
Albert Ludwig University of Freiburg

Master's Thesis
Mathematics

# Neural Word Embeddings as Matrix Factorization

*Author:*
Theresa Klumpp

*Supervised by:*
Prof. P. Pfaffelhuber
Prof. H. Bast

Submission by 17.12.2019

# Contents

# Abstract

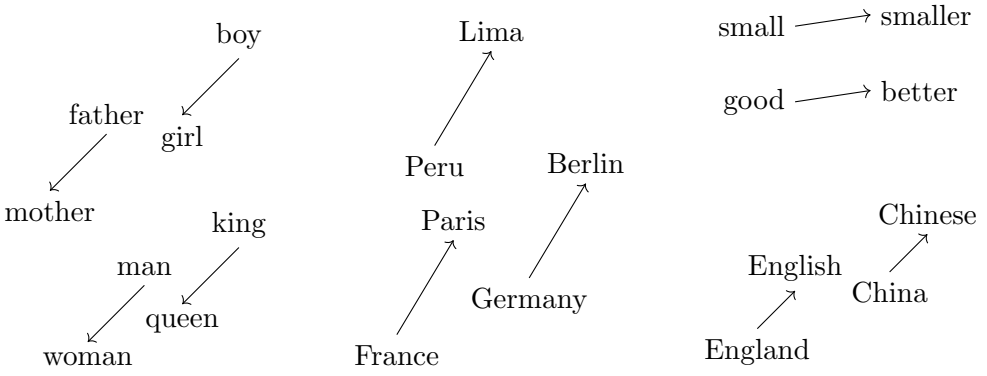
The problem of finding continuous word vectors is central and well studied in natural language processing (NLP). In this thesis, we consider two neural networks that yield low-dimensional word embeddings: skip-gram and skip-gram with negative sampling (SGNS), introduced by Mikolov et al. in [10] and [11]. Assuming the dimension of these word embeddings was large (at least as large as the size of the vocabulary), we show that a different method, namely the factorization of a word-context matrix using singular value decomposition (SVD), also maximizes the objective function of the neural network. Even though we cannot make this link for low-dimensional embeddings, we can still use SVD and show that this method performs even better in some linguistic tasks than the neural network that it stems from. This connection between the SGNS neural network and SVD was presented by Levy and Goldberg in [8]. We analyze the cosine similarity, commonly used in NLP to measure the similarity between word vectors, and we derive a formula that gives us a heuristic understanding of what 'close' in $\mathbb{R}^d$ is. We compare the different methods for finding low-dimensional word vectors and evaluate them on word similarity and semantic and syntactic analogy tasks. Our data clearly shows that SVD prefers a low number of 'negative samples', whereas the SGNS neural network generally performs better for a higher number. Furthermore, the neural network is evidently superior in the analogy tasks, while SVD gains the advantage on word similarity tasks, if the number of negative samples is low.

# Zusammenfassung

Das Problem, stetige Wortvektoren zu finden, ist in der Computerlinguistik (CL) zentral und gut erforscht. In dieser Arbeit betrachten wir zwei neuronale Netze, die niedrigdimensionale Word-Embeddings liefern: Skip-Gram und Skip-Gram mit Negative-Sampling (SGNS), vorgestellt von Mikolov et al. in [10] und [11]. Ausgehend von der Annahme, dass die Dimension dieser Word-Embeddings groß wäre (mindestens so groß wie die Größe des Vokabulars), zeigen wir, dass ein anderes Verfahren, nämlich das Faktorisieren einer Wort-Kontext-Matrix mithilfe von Singulärwertzerlegung (SWZ), auch die Zielfunktion des neuronalen Netzes maximiert. Obwohl wir diese Verknüpfung für niedrigdimensionale Embeddings nicht machen können, können wir dennoch SWZ anwenden und zeigen, dass diese Methode in manchen Linguistik-Aufgabenstellungen sogar besser funktioniert als das neuronale Netz, von dem sie abstammt. Dieser Zusammenhang zwischen dem neuronalen SGNS-Netz und der SWZ wurde von Goldberg und Levy in [8] dargelegt. Wir untersuchen die Kosinus-Ähnlichkeit, die üblicherweise in der CL verwendet wird, um die Ähnlichkeit zwischen Wortvektoren zu messen, und wir leiten eine Formel her, die uns ein heuristisches Verständnis dafür gibt, was 'nah' in $\mathbb{R}^d$ ist. Wir vergleichen die verschiedenen Verfahren zur Findung von niedrigdimensionalen Wortvektoren und werten sie anhand von Aufgabenstellungen zu Wortähnlichkeit und semantischen und syntaktischen Analogien aus. Unsere Daten zeigen deutlich, dass SWZ wenige 'Negative-Samples' bevorzugt, wohingegen das neuronale SGNS-Netz allgemein für eine höhere Anzahl besser funktioniert. Desweiteren ist das neuronale Netz in den Analogie-Aufgaben klar überlegen, während SWZ in den Aufgaben zur Wortähnlichkeit besser abschneidet, wenn die Anzahl an Negative-Samples klein ist.

# 1 Introduction

This thesis is concerned with word embeddings or word vectors. Since words themselves are simply combinations of letters, the spelling usually does not correlate with the meaning of the word. For example, the letters in 'desert' and 'dessert' are almost the same, but their meanings are very different. On the contrary, 'automobile' and 'car' consist of very different letters, even though they are synonyms. Particularly, words do not have a euclidean structure, so we desire to represent words by vectors in $\mathbb{R}^d$ that reflect similarities and dissimilarities. In our example, vec('automobile') and vec('car') should be closer together than vec('desert') and vec('dessert'). We will see that our word embeddings even represent relationships between the words (see Figure 1.1).




(a) man-woman (b) capitals (c) syntactic relations

Figure 1.1: Examples of various relations between words

These word representations are important, for instance, for neural networks that use words as input. In this thesis, we will compare and analyze different methods to compute such word embeddings.

## 1.1 Notation and Background

Our models will make use of the distributional hypothesis that words in similar contexts have similar meanings: "You shall know a word by the company it keeps!" (J.R. Firth, British linguist in 1957). Before we go into more detail, we will give a definition of "context" and introduce some notation.

The underlying data for the network comes from a large text $w(1), w(2), \ldots, w(t)$, usually containing billions of words. The *word vocabulary*, denoted by $V_W$, is the set of all distinct words from this text. We number the words in the vocabulary and denote the $i$-th word as $w_i$. We then define the *context* for a specific occurrence of a word to be the terms surrounding it in a window of size $L$.

*Example.* Consider the following example sentence.

`Everybody wants to visit sunny Freiburg.`

Using a window of size 2 to each side yields the following word-context pairs.

- (everybody, wants), (everybody, to)
- (wants, everybody), (wants, to), (wants, visit)
- (to, everybody), (to, wants), (to, visit), (to, sunny)
- (visit, wants), (visit, to), (visit, sunny), (visit, Freiburg)
- (sunny, to), (sunny, visit), (sunny, Freiburg)
- (Freiburg, visit), (Freiburg, sunny)

This way, we retrieve word-context pairs $(w_i, c_j)$ from the text. Our data will be the collection of these observed word-context pairs, which we denote by $D$.

It is important to mention that other definitions of 'context' are possible. We denote the context vocabulary as $V_C$. Our definition of context implies that $V_C = V_W$, but this might not be true for a different definition of context. Nevertheless, we will keep the distinction between the two vocabularies, since it is always important to know whether a term has the role of a word or a context and we will obtain different embeddings for the words and the contexts.

We denote the number of times a word-context pair $(w, c)$ appears in $D$ as $\#(w, c)$. Likewise, we denote the number of times $w$ and $c$ appear in $D$ as $\#(w)$ and $\#(c)$ respectively, i.e. $\#(w) = \sum_{c' \in V_C} \#(w, c')$ and $\#(c) = \sum_{w' \in V_W} \#(w', c)$.

Furthermore, we denote a word embedding for a word $w$ as $\vec{w}$ and similarly, we write $\vec{c}$ for a context embedding of $c$.

In 2013, Mikolov et al. introduced *skip-gram* [10], a neural network that computes low-dimensional, dense word vectors. It is often easy to get quite a good understanding of the meaning of a word by its context, even if one has never heard the word before. Mikolov et al. employed this fact in their neural network. The input is a one-hot encoded word, that is a $|V_W|$-dimensional sparse vector with a single 1 at position $i$ to represent the $i$-th word in the vocabulary $V_W$. Then, the skip-gram network makes a prediction about the terms that appear in this word's context. However, this method is greatly expensive computationally. Therefore, Mikolov et al. presented *negative sampling* [11], an extension to improve training speed and the quality of the vectors.

## 1.2 Contributions

The key contributions of this thesis are:

- Gaining an understanding of the objective functions of skip-gram and SGNS and the statistical models behind them.

- Finding a maximum for skip-gram's objective. Goldberg and Levy showed in [8] that SGNS has a maximum for

$$\vec{w} \cdot \vec{c} = \log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c) \cdot k}\right),$$

where $k$ is the number of 'negative samples' we choose. The proof for skip-gram is more difficult, because the objective cannot be written as a sum of one-dimensional functions. We will show the negative definiteness of the Hessian matrix and we will identify a local maximum for

$$\vec{w} \cdot \vec{c} = \log \#(w,c).$$

- Showing the connection between the neural networks and SVD: both maximize the same function if the dimension is large enough.

