [Gabrilovich, 2009]. The same data set and approach is used in Hershcovich et al. [2019] which we wish to compare with our own results.
In our evaluation as well of testing of the models, we can only use input and output to find out how the model performs.
Using Gensim we can calculate Spearman's α between similarity scores of our models and the human annotators. This is done using the WS353 dataset and SimLex dataset which were downloaded from their respective websites.

2.7 Exploratory data analysis of semantic relations

Using WordNet, we can find all the semantic relations of the word pairs of WS353 and see which relations are most similar by using cosine similarity between the word vectors of the model. WordNet is a database of synsets developed by Stanford University which has public python API (ref).
In our exploratory data analysis, all the word pair relation are investigated by retrieving all synsets from the WS353 word pairs evaluated by the model.This make us able to decide all the relations between the word pairs by making comparison of all lemmas,

hypernyms, hyponyms, antronyms etc..
When using the WordNet API, we use a py script implementing a similar method to Hershcovich et al. [2019] with the word pairs. The original script can be found on GitHub (GitHub link).

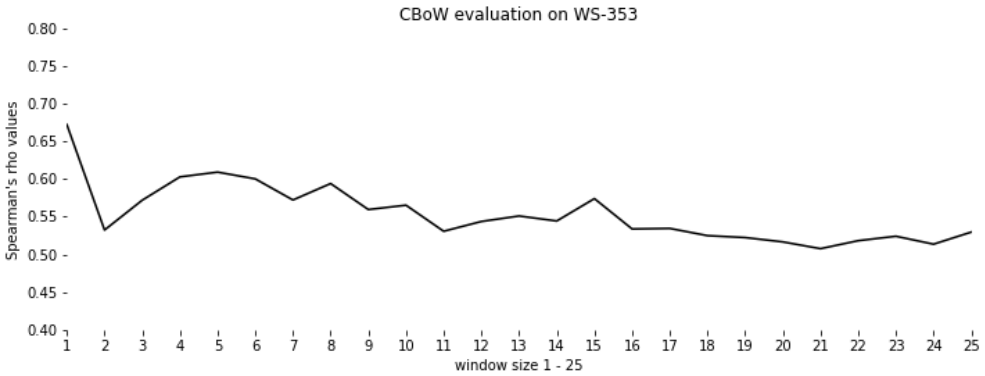
2.8 Significance test

We use Sci Py's implementation of the hyper-geometric test and its survival function to calculate p-values (SciPy Documentation. To test see if the occurrence of semantic relations are significant, we set X to be the occurrence of a relations between top-30 word pairs, M the total number of pairs, n the total number of relation occurrences and N to be count of word pairs in the top-30. We find the counts and test using a another script (Github link).
When testing our hypothesis, we use a standard significance level $\alpha = 0.05$ 0.05, and investigate our hypothesis for each model by $p < \alpha$, that is we determine if our findings are significant.

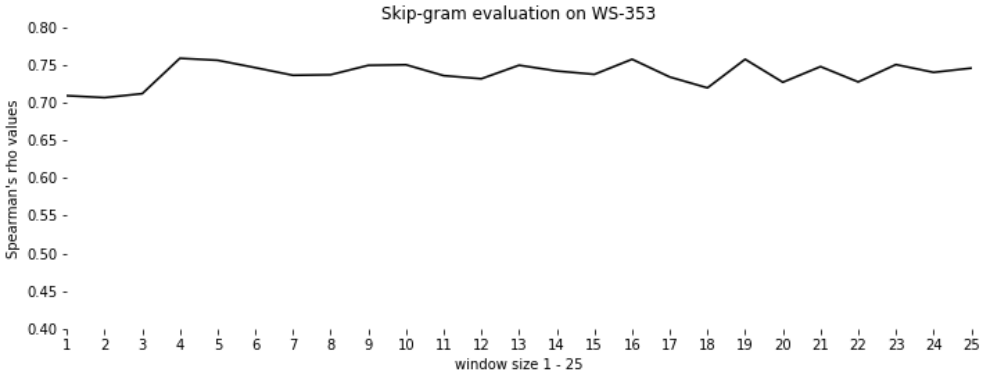
3 Preliminary results

3.1 Evaluation dataset WS353

Depending on the algorithm we use, we get different results when evaluating on WS-353. Using different parameters with BlazingText and a increasing window size, we are not able to reproduce the findings of Hershcovich et al. [2019] and Mckinney-Bock and Bedrick [2019].
When using the Word2Vec implementation found in Gensim by Radimrehurek, which is based on the original implementation by Mikolov et al. [2013a] in C, we are though able to reproduce the results.