- Proving that the angle between two vectors provides a metric on the sphere. We will use a different distance, the cosine distance, to compare word vectors. However, we will show that in our case both lead to the same results, which justifies the use of the cosine distance even though it is not a metric itself.

- We find a formula for the expectation of the distance of the closest vector assuming uniform distribution on the sphere. This gives us an understanding of what 'near' for vectors in $\mathbb{R}^d$ means, or rather what 'similar' for our words means.

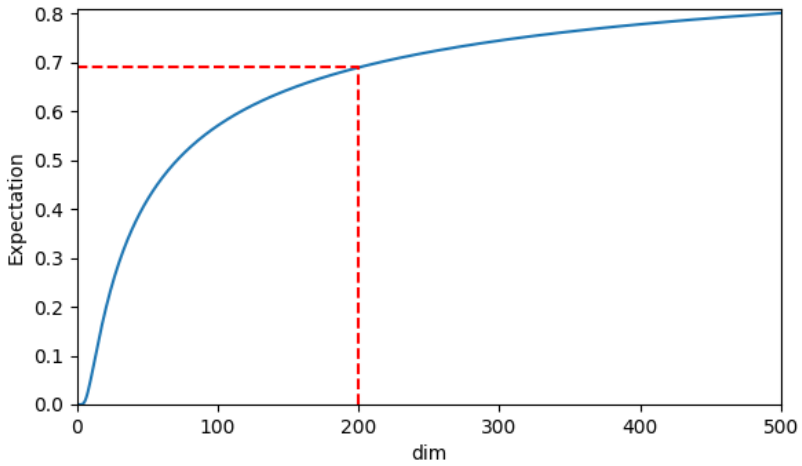



Figure 1.2: Expectation of the cosine distance to the nearest vector for 159,862 vectors depending on the embedding dimension.

The models we evaluate in Chapter 7 are based on a vocabulary of 159,862 words and we chose the embedding dimension 200 for most of our models. As we can see in Figure 1.2, for any query, we expect a word to be as close as about 0.69. Any words closer than this can be considered relevant to our query.

- An implementation of the SGNS neural network and the SVD variant for both skip-gram and SGNS. The models are accessible as a web app[1].

[1] `http://word2vec.cs.uni-freiburg.de/`

- Reproduction and understanding of the qualitative behavior of the objective functions. We will look at the performance of different models as $k$ changes (similar to Table 1 in [8]). When it comes to maximizing the objective, the neural networks benefit from a higher number of negative samples ($k$), while SVD suffers from it. Furthermore, we study the column denoted by SPPMI in [8], since our values differ from Levy and Goldberg's values.
- Evaluation of our models on word similarity and analogy tasks. All our experiments indicate that SVD behaves best for low values of $k$, the number of negative samples. The neural network seems to perform better the more negative samples we have. However, our data points to the fact that the quality of the vectors starts to decrease if $k$ gets too big, as we observe a decrease in performance for $k = 15$ in some experiments. While neither the network nor the SVD model was clearly superior on the word similarity tasks, the SGNS network gains the advantage on the analogy experiments.

# 2 Related Work

In this chapter, we will give a brief overview of the history of word representations and of important work that has been done in the field.

## 2.1 Distributional hypothesis and count-based models

> "You shall know a word by the company it keeps."

This famous quote by J.R Firth (1957) summarizes the distributional hypothesis, which became the basis of many word representations methods. The distributional hypothesis originated in linguistics in the 1950s [6] and the main idea is that words in similar contexts have similar meanings. The more semantically similar two words are, the more similar they will be distributed and thus the more they will tend to occur in similar linguistic contexts.

There is a variety of count-based models which use this hypothesis to retrieve word representations [9, 15, 7]. They mostly use a word-word co-occurrence matrix such that the entry $M_{ij}$ gives some measure of association between the $i$-th and the $j$-the word, for instance the number of times they appear in the context together. We then assign each word the corresponding row, column or a combination of the two (for instance their sum or concatenation) to get a set of word representations. The more similar the contexts of two words are, the more similar the vectors will be and by the distributional hypothesis, these words will be semantically similar. Since the resulting word representations are high-dimensional and sparse, it is common to use methods like Singular Value Decomposition (SVD) to reduce the rank of $M$, yielding low-dimensional, dense word representations.

## 2.2 Neural-Network based Methods

Since the early 2000s, different approaches for finding word representations using neural networks have been developed [1, 13]. These representations are referred to as 'word embeddings'. In this section, we will look at two more recent models.

**Word2Vec** was invented by Mikolov and colleagues and presented in 2013 in [10]. They introduce two similar model architectures: continuous bag-of-words (CBOW) and skip-gram. Both are neural networks with a single hidden layer. While CBOW predicts the current word from the words in its context, the skip-gram model predicts the context words from the current word (see Figure 2.1). This way, both models learn similar vectors for similar words. But more surprisingly, they even reflect linear relationships between the word vectors and are therefore able to capture word analogies (see Figure 1.1). With the extension 'negative sampling', which was presented in [11], the skip-gram network is efficient to train and is among the state-of-the-art methods for word-embeddings.

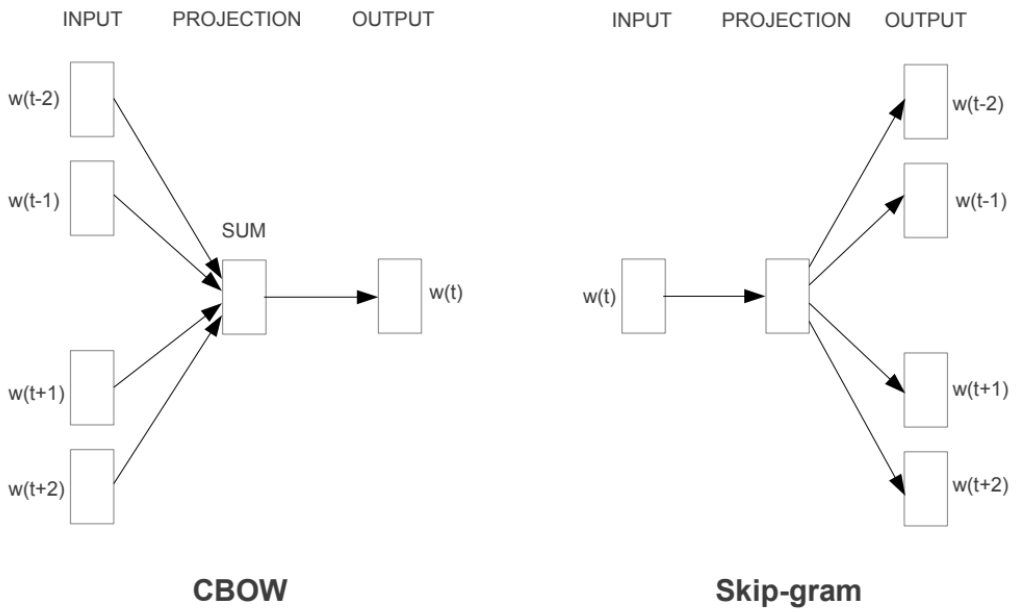



Figure 2.1: The CBOW and skip-gram architecture (Figure from [10])

We will present a more detailed description of the skip-gram model and its extension 'negative sampling' in chapter 3 of this thesis.

Word2Vec is highly popular among NLP researchers. In this thesis, we will focus on one of the follow-up papers written by Levy and Goldberg in 2014 [8]. They show that skip-gram with negative sampling is implicitly factorizing a co-occurrence matrix.

**The Global Vector model (GloVe)** is another word embedding method, which was introduced by Pennington et al. in 2014 [14], a year after Word2Vec was published. It tries to combine the advantages of count-based models, which are efficient usage of statistics and fast training, and the advantages of prediction-based models in capturing more complex structures like word analogies. The GloVe model focuses on word-word co-occurrences over the *entire corpus* (hence the name *global* vectors). This model turns out to be very good with rare words and performs well on word similarity and analogy tasks.

# 3 The neural networks

We think of two words as being similar if they often appear in the same context. "If A and B have almost identical environments [...], we say they are synonyms" (Zellig Harris, 1954 [6]). For example, the words 'big' and 'large' would probably have similar contexts. Thus, our goal is to use this distributional hypothesis to find vectors such that two words that often appear in similar contexts will be likely to have similar embeddings.

Assume we have word and context embeddings such that the dot products of frequent word-context pairs (that is pairs with large $\#(w, c)$) are high while the dot products of infrequent pairs are small. This means that embeddings of words that appear in similar contexts will have similar dot products with most context vectors. Since a $d$-dimensional vector is uniquely determined by its dot product with $d$ linearly independent vectors, words that appear in similar contexts will be likely to have similar embeddings.

Therefore, we want to find a function that is maximized when the dot products for frequent word-context pairs are large while the dot products for infrequent pairs are small. Even though we might be able to think of such equations, it is not obvious which of them will lead to good word embeddings.