As found in Hershcovich et al. [2019] the difference between window size 1 and 2 is not evident when using the Skip-gram algorithm.



We from here on concentrate on the W2V models made with the CBoW algortihm.

3.2 Exploratory data analysis of words pair relations

The top-10 ranked word pairs by the CBoW model with a context window size 1 are shown in tabel 1.
We see that the top-5 word pairs have a hyponym relation and several pairs have hypernym relation. This indicate that the wordpairs are each others hyponyms and reversely each others hypernyms and thereby that they are sister terms.
When looking at the relation of top-10 word pairs across models, we see a similar distribution of hyponyms. This could indicate that all models rate sister terms higher.

Tabel 1: Top-Ranked CBoW word-pairs and their relation

football	basketball	co_hyponyms
physics	chemistry	co_hyponyms
television	radio	co_hyponyms
rock	jazz	co_hyponyms
liquid	water	hyponyms
man	woman	antonyms
skin	eye	None
glass	metal	None
type	kind	hypernyms
planet	moon	None
king	queen	identity
food	fruit	None
money	cash	co_hyponyms
planet	sun	co_member_meronyms
psychology	science	hypernyms

Tabel 2: Top-10 word-pair relation

W1	**W2**	**W5**	**W10**	**W15**	**W25**
co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms
co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	None	co_hyponyms
co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	co_hyponyms
co_hyponyms	hyponyms	co_hyponyms	co_hyponyms	co_hyponyms	None
hyponyms	hypernyms	hypernyms	None	hyponyms	hyponyms
antonyms	co_hyponyms	hyponyms	hyponyms	antonyms	None
None	None	None	antonyms	co_hyponyms	None
None	antonyms	identity	identity	hypernyms	co_hyponyms
hypernyms	None	hypernyms	hypernyms	identity	antonyms
None	None	antonyms	None	hypernyms	identity

	hypernyms	**hyponyms**	**member meronyms**	**part meronyms**	**antonyms**
W1	0.87	0.01**	0.12	1.00	0.35
W2	0.97	0.01**	0.12	0.58	0.35
W5	0.97	0.04*	0.12	0.12	0.35
W6	0.87	0.12	0.12	0.58	0.35
W7	0.68	0.04	0.12	0.12	0.35
W8	0.97	0.01	0.58	0.12	0.35
W9	0.42	0.04	0.58	0.58	0.35
W10	0.87	0.04	0.58	0.12	0.35
W15	0.87	0.01	0.58	0.58	0.35
W25	0.97	0.44	0.35	0.58	0.35

Tabel 3: P-values from Hyper-geom Test. * indicate $p < 0.05$ and ** indicate $p < 0.1$. Decimals are rounded without loss of mutual precision.

3.3 Significance test of occurrence of words

To check if the word relations are significant, we calculate a p value for all relations in each model using the parameters described in the last section. In our program, we check for the relations: synonyms, hypernyms, hyponyms, member holonyms, member meronyms, part holonyms, part meronyms. substance holonyms, substance meronyms, antonyms, pertainyms and derivationally related forms.
In the tabel of p-values all, non-occuring word relation are removed.
We see that only the hyponym relation seem to be significant for the word pairs of the WS353 dataset which indicate that this relation is very important for the way the model measures similarity.
We see that this is true for models with a context windows of all sizes except 6, 17 and 25. which indicate that most models rate hyponym word pairs very similar across models with different window sizes. The most significant cases are for models with a context window size of 1 and 2.