In the following two sections, we will present two neural networks with objective functions that have this desired property. We will see in chapter 7 that they do indeed lead to good results.

## 3.1 Skip-gram

Skip-gram is a classification network whose input is a word $w_i$ and whose output is a probability distribution over the context words. Assume we pick a term from the context of $w_i$ at random. For each context in our vocabulary, the network will predict the probability of it being the term we chose at random. We denote this probability by $\mathrm{P}(c \mid w) = \frac{\#(w,c)}{\#(w)}$.

In more detail, for a single word-context pair $(w_i, c_j) \in D$, the input $w_i$ is given using one-hot representation. That is a $|V_W|$-dimensional vector with a single one at the $i$-th component and zeros otherwise to represent the $i$-th word in the vocabulary. In the same way, the label for this specific pair is a $|V_C|$-dimensional vector with a single one at the $j$-th component. The network will learn the probabilities $\mathrm{P}(c \mid w)$ from the statistics of our sample collection $D$. For example, for the input word 'Freiburg', the output probabilities for 'Breisgau' or 'university' will be much higher than for terms like 'elephant' or 'desert' after training. This is because the word-context pairs (Freiburg, Breisgau) and (Freiburg, university) are found more often in the collection $D$ than the pairs (Freiburg, elephant) or (Freiburg, desert). We are not primarily interested in achieving high accuracy on predicting the context words, but rather in the weights learned by the network.

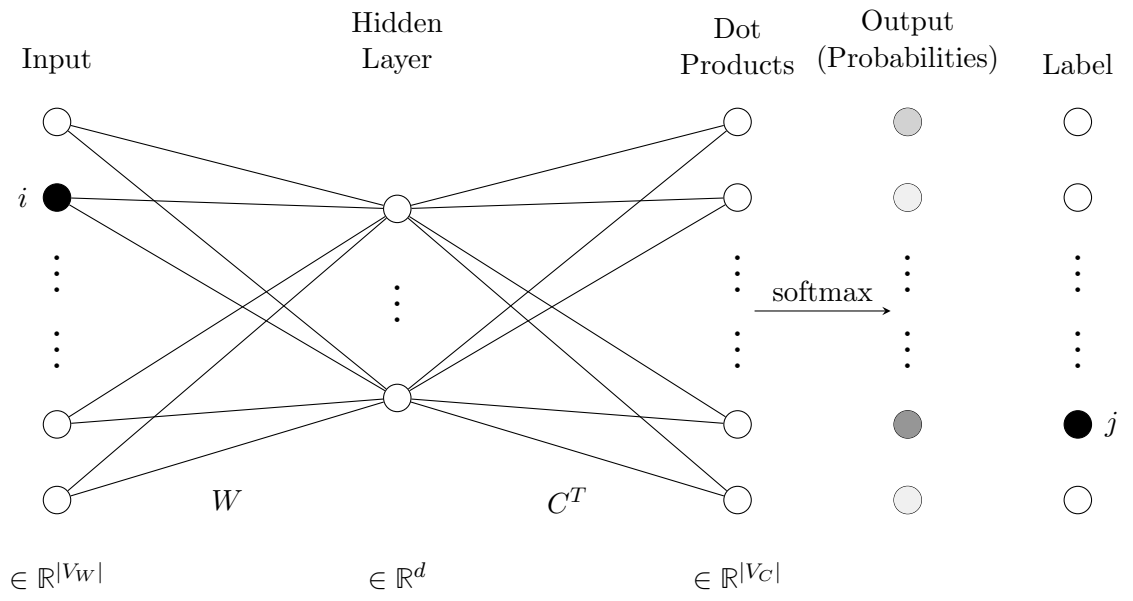



Figure 3.1: Skip-gram neural network for a single word-context pair $(w_i, c_j)$

We use two matrices, $W \in \mathrm{Mat}_{|V_W| \times d}$ and $C \in \mathrm{Mat}_{|V_C| \times d}$ for the linear transformations between the layers (see Figure 3.1). Here, $d$, the embedding dimension, is a parameter we choose. Picking a smaller value for $d$ will result in decreasing precision but improved computation time and working memory. After training, we will use the $i$-th row of $W$ to represent $w_i$, i.e.

$$\vec{w}_i := W_i$$

and similarly

$$\vec{c}_j := C_j.$$

Since the input of the network is a one-hot (row) vector, multiplication by $W$ results in the $i$-th row of $W$, that is the word representation for $w_i$. Multiplying this by $C^T$, the transpose of $C$, yields a vector containing the dot products of $w_i$ with each of the $c_k$. To obtain a probability distribution, we then apply the softmax function $\sigma : \mathbb{R}^n \to \mathbb{R}^n$, which is given by

$$\sigma\left(x\right)_j = \frac{\exp\left(x_j\right)}{\sum_{k=1}^{n} \exp\left(x_k\right)}.$$

Hence, the conditional probability $\mathrm{P}\left(c \mid w\right)$ resulting from this network is given by

$$\mathrm{P}_{W,C}\left(c \mid w\right) = \frac{\exp\left(\vec{w} \cdot \vec{c}\right)}{\sum_{c' \in V_C} \exp\left(\vec{w} \cdot \vec{c'}\right)}. \tag{3.1}$$

$W$ and $C$ are the word and context representation matrices as explained above.

Using gradient descent, skip-gram tries to minimize the log loss, or equivalently maxi-

mize

$$\begin{aligned}\ell_{\mathrm{SG}}(W, C) &= \sum_{(w,c)\in D} \log \mathrm{P}_{W,C}(c \mid w) \\ &= \sum_{(w,c)\in D} \left( \vec{w} \cdot \vec{c} - \log \left( \sum_{c' \in V_C} \exp\left( \vec{w} \cdot \vec{c'} \right) \right) \right). \end{aligned} \tag{3.2}$$

Observe that for a single word-context pair $(w_i, c_j)$, the network will maximize the dot product of $\vec{w_i}$ and $\vec{c_j}$ while minimizing the dot products of $\vec{w_i}$ with any other context vector $\vec{c_l}$, where $l \neq j$. The more frequent a pair $(w, c)$ is, the more often it will do this. Hence, this function satisfies the desired property that it is maximized when the dot products for frequent word-context pairs are large while the dot products for infrequent pairs are small.

## 3.2 Skip-gram with negative sampling (SGNS)

When training on a large dataset, there is a large number of entries in the matrices, i.e. weights in the network, to be updated during backpropagation. The matrices $W$ and $C$ have $d \cdot |V_W|$ and $d \cdot |V_C|$ entries, respectively. We are interested in making changes that reduce the number of weights which determine an entry in the output layer. Negative sampling is a modification in which the loss function only depends on $(k + 2) \cdot d$ weights for each sample, where $k$ can be chosen and is usually between 1 and 20, compared to $(|V_C| + 1) \cdot d$ weights in the original skip-gram model.

The network is very similar to skip-gram. The main change in the architecture is that we apply the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$ pointwise as an activation function instead of softmax (see Figure 3.2). Note that this means that the values in the output layer do not form a probability distribution on the context vocabulary, meaning they do not sum up to 1. In this case, a single value in the output layer is independent from all but one row of $C$. In contrast, in the skip-gram model, a value depends on *every* entry of $C$. This gives us the improvement that makes training feasible.

When training on a single word-context pair $(w_i, c_j) \in D$ in the SGNS-model, we choose $k$ contexts $c_{j_1}, \cdots, c_{j_k}$ at random, according to their occurrence in $D$, that is according to the probability distribution $\mathrm{P_D}(c) = \frac{\#(c)}{|D|}$. The assumption is that randomly picking contexts will be likely to result in a 'negative sample', meaning that the pair will be an unobserved pair, that is $\#(w, c)$ will be 0 or a small number. The labels are 1 at position $j$ and 0 at positions $j_1, \ldots, j_k$ (see Figure 3.2). Using log-loss again, the SGNS network is optimized with respect to

$$\begin{aligned}&\log \sigma(\vec{w_i} \cdot \vec{c_j}) + \sum_{l=1}^{k} \log\left(1 - \sigma(\vec{w_i} \cdot \vec{c}_{j_l})\right) \\ =& \log \sigma(\vec{w_i} \cdot \vec{c_j}) + \sum_{l=1}^{k} \log \sigma(-\vec{w_i} \cdot \vec{c}_{j_l}). \end{aligned} \tag{3.3}$$