3.4 Hypothesis

As shown in our preliminary results, we were able to reproduce the fact that CBoW models with a context window size of 1 models are better to find similarity between words than models with larger context windows.
When looking into into the word embedding and the semantic relations of the top-word pairs, we furthermore found that the words which were rated with a high similarity across models have a hyponym relation and thereby share many of the same hypernyms which indicate they could be sister terms. This finding was especially significant for models with a window size of 1 and 2 which mean that this could explain the phase transition.
Hyp_0: Word2Vec models with narrow context windows find that the most similar word pairs are sister terms.
Hyp_1: Word2Vec models with narrow context windows find that the most similar word pairs are not sister terms.
In following section, we will test this hypothesis on our test dataset, SimLex.

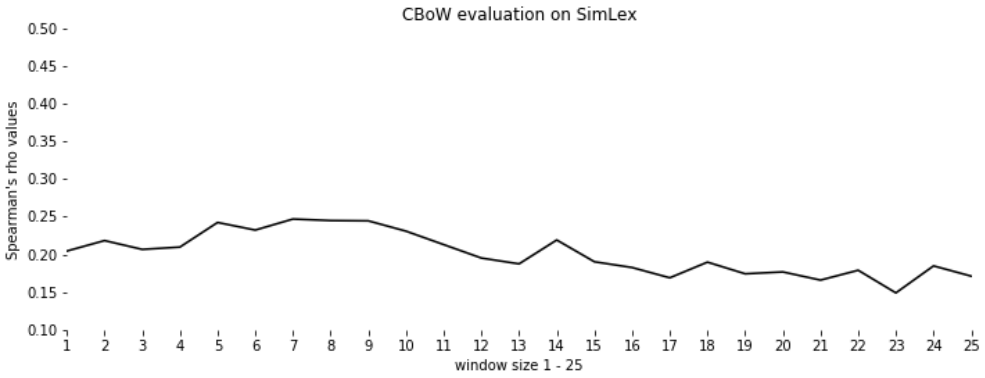
Tabel 4: Word pair relation top-10 SimLex

W1	**W2**	**W10**	**W15**	**W25**
hypernyms	hypernyms	hypernyms	hypernyms	hypernyms
identity	None	None	None	None
co_hyponyms	None	None	antonyms	antonyms
antonyms	antonyms	None	None	None
None	co_hyponyms	antonyms	None	None
None	co_hyponyms	identity	identity	None
None	None	identity	identity	None
antonyms	None	None	None	identity
co_hyponyms	None	co_hyponyms	None	co_hyponyms
similar_tos	hyponyms	None	indirect_part_holonyms	identity

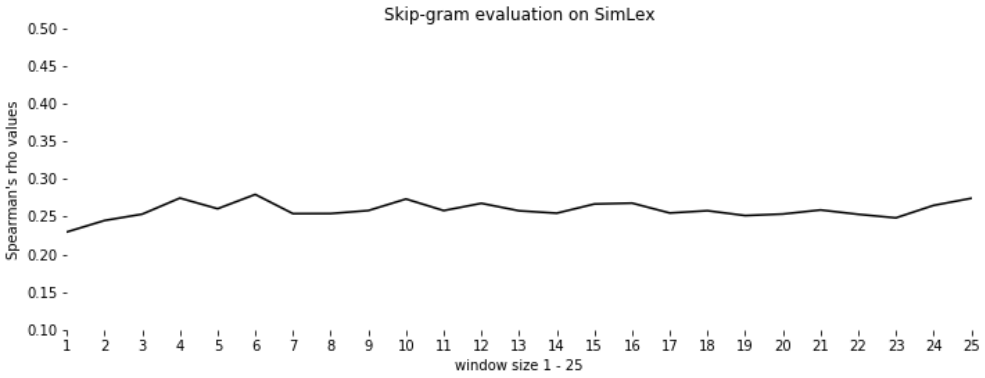
4 Results

4.1 Evaluation dataset SimLex

Running the same evaluation of our W2V model with the CBoW algorithm on SimLex as on WS353, we get a very different Spearmans ρ.



This could mean that, we will not see the same phase transition phenomena on the SimLex dataset as with the Ws353.



Using Skip-gram, we as before do not see the phase transition.

4.2 Hyper-geom test of word pair relations

We see in the relations of the top 10 pairs in tabel 4 that the most similar word pair relation are very different compared to those found in Ws353.