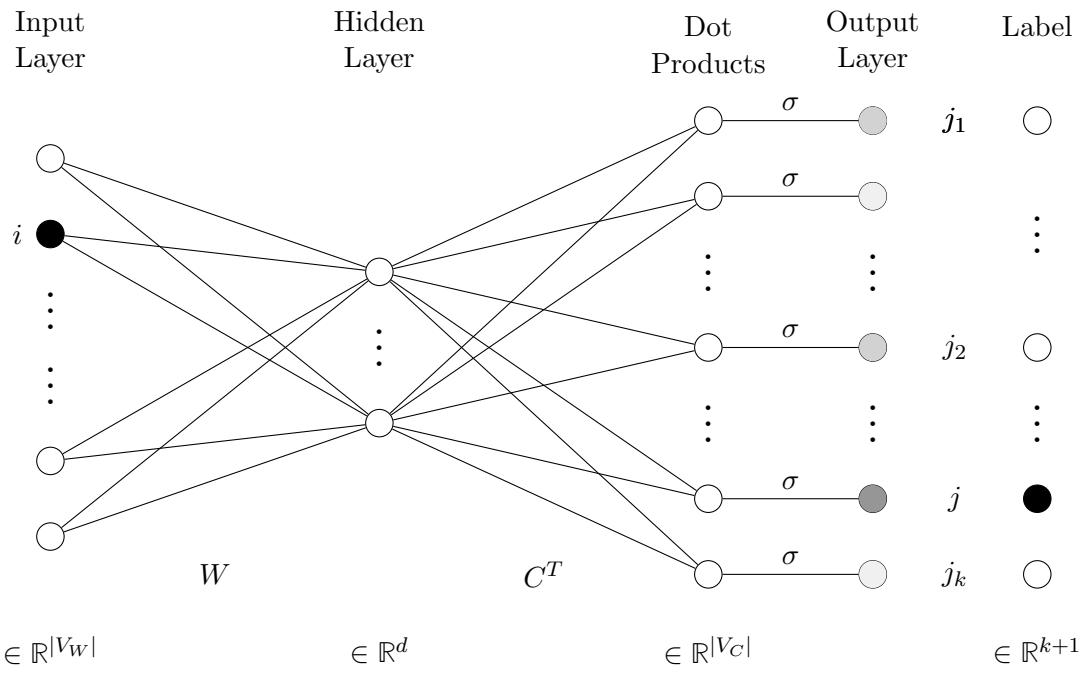



Figure 3.2: Skip-Gram with negative sampling

Since it is very difficult to work with equation (3.3) and to put the sampling in mathematical terms, we approximate this function using the expectation. This yields the *local* objective function for a specific word-context pair:

$$\log \sigma\left(\vec{w}_i \cdot \vec{c}_j\right) + k \cdot \mathbb{E}_{c_N \sim \mathrm{P}_D}\left[\log \sigma\left(-\vec{w}_i \cdot \vec{c}_N\right)\right]. \tag{3.4}$$

Another way to view this is that we are trying to maximize (3.4) and we use the Monte Carlo method to estimate the expectation.

We sum over all observed samples in the corpus to get the global objective function

$$\begin{aligned}\ell_{\mathrm{SGNS}}(W, C) &= \sum_{(w,c)\in D} \Big( \log \sigma\left(\vec{w} \cdot \vec{c}\right) + k \cdot \mathbb{E}_{c_N \sim \mathrm{P}_D}\left[\log \sigma\left(-\vec{w} \cdot \vec{c}_N\right)\right] \Big) \\ &= \sum_{w \in V_W} \sum_{c \in V_C} \#(w, c) \cdot \Big( \log \sigma\left(\vec{w} \cdot \vec{c}\right) + k \cdot \mathbb{E}_{c_N \sim \mathrm{P}_D}\left[\log \sigma\left(-\vec{w} \cdot \vec{c}_N\right)\right] \Big), \end{aligned} \tag{3.5}$$

where $W$ and $C$ are the word and context representation matrices containing the vectors $\vec{w}$ and $\vec{c}$.

It is now easy to see that for each training sample $(w, c)$, the objective function only depends on $k + 1$ rows of $C$ as opposed to *all* rows in skip-gram's objective function (3.2).

We have seen that the skip-gram model learns the probabilities $\mathrm{P}(c \mid w)$ from the statistics of the data. In the following paragraph, we want to explore what this corresponds to in the SGNS model.

Consider the following probability experiment. A box contains a red ball labeled $(w, c)$ for each word-context pair $(w, c)$ in $D$. Then, for each of these pairs, we draw $k$ contexts $c_j$

according to the probability distribution $\mathrm{P_D}(c)$ and add a blue ball labeled $(w, c_j)$. The box now contains $|D|$ red balls and $k \cdot |D|$ blue balls, the negative samples. We now draw a ball from the box labeled $(w, c)$ and ask the question whether this ball is red or blue, that is: Did the pair $(w, c)$ come from the data $D$ or was it put together at random? The probability that the pair came from the data is denoted by $\mathrm{P}(D = 1 \mid w, c)$. For example, for the pair (Freiburg, Breisgau), this probability should be high. In contrast, for the pair (Freiburg, desert), the probability that the context 'desert' was a negative sample, that is $1 - \mathrm{P}(D = 1 \mid w, c)$, should be high.

There is a total of $k \cdot \#(w)$ negative samples to choose for $w$ during one run over all samples. Hence, the expectation for the number of occurrences of $c$ as a negative sample for $w$ is $k \cdot \#(w) \cdot \mathrm{P_D}(c)$ and thus, we have

$$\begin{aligned} \mathrm{P}(D = 1 \mid w, c) &= \frac{\#(w, c)}{\#(w, c) + k \cdot \#(w) \cdot \mathrm{P_D}(c)} \\ &= \frac{\#(w, c)}{\#(w, c) + k \cdot \#(w) \cdot \frac{\#(c)}{|D|}}. \end{aligned} \tag{3.6}$$

Again, the network will learn the probabilities $\mathrm{P}(D = 1 \mid w, c)$ from the statistics of our sample collection $D$ and the randomly chosen contexts. Looking at the architecture, we see that the probabilities resulting from the SGNS network are given by

$$\mathrm{P}_{W,C}(D = 1 \mid w, c) = \sigma(\vec{w} \cdot \vec{c}), \tag{3.7}$$

where $W$ and $C$ are the word and context representation matrices, respectively.

# 4 The optimum assuming high embedding dimension

We have seen how skip-gram and SGNS learn low-dimensional word and context vectors. We called the matrices containing these vectors $W$ and $C$, respectively. In this chapter, we will analyze the matrix $M := W \cdot C^T$ in more depth.

$M$ is a $|V_W| \times |V_C|$-dimensional matrix, where the entry at row $i$ and column $j$ is the dot product $\vec{w}_i \cdot \vec{c}_j$. Looking at the objective functions $\ell_{\text{SG}}(W, C)$ and $\ell_{\text{SGNS}}(W, C)$ (equations 3.2 and 3.5), we observe that they only depend on the pairwise dot products of the embeddings and not on the entries of the vectors themselves. In other words, the network only depends on the matrix $M$ rather than on the matrices $W$ and $C$. This can also be verified looking at the architecture of the networks (Figure 3.1 and 3.2). Our matrix $M$ has rank at most $d$, where $d$ is the embedding dimension. This means for small $d$, the rows and columns of $M$ are linearly dependent. In this chapter, we will assume the embedding dimension is high enough (that is at least the size of the vocabulary) to choose each dot product independently from the other ones. Under this assumption, we can compute the maximum of the objective function with respect to the dot products.

In the following sections, we will need to make the assumption that $\#(w, c)$ is strictly greater than 0 for all word-context pairs $(w, c)$, as we will need to compute its logarithm, among other things. This is, of course, not a realistic assumption and we will address this problem in section 4.3.

## 4.1 Objective function of the SGNS model

**Lemma 4.1.** *For the function in (3.5), we have*

$$\ell_{SGNS}(W, C) = \sum_{w \in V_W} \sum_{c \in V_C} \ell_{w,c}(\vec{w}, \vec{c}),$$

*where*

$$\ell_{w,c}(\vec{w}, \vec{c}) = \#(w, c) \cdot \log \sigma(\vec{w} \cdot \vec{c}) + k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \log \sigma(-\vec{w} \cdot \vec{c}). \tag{4.1}$$

*Proof.* It is

$$\begin{aligned}
\ell_{\mathrm{SGNS}}(W,C) &= \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)\cdot\left(\log\sigma(\vec{w}\cdot\vec{c}) + k\cdot\mathbb{E}_{c_N\sim \mathrm{P_D}}\left[\log\sigma(-\vec{w}\cdot\vec{c}_N)\right]\right)\\
&= \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)\cdot\log\sigma(\vec{w}\cdot\vec{c})\\
&\quad + \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)\cdot k\cdot\mathbb{E}_{c_N\sim \mathrm{P_D}}\left[\log\sigma(-\vec{w}\cdot\vec{c}_N)\right]\\
&= \sum_{w\in V_W}\sum_{c\in V_C} \#(w,c)\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_W}\#(w)\cdot k\cdot\mathbb{E}_{c_N\sim \mathrm{P_D}}\left[\log\sigma(-\vec{w}\cdot\vec{c}_N)\right]
\end{aligned} \tag{4.2}$$

and we will now look at the second sum and use the definition of the expectation to see that

$$\begin{aligned}
&\sum_{w\in V_W}\#(w)\cdot k\cdot\mathbb{E}_{c_N\sim \mathrm{P_D}}\left[\log\sigma(-\vec{w}\cdot\vec{c}_N)\right]\\
&= \sum_{w\in V_W}\#(w)\cdot k\cdot\left(\sum_{c_N\in V_C}\mathrm{P_D}(c_N)\log\sigma(-\vec{w}\cdot\vec{c}_N)\right)\\
&= \sum_{w\in V_W}\sum_{c_N\in V_C}\#(w)\cdot k\cdot\mathrm{P_D}(c_N)\log\sigma(-\vec{w}\cdot\vec{c}_N)\\
&= \sum_{w\in V_W}\sum_{c\in V_C} k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\log\sigma(-\vec{w}\cdot\vec{c}).
\end{aligned}$$

Plugging this result into equation 4.2 yields

$$\begin{aligned}
\ell_{\mathrm{SGNS}}(W,C) &= \sum_{w\in V_W}\sum_{c\in V_C}\#(w,c)\log\sigma(\vec{w}\cdot\vec{c}) + \sum_{w\in V_W}\sum_{c\in V_C} k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\log\sigma(-\vec{w}\cdot\vec{c})\\
&= \sum_{w\in V_W}\sum_{c\in V_C}\left(\#(w,c)\cdot\log\sigma(\vec{w}\cdot\vec{c}) + k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\log\sigma(-\vec{w}\cdot\vec{c})\right),
\end{aligned}$$

which is what we wanted to show. □

Now, we define

$$\ell_{w,c}(x) = \#(w,c)\cdot\log\sigma(x) + k\cdot\#(w)\cdot\frac{\#(c)}{|D|}\log\sigma(-x) \tag{4.3}$$

and for $X = (x_{ij})$ let

$$\ell_{\mathrm{SGNS}}(X) = \sum_{i=1}^{|V_W|}\sum_{j=1}^{|V_C|}\ell_{w_i,c_j}(x_{ij}). \tag{4.4}$$

Then, it is $\ell_{w,c}(\vec{w},\vec{c}) = \ell_{w,c}(\vec{w}\cdot\vec{c})$ and $\ell_{\mathrm{SGNS}}(W,C) = \ell_{\mathrm{SGNS}}\left(W\cdot C^T\right)$ and we can find the maximum of $\ell_{w,c}(\vec{w},\vec{c})$ by considering $\ell_{w,c}(x)$.

**Theorem 4.1.** *The function $\ell_{w,c}(x)$ takes its maximum for*

$$x^*_{w,c} = \log\left(\frac{\#(w,c)\cdot|D|}{\#(w)\cdot\#(c)}\right) - \log k.$$

*Proof.* Since $\sigma(x) = \frac{1}{1+e^{-x}}$, we can compute $\frac{\partial \sigma}{\partial x}(x)$ using the quotient rule:

$$\begin{aligned}\frac{\partial \sigma}{\partial x}(x) &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\ &= \frac{1}{1+e^{-x}} \cdot \frac{1}{1+e^{x}} \\ &= \sigma(x) \cdot \sigma(-x).\end{aligned}$$

Using this, we can derive $\ell_{w,c}(x)$ and it is

$$\begin{aligned}\frac{\partial \ell_{w,c}}{\partial x}(x) &= \#(w,c) \cdot \frac{1}{\sigma(x)} \cdot \frac{\partial \sigma}{\partial x}(x) + k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \cdot \frac{1}{\sigma(-x)} \cdot (-1) \cdot \frac{\partial \sigma}{\partial x}(-x) \\ &= \#(w,c) \cdot \sigma(-x) - k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \cdot \sigma(x) \\ &= \#(w,c) \cdot \frac{1}{1+e^{x}} - k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \cdot \frac{1}{1+e^{-x}}.\end{aligned}$$

To compute the extrema, we now let $\frac{\partial \ell_{w,c}}{\partial x}(x) = 0$ or equivalently

$$\begin{aligned}0 =& \frac{\partial \ell_{w,c}}{\partial x}(x) \cdot (1+e^{x}) \cdot \left(1+e^{-x}\right) \cdot e^{x} \\ =& \#(w,c)\left(e^{x}+1\right) - k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \cdot \left(e^{x}+e^{2x}\right) \\ =& -k \cdot \#(w) \cdot \frac{\#(c)}{|D|} \cdot e^{2x} + \left(\#(w,c) - k \cdot \#(w) \cdot \frac{\#(c)}{|D|}\right) e^{x} + \#(w,c).\end{aligned}$$

Substituting $y = e^{x}$ and dividing by $-k \cdot \#(w) \cdot \frac{\#(c)}{|D|}$, we arrive at

$$\begin{aligned}0 =& y^2 + \left(1 - \frac{\#(w,c)}{k \cdot \#(w) \cdot \frac{\#(c)}{|D|}}\right) \cdot y - \frac{\#(w,c)}{k \cdot \#(w) \cdot \frac{\#(c)}{|D|}} \\ =& (y+1)\left(y - \frac{\#(w,c)}{k \cdot \#(w) \cdot \frac{\#(c)}{|D|}}\right).\end{aligned}$$

Since $-1 = e^{x}$ does not have a solution in $\mathbb{R}$, the derivative $\frac{\partial \ell_{w,c}}{\partial x}(x)$ only has one zero, namely

$$x = \log\left(\frac{\#(w,c)}{k \cdot \#(w) \cdot \frac{\#(c)}{|D|}}\right) = \log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c)}\right) - \log k.$$

This is, in fact, a local and global maximum because

$$\frac{\partial^2}{(\partial x)^2} \ell_{w,c}(x) = \sigma(x) \cdot \sigma(-x) \cdot \left(-\#(w,c) - k \cdot \#(w) \cdot \frac{\#(c)}{|D|}\right) < 0$$

for all $x \in \mathbb{R}$. □

The preceding theorem tells us the optimal value of $\vec{w} \cdot \vec{c}$ for each word-context pair $(w, c)$. This means that the SGNS model reaches its optimum if $W \cdot C^T$ equals the matrix $M^{\text{SGNS}}$ with

$$M_{ij}^{\text{SGNS}} = \log\left(\frac{\#(w_i, c_j) \cdot |D|}{\#(w_i) \cdot \#(c_j) \cdot k}\right).$$

Note that

$$\begin{aligned}
\log\left(\frac{\#(w, c) \cdot |D|}{\#(w) \cdot \#(c)}\right) &= \log\left(\frac{\frac{\#(w,c)}{|D|}}{\frac{\#(w)}{|D|} \cdot \frac{\#(c)}{|D|}}\right) \\
&= \log\left(\frac{p(w, c)}{p(w) \cdot p(c)}\right) \\
&= \text{PMI}(w; c),
\end{aligned}$$

the Pointwise Mutual Information of $w$ and $c$ and hence

$$M_{ij}^{\text{SGNS}} = \text{PMI}(w_i; c_j) - \log k.$$

We can now compute the probability $\text{P}_{W,C}(D = 1 \mid w, c)$ assuming that $W \cdot C^T = M^{\text{SGNS}}$. It is

$$\begin{aligned}
\text{P}_{W,C}(D = 1 \mid w, c) &= \sigma(\vec{w} \cdot \vec{c}) \\
&= \frac{1}{1 + \exp\left(-\log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c) \cdot k}\right)\right)} \\
&= \frac{1}{1 + \frac{\#(w) \cdot \#(c) \cdot k}{\#(w,c) \cdot |D|}} \\
&= \frac{\#(w, c)}{\#(w, c) + \#(w) \cdot \frac{\#(c)}{|D|} \cdot k} \\
&= \text{P}(D = 1 \mid w, c).
\end{aligned}$$

Thus, for the optimal values for the dot products, the modeled probabilities are equal to the actual probabilities we computed in equation 3.6.

## 4.2 Objective function of the skip-gram model

For the skip-gram model, the task of finding the optimum value for the dot products is much harder, due to the previously discussed fact that each output neuron depends on a large number of weights.

Let $X = (x_{ij})$ and let

$$\ell_{\text{SG}}(X) = \sum_{i=1}^{|V_W|} \sum_{j=1}^{|V_C|} \#(w_i, c_j) \cdot \left(x_{ij} - \log\left(\sum_{k=1}^{|V_C|} \exp(x_{ik})\right)\right). \tag{4.5}$$

Then, $\ell_{\text{SG}}(W, C)$ in equation 3.2 is equal to $\ell_{\text{SG}}(W \cdot C^T)$.