Tabel 5: P-values of word pair relations on SimLex. * indicate p <0.05 and ** indicate p <0.01. Rounding is done without loss of precision.

	synonyms	**hypernyms**	**hyponyms**	**member_meronyms**	**part_holonyms**	**substance_holonyms**	**antonyms**
W1	0.24	0.32	0.34	0.56	0.87	1.00	0.00**
W2	0.05	0.32	0.59	0.56	0.55	1.00	0.00**
W3	0.02	0.32	0.70	0.56	0.87	1.00	0.06
W4	0.60	0.32	0.46	0.56	0.87	1.00	0.02*
W5	0.41	0.65	0.80	0.56	0.55	0.34	0.02*
W6	0.41	0.65	0.88	0.56	0.87	0.34	0.02*
W7	0.90	0.48	0.59	0.56	0.55	0.34	0.02*
W8	0.60	0.65	0.59	0.56	0.55	0.34	0.02*
W9	0.77	0.48	0.23	0.56	0.55	0.34	0.06
W10	0.90	0.65	0.46	0.56	0.55	0.34	0.06
W11	0.96	0.48	0.46	0.11	0.55	0.34	0.06
W12	0.90	0.65	0.46	0.56	0.55	0.34	0.06
W13	0.77	0.48	0.34	0.56	0.87	0.34	0.06
W14	0.96	0.32	0.34	0.11	0.87	0.34	0.06
W15	0.90	0.19	0.46	0.56	0.87	0.34	0.06
W16	0.90	0.79	0.46	0.56	0.55	0.34	0.06
W17	0.90	0.19	0.34	0.56	0.55	0.34	0.06
W18	0.90	0.32	0.34	0.56	0.55	0.34	0.15
W19	0.96	0.32	0.34	0.56	0.87	0.34	0.06
W20	0.90	0.65	0.46	0.56	0.55	0.34	0.02*
W21	0.90	0.79	0.15	0.56	0.55	0.34	0.06
W22	0.90	0.48	0.46	0.56	0.87	0.34	0.06
W23	0.77	0.19	0.46	0.56	0.55	0.34	0.06
W24	0.96	0.19	0.46	0.56	0.55	0.34	0.06
W25	0.96	0.65	0.23	0.56	0.87	0.34	0.06

When make a hyper-geom test, we also get a very different result. In this test we use an X = 90 because the dataset is bigger than that of Ws353. Similar reuslts are found using a X = 30 though. Again, all relations are tested and non-occuring relations are not included in the tabel (tabel 5). We see from the table that we cannot confirm our hypothesis on the SimLex dataset because the occurence of hypnyms and hypernyms is very insignificant.
The occurence of antonyms is the only significant relation.

5 Appendix

Litteratur

Amazon. Blazing documentation, 2020. URL https://docs.aws.amazon.com/sagemaker/latest/dg/blazingtext.html.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1023. URL https://www.aclweb.org/anthology/P14-1023.

Yadolah. Dodge. *The Concise Encyclopedia of Statistics*. Springer reference. Springer New York, New York, NY, 1st. ed. edition, 2008. ISBN 0-387-32833-5.

E. Gabrilovich. Wordsim353 - similarity and relatedness, 2009. URL http://alfonseca.org/eng/research/wordsim353.html.

Zellig S. Harris. Distributional structure. WORD, 10(2-3):146–162, 1954. doi: 10.1080/00437956.1954.11659520. URL https://doi.org/10.1080/00437956.1954.11659520.

Daniel Hershcovich, Assaf Toledo, Alon Halfon, and Noam Slonim. Syntactic interchangeability in word embedding models. pages 70–76, 2019. URL https://www.aclweb.org/anthology/W19-2009.

Matt Mahoney. About the test data, 9 2011. URL http://mattmahoney.net/dc/textdata.html. text8 can found on http://mattmahoney.net/dc/text8.zip.

Katy Mckinney-Bock and Steven Bedrick. Classification of semantic paraphasias: Optimization of a word embedding model. *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for*, 2019. doi: 10.18653/v1/w19-2007.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv.org*, 2013a. URL http://search.proquest.com/docview/2086087644/.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. *arXiv.org*, 2013b. URL http://search.proquest.com/docview/2085905727/.

Radim Rehurek. Gensim.word2vec documentation, 2019. URL https://radimrehurek.com/gensim/models/word2vec.html.