We would expect $\ell_{\mathrm{SG}}(W, C)$ to take a maximum for the word and context matrices $W$ and $C$ being such that $\mathrm{P}_{W,C}(c \mid w)$ in 3.1 equals $\mathrm{P}(c \mid w)$, the actual probability that a randomly chosen context of $w$ is $c$. For the representation matrices being such that $\vec{w} \cdot \vec{c} = \log \#(w, c)$, we get

$$\begin{aligned}
\mathrm{P}_{W,C}(c \mid w) &= \frac{\exp(\log \#(w, c))}{\sum_{c' \in V_C} \exp(\log \#(w, c'))} \\
&= \frac{\#(w, c)}{\sum_{c' \in V_C} \#(w, c')} \\
&= \frac{\#(w, c)}{\#(w)} \\
&= \mathrm{P}(c \mid w),
\end{aligned}$$

which motivates the following

**Theorem 4.2.** *The function $\ell_{SG}(X)$ has a local maximum at $X^* = \left(x^*_{ij}\right)$ with*

$$x^*_{ij} = \log \#(w_i, c_j).$$

*Proof.* We compute the partial derivatives and show that they are 0 at our point. It is

$$\begin{aligned}
\frac{\partial \ell_{\mathrm{SG}}}{\partial x_{i_0 j_0}}(X) &= \#(w_{i_0}, c_{j_0}) - \frac{\partial}{\partial x_{i_0 j_0}} \left( \sum_i \sum_j \#(w_i, c_j) \cdot \log \left( \sum_k \exp(x_{ik}) \right) \right) \\
&= \#(w_{i_0}, c_{j_0}) - \frac{\partial}{\partial x_{i_0 j_0}} \left( \sum_j \#(w_{i_0}, c_j) \cdot \log \left( \sum_k \exp(x_{i_0 k}) \right) \right) \\
&= \#(w_{i_0}, c_{j_0}) - \sum_j \#(w_{i_0}, c_j) \cdot \frac{1}{\sum_k \exp(x_{i_0 k})} \cdot \exp(x_{i_0 j_0}) \\
&= \#(w_{i_0}, c_{j_0}) - \#(w_{i_0}) \cdot \frac{\exp(x_{i_0, j_0})}{\sum_k \exp(x_{i_0 k})}
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial \ell_{\mathrm{SG}}}{\partial x_{i_0 j_0}}(X^*) &= \#(w_{i_0}, c_{j_0}) - \#(w_{i_0}) \cdot \frac{\#(w_{i_0}, c_{j_0})}{\sum_k \#(w_{i_0}, c_k)} \\
&= \#(w_{i_0}, c_{j_0}) - \#(w_{i_0}) \cdot \frac{\#(w_{i_0}, c_{j_0})}{\#(w_{i_0})} \\
&= 0.
\end{aligned}$$

We will now compute all second order partial derivatives. It is

$$\begin{aligned}\frac{\partial^2 \ell_{\mathrm{SG}}}{\left(\partial x_{i_0 j_0}\right)^2}(X) &= \frac{\partial}{\partial x_{i_0 j_0}}\left(\#\left(w_{i_0}, c_{j_0}\right) - \#\left(w_{i_0}\right) \cdot \frac{\exp\left(x_{i_0 j_0}\right)}{\sum_k \exp\left(x_{i_0 k}\right)}\right) \\ &= -\#\left(w_{i_0}\right) \cdot \frac{\partial}{\partial x_{i_0 j_0}}\left(\frac{\exp\left(x_{i_0 j_0}\right)}{\sum_k \exp\left(x_{i_0 k}\right)}\right) \\ &= -\#\left(w_{i_0}\right) \cdot \frac{\exp\left(x_{i_0 j_0}\right)\sum_k \exp\left(x_{i_0 k}\right) - \exp\left(x_{i_0, j_0}\right)\exp\left(x_{i_0 j_0}\right)}{\left(\sum_k \exp\left(x_{i_0 k}\right)\right)^2} \\ &= -\#\left(w_{i_0}\right) \cdot \exp\left(x_{i_0 j_0}\right) \cdot \frac{\sum_{k \neq j_0} \exp\left(x_{i_0 k}\right)}{\left(\sum_k \exp\left(x_{i_0 k}\right)\right)^2}.\end{aligned}$$

Evaluated at $x^*_{ij} = \log \#\left(w_i, c_j\right)$, we get

$$\begin{aligned}\frac{\partial^2 \ell_{\mathrm{SG}}}{\left(\partial x_{i_0 j_0}\right)^2}\left(X^*\right) &= -\#\left(w_{i_0}\right) \cdot \#\left(w_{i_0}, c_{j_0}\right) \cdot \frac{\sum_{k \neq j_0} \#\left(w_{i_0}, c_k\right)}{\left(\sum_k \#\left(w_{i_0}, c_k\right)\right)^2} \\ &= -\#\left(w_{i_0}, c_{j_0}\right) \cdot \frac{\sum_{k \neq j_0} \#\left(w_{i_0}, c_k\right)}{\#\left(w_{i_0}\right)} \\ &= -\#\left(w_{i_0}, c_{j_0}\right) \cdot \frac{\#\left(w_{i_0}\right) - \#\left(w_{i_0}, c_{j_0}\right)}{\#\left(w_{i_0}\right)} \\ &= \#\left(w_{i_0}, c_{j_0}\right) \cdot \left(\frac{\#\left(w_{i_0}, c_{j_0}\right)}{\#\left(w_{i_0}\right)} - 1\right).\end{aligned}$$

For $i_0 \neq i_1$, we have

$$\begin{aligned}\frac{\partial^2 \ell_{\mathrm{SG}}}{\partial x_{i_0 j_0} \partial x_{i_1 j_1}}(X) &= \frac{\partial}{\partial x_{i_1 j_1}}\left(\#\left(w_{i_0}, c_{j_0}\right) - \#\left(w_{i_0}\right) \cdot \frac{\exp\left(x_{i_0 j_0}\right)}{\sum_k \exp\left(x_{i_0 k}\right)}\right) \\ &= 0\end{aligned}$$

for all $X$, since there is no $i_1$ in the function. Lastly, for $j_0 \neq j_1$, it is

$$\begin{aligned}\frac{\partial^2 \ell_{\mathrm{SG}}}{\partial x_{i_0 j_0} \partial x_{i_0 j_1}}(X) &= \frac{\partial}{\partial x_{i_0 j_1}}\left(\#\left(w_{i_0}, c_{j_0}\right) - \#\left(w_{i_0}\right) \cdot \frac{\exp\left(x_{i_0 j_0}\right)}{\sum_k \exp\left(x_{i_0 k}\right)}\right) \\ &= -\#\left(w_{i_0}\right) \cdot \exp\left(x_{i_0 j_0}\right) \frac{\partial}{\partial x_{i_0 j_1}}\left(\frac{1}{\sum_k \exp\left(x_{i_0 k}\right)}\right) \\ &= -\#\left(w_{i_0}\right) \cdot \exp\left(x_{i_0 j_0}\right) \frac{\exp\left(x_{i_0 j_1}\right)}{\left(\sum_k \exp\left(x_{i_0 k}\right)\right)^2}\end{aligned}$$

and for $x^*_{ij} = \log \#\left(w_i, c_j\right)$ we have

$$\begin{aligned}\frac{\partial^2 \ell_{\mathrm{SG}}}{\partial x_{i_0 j_0} \partial x_{i_0 j_1}}\left(X^*\right) &= -\#\left(w_{i_0}\right) \cdot \#\left(w_{i_0}, c_{j_0}\right) \frac{\#\left(w_{i_0}, c_{j_1}\right)}{\left(\sum_k \#\left(w_{i_0}, c_k\right)\right)^2} \\ &= -\frac{\#\left(w_{i_0}, c_{j_0}\right) \#\left(w_{i_0}, c_{j_1}\right)}{\#\left(w_{i_0}\right)}.\end{aligned}$$

Let $H$ be the Hessian matrix of $\ell_{\mathrm{SG}}$ at $X^*$. Summarizing the above, we have

$$H_{(ij)(kl)} = \begin{cases} \#\left(w_i, c_j\right) \cdot \left(\frac{\#\left(w_i, c_j\right)}{\#\left(w_i\right)} - 1\right) & \text{if } i = k \text{ and } j = l \\ -\frac{\#\left(w_i, c_j\right) \#\left(w_i, c_l\right)}{\#\left(w_i\right)} & \text{if } i = k \text{ and } j \neq l \\ 0 & \text{if } i \neq k. \end{cases} \tag{4.6}$$

As the last step of this proof, we need to show that $H$ is negative definite. We will show that our Hessian $H$ has the property that

$$H_{(ij)(ij)} = \sum_{l \neq j} H_{(ij)(il)}. \tag{4.7}$$

This is because

$$\begin{aligned}
\sum_{l \neq j} H_{(ij)(il)} &= \sum_{l \neq j} -\frac{\#(w_i, c_j)\,\#(w_i, c_l)}{\#(w_i)} \\
&= \frac{\#(w_i, c_j)}{\#(w_i)} \cdot \left( \#(w_i, c_j) - \sum_l \#(w_i, c_l) \right) \\
&= \#(w_i, c_j) \cdot \left( \frac{\#(w_i, c_j)}{\#(w_i)} - 1 \right) \\
&= H_{(ij)(ij)}.
\end{aligned}$$

We will show the negative definiteness of $H$ using the definition, that is by showing that $v^T \cdot H \cdot v < 0$ for all $v \neq 0$. In our case, this translates to

$$\sum_{i,j,k,l} v_{ij} H_{(ij)(kl)} v_{kl} < 0.$$

Using 4.6 and 4.7 and the fact that $H$ is symmetric, we have

$$\begin{aligned}
v^T \cdot H \cdot v &= \sum_{i,j,k,l} v_{ij} H_{(ij)(kl)} v_{kl} \\
&= \sum_{i,j,l} v_{ij} H_{(ij)(il)} v_{il} \\
&= \sum_i \left( \sum_j \sum_{l<j} v_{ij} v_{il} H_{(ij)(il)} + \sum_j \sum_{l>j} v_{ij} v_{il} H_{(ij)(il)} + \sum_j v_{ij}^2 H_{(ij)(ij)} \right) \\
&= \sum_i \left( 2 \cdot \sum_j \sum_{l<j} v_{ij} v_{il} H_{(ij)(il)} + \sum_j v_{ij}^2 H_{(ij)(ij)} \right) \\
&= \sum_i \left( 2 \cdot \sum_j \sum_{l<j} v_{ij} v_{il} H_{(ij)(il)} + \sum_j \sum_{l \neq j} v_{ij}^2 H_{(ij)(il)} \right) \\
&= \sum_i \left( 2 \cdot \sum_j \sum_{l<j} v_{ij} v_{il} H_{(ij)(il)} + \sum_j \sum_{l<j} \left( v_{ij}^2 + v_{il}^2 \right) H_{(ij)(il)} \right) \\
&= \sum_i \sum_j \sum_{l<j} \left( 2 \cdot v_{ij} v_{il} + v_{ij}^2 + v_{il}^2 \right) H_{(ij)(il)} \\
&= \sum_i \sum_j \sum_{l<j} \left( v_{ij} + v_{il} \right)^2 H_{(ij)(il)} \\
&= -\sum_i \sum_j \sum_{l<j} \left( v_{ij} + v_{il} \right)^2 \frac{\#(w_i, c_j)\,\#(w_i, c_l)}{\#(w_i)}.
\end{aligned}$$

In order for the matrix to be (strictly) negative definite, it remains to show that there are indices $i_0, j_0$ and $l_0$ with $j_0 \neq l_0$ such that $v_{i_0 j_0} + v_{i_0 l_0} \neq 0$. Assume no such indices existed. Then, let $i$ and $j$ be arbitrary indices. Let $k$ and $l$ be indices such that $j, k$ and $l$ are pairwise different. Then, $v_{ij} + v_{ik} = v_{ij} + v_{il} = v_{ik} + v_{il} = 0$ and $v_{ij} = -v_{ik} = v_{il} = -v_{ij}$ and thus $v_{ij} = 0$. However, this cannot be true for all $i$ and $j$ since $v \neq 0$. Therefore, there must be indices as claimed.

This shows that $H$ is negative definite and therefore proves that there is a maximum for $x^*_{ij} = \log \# (w_i, c_j)$. □

In conclusion, the skip-gram model reaches a local maximum if $W \cdot C^T$ equals the matrix $M^{\mathrm{SG}}$ with

$$M^{\mathrm{SG}}_{ij} = \log \# (w_i, c_j) .$$

## 4.3 Non-negative matrices

We have found the optimal values for the pairwise dot products of the word and context embeddings:

$$M^{\mathrm{SGNS}}_{ij} = \log \left( \frac{\# (w_i, c_j) \cdot |D|}{\# (w_i) \cdot \# (c_j) \cdot k} \right) = \mathrm{PMI} (w_i; c_j) - \log k$$

and

$$M^{\mathrm{SG}}_{ij} = \log \# (w_i, c_j) .$$

In order for these expressions to be well-defined, we made the assumption that $\# (w, c)$ is greater than 0 for all words $w$ and contexts $c$. However, most word-context pairs will never appear in a text. One approach to address this problem is to pretend that each pair has been seen by adding a 'fake' sample to the data for every pair. There is, however, a different approach to solving this problem.

We replace all ill-defined entries of $M^{\mathrm{SG}}$ and $M^{\mathrm{SGNS}}$ by 0. Another neat advantage of this is that the resulting matrices are sparse, since most word-context pairs are never observed. This works great for the skip-gram model. However, it leads to a new problem in the SGNS model. There are word-context pairs $(w, c)$ with small $\# (w, c)$, but large $\# (w)$ and $\# (c)$, that is frequent words that only appear together a few times. These word-context pairs are uncorrelated and it is

$$\frac{\# (w, c) \cdot |D|}{\# (w) \cdot \# (c) \cdot k} < 1$$

and their entry in $M^{\mathrm{SGNS}}$ is therefore negative. This means that there is an inconsistency, since pairs that *never* appear together will be assigned a higher value (namely 0) than pairs of frequent words that only appear together in the corpus *a few times*. Thus, we will use the matrices

$$M^{\mathrm{SG+}}_{ij} = \max (\log \# (w_i, c_j) , 0) \tag{4.8}$$

and

$$M^{\mathrm{SGNS+}}_{ij} = \max (\mathrm{PMI} (w_i; c_j) - \log k, 0) , \tag{4.9}$$

using the convenient definition of $\log (0) = -\infty$. In their paper [8], Levy and Goldberg explain that there is also human intuition behind this:

"[...] humans can easily think of positive associations (e.g. 'Canada' and 'snow') but find it much harder to invent negative ones ('Canada' and 'desert'). This suggests that the perceived similarity of two words is more influenced by the positive context they share than by the negative context they share. It therefore makes some intuitive sense to discard the negatively associated contexts and mark them as 'uninformative' (0) instead."

# 5 Singular Value Decomposition

In the previous chapter, we found values for the dot products of each word-context pair that maximize the objective function for each model, skip-gram and SGNS. In this chapter, we denote the matrix containing the optimal dot products as $M^{\mathrm{OPT}}$. If it is possible to find matrices $W \in \mathrm{Mat}_{|V_W| \times d}$ and $C \in \mathrm{Mat}_{|V_C| \times d}$ such that $M^{\mathrm{OPT}} = W \cdot C^T$, then using $W$ as $C$ as word and context representations will maximize the objective function.

Clearly, for a small embedding dimension $d$, it is not possible to find these matrices, since the rank of the product cannot be greater than $d$. However, we can try to find a *low-rank approximation* $M_d$ of $M^{\mathrm{OPT}}$ and then factorize $M_d$ into $W \cdot C^T$.

Singular Value Decomposition (SVD) provides a factorization

$$M^{\mathrm{OPT}} = U \cdot \Sigma \cdot V^T,$$

where $U \in \mathrm{Mat}_{|V_W| \times |V_W|}$ and $V \in \mathrm{Mat}_{|V_C| \times |V_C|}$ are orthogonal and $\Sigma \in \mathrm{Mat}_{|V_W| \times |V_C|}$ is diagonal with the non-negative singular values on the diagonal. We assume the singular values on the diagonal are in decreasing order. In that case, $\Sigma$ is uniquely determined. Let $\Sigma_d$ be the $d \times d$-dimensional diagonal matrix with the $d$ largest singular values on its diagonal. Due to our assumption, these are the first $d$ entries of $\Sigma$. Moreover, let $U_d$ and $V_d$ be the matrices containing the first $d$ columns of $U$ and $V$ (see Figure 5.1). Then, the Eckart-Young-Mirsky theorem [4] states that $M_d = U_d \cdot \Sigma_d \cdot V_d^T$ is the best rank $d$ approximation of $M^{\mathrm{OPT}}$ with respect to the Frobenius norm and the spectral norm (for a formal proof, see [2]). That is

$$\begin{aligned} M_d &= \underset{M|\mathrm{rk}(M)=d}{\arg\min} ||M - M^{\mathrm{OPT}}||_{\mathrm{F}} \\ &= \underset{M|\mathrm{rk}(M)=d}{\arg\min} ||M - M^{\mathrm{OPT}}||_2, \end{aligned} \tag{5.1}$$

where $||A||_{\mathrm{F}} = \sqrt{\sum_i \sum_j {a_{i,j}}^2}$ and $||A||_2 = \max_{||x||=1} ||Ax||_2$, where $||v||_2$ is the Euclidean norm for vectors. This process is called *truncated SVD*. We can now use the matrices $U_d$, $\Sigma_d$ and $V_d$ to define the word and context matrices $W$ and $C$ as follows.

$$W = U_d \cdot \sqrt{\Sigma_d}$$

and

$$C = V_d \cdot \sqrt{\Sigma_d},$$

where $\sqrt{\Sigma_d}$ is the diagonal matrix containing the square roots of the non-negative entries of $\Sigma_d$. Then,

$$\begin{aligned} W \cdot C^T &= U_d \cdot \sqrt{\Sigma_d} \cdot \sqrt{\Sigma_d}^T \cdot V_d^T \\ &= U_d \cdot \Sigma_d \cdot V_d^T \\ &= M_d \end{aligned}$$

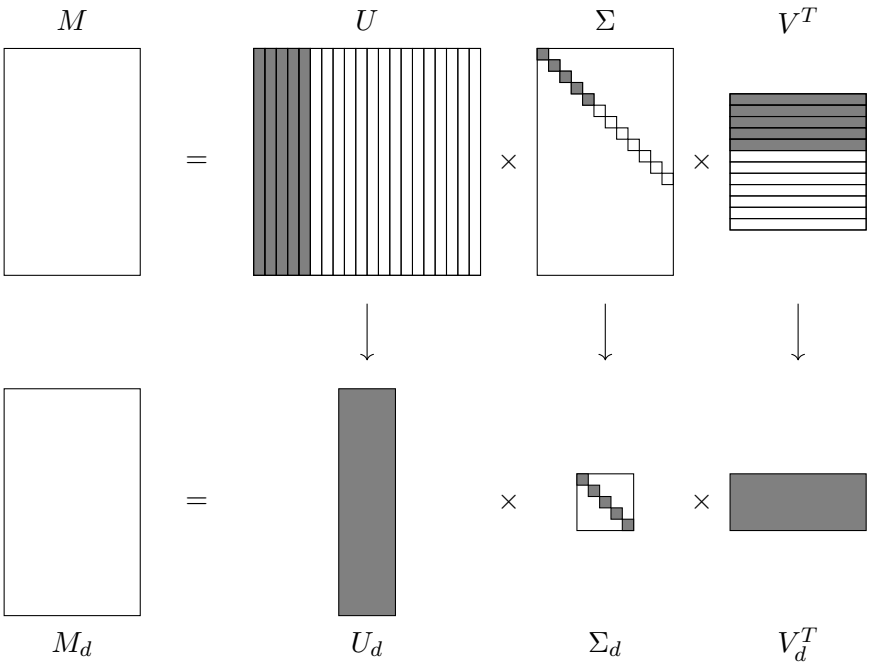



Figure 5.1: Truncated Singular Value Decomposition

and $W \cdot C^T$ is the best possible rank $d$ approximation of $M^{\text{OPT}}$.

In conclusion, SVD maximizes a function that is different from the objective function of our neural networks. However, we have seen that for large $d$, the SVD embeddings maximize the objective functions of our neural networks.

# 6 Mathematical analysis of the results

In order to evaluate results, we need to have a *measure of distance* between two word vectors. Using this measure, we are able to speak of the similarity between words. We also want to be able to do simple computations using the word vectors. For instance, we wish to answer analogy questions of the form "$a$ is to $a^*$ as $b$ is to ?". Later on, we will see that this translates to finding the closest word vector to the vector $a^* - a + b$.

## 6.1 Analyzing different distance measures

In natural language processing, it is common to use the cosine distance, even though it is not a metric. In this section, we will find a distance measure that outputs the same ordering of our results as the cosine distance, but is an actual metric.

**Definition 6.1.** Let $v, w \in \mathbb{R}^n$ be two vectors. The *cosine similarity* $S_C(v, w)$ of $v$ and $w$ is defined as

$$S_C(v, w) = \frac{v \cdot w}{||v|| \cdot ||w||} = \cos(\theta),$$

where $v \cdot w$ is the dot product and $\theta$ is the angle between the vectors $v$ and $w$.

**Definition 6.2.** Let $v, w \in \mathbb{R}^n$ be two vectors. The *cosine distance* $D_C(v, w)$ of $v$ and $w$ is defined as

$$D_C(v, w) = 1 - \frac{v \cdot w}{||v|| \cdot ||w||} = 1 - S_C(v, w),$$

where $v \cdot w$ is the scalar product.

**Lemma 6.1.** *On the $(n-1)$-sphere, the cosine distance is a positive-definite, symmetric function, but it is not a metric, since it does not satisfy the triangle inequality.*

*Proof.* For two vectors $v, w \in S^{n-1}$ on the $(n-1)$-sphere, the cosine distance is

$$D_C(v, w) = 1 - v \cdot w,$$

since their norm is 1. It is easy to see that $D_C$ is a positive-definite, symmetric function on $S^{n-1}$. To show that $D_C$ does not satisfy the triangle inequality, consider the following counterexample. Let $x = \left(\frac{1}{2}, \frac{\sqrt{3}}{2}\right)$, $y = (1, 0)$ and $z = \left(\frac{1}{2}, -\frac{\sqrt{3}}{2}\right)$. Then, $x, y, z \in S^1$ and

$$\begin{aligned} D_C(x, y) + D_C(y, z) &= \left(1 - \frac{1}{2}\right) + \left(1 - \frac{1}{2}\right) \\ &= 1 < \frac{3}{2} \\ &= 1 - \left(\frac{1}{4} - \frac{3}{4}\right) \\ &= D_C(x, z). \end{aligned}$$

*Note:* This works for any dimension, as we can add zeros, which does not change the scalar product. □

We can slightly change the distance by taking the *angle* between two vectors (rather than the cosine of the angle) and we will see that this is, in fact, a metric on the sphere.

**Definition 6.3.** Let $v, w \in S^{n-1}$ be two vectors of length 1. Then, we define the *angular distance* $D_A(v, w)$ of $v$ and $w$ to be

$$D_A(v, w) = \cos^{-1}(S_C(v, w)) = \cos^{-1}(v \cdot w).$$

*Remark.* The angular distance is simply the angle between the two vectors.

**Theorem 6.1.** *The angular distance defines a metric on the* $(n-1)$*-sphere.*

*Proof.* It is $D_A(v, w) \in [0, \pi]$ for all $v, w \in S^{n-1}$. Moreover, $D_A(v, w)$ is 0 if and only if $v \cdot w$ is 1, which is equivalent to $v = w$. This means that $D_A$ is a positive-definite function. $D_A$ is clearly symmetric, so we only need to show the triangle inequality. This can easily be verified geometrically. The sum of the angle between vectors $u$ and $v$ and the angle between $v$ and $w$ cannot be less than the angle between $u$ and $w$, since we can rotate $u$ to $v$ first and then to $w$. We will still give a formal proof.

Let $u, v, w \in S^{n-1}$ be elements on the $(n-1)$-sphere. We need to show that

$$D_A(u, v) + D_A(v, w) \geq D_A(u, w).$$

We can assume without loss of generality that $v = (1, 0, \ldots, 0)^T$ is the north pole, because the dot product of two vectors is invariant under rotation or more specifically under multiplication by orthogonal matrices. Let $u_1$ and $v_1$ be the first coordinate of $u$ and $v$ respectively. We need to show that

$$\cos^{-1}(u_1) + \cos^{-1}(w_1) \geq \cos^{-1}(u \cdot w).$$

If the left hand side is greater than $\pi$, there is nothing to show. If it is less than $\pi$, we can apply the cosine to both sides and we need to show that

$$\cos\left(\cos^{-1}(u_1) + \cos^{-1}(w_1)\right) \leq u \cdot w,$$

as the cosine is strictly monotonically decreasing on the interval $[0, \pi]$. Let $u'$ and $w'$ be the vectors containing the last $n-1$ entries of $u$ and $v$ respectively. Using the angle sum identity and later the Cauchy-Schwarz inequality, we get

$$\begin{aligned}
\cos\left(\cos^{-1}(u_1) + \cos^{-1}(w_1)\right) &= u_1 w_1 - \sin\left(\cos^{-1}(u_1)\right)\sin\left(\cos^{-1}(w_1)\right) \\
&= u_1 w_1 - \sqrt{1 - \cos\left(\cos^{-1}(u_1)\right)^2}\sqrt{1 - \cos\left(\cos^{-1}(w_1)\right)^2} \\
&= u_1 w_1 - \sqrt{1 - u_1^2}\sqrt{1 - w_1^2} \\
&= u_1 w_1 - \sqrt{\sum_{i=2}^{n} u_i^2}\sqrt{\sum_{i=2}^{n} w_i^2} \\
&= u_1 w_1 - ||-u'|| \cdot ||w'|| \\
&\leq u_1 w_1 + u' \cdot w' \\
&= u \cdot w.
\end{aligned}$$

□

We are interested in sorting our words by their proximity to a certain vector or point on the sphere. The next theorem states that, in this case, the choice of the distance (cosine or angular) does not change the order of the word vectors.

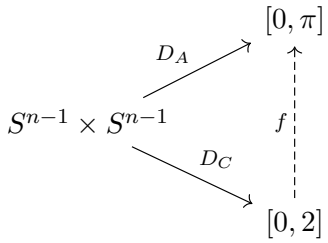
**Theorem 6.2.** *For the cosine distance $D_C$ and the angular distance $D_A$, we have*

*(i)* $D_C(v,w) < D_C(x,y) \iff D_A(v,w) < D_A(x,y)$ *and*

*(ii)* $D_C(v,w) = D_C(x,y) \iff D_A(v,w) = D_A(x,y)$

*for all* $v, w, x, y \in S^{n-1}$.

*Proof.* Consider the following diagram.




For $f(a) = \cos^{-1}(1-a)$, we have

$$\begin{aligned} f(D_C(x,y)) &= f(1 - x \cdot y) \\ &= \cos^{-1}(1 - (1 - x \cdot y)) \\ &= \cos^{-1}(x \cdot y) \\ &= D_A(x,y). \end{aligned}$$

Hence, the diagram is commutative. Deriving $f$ shows that it is a strictly monotonically increasing function:

$$\frac{\partial f}{\partial a}(a) = \frac{1}{\sqrt{2a - a^2}} > 0 \quad \text{for } a \in (0,2).$$

Since $f$ is strictly increasing, $D_C(v,w) < D_C(x,y)$ is equivalent to

$$D_A(v,w) = f(D_C(v,w)) < f(D_C(x,y)) = D_A(x,y)$$

and the same is true for equality. □

**Corollary 6.1.** *Let $x \in S^{n-1}$ be arbitrary and let $F \subset S^{n-1}$ be a finite subset. Then, the ordering of the elements of $F$ by their distance to $x$ does not depend on which distance, cosine distance or angular distance, we use.*

For the evaluation of our results, we will still use the cosine distance, since it is easier to compute and it is the common measure to use for word embedding tasks. This section serves as a justification to use the cosine distance, even though it is not a metric itself.

## 6.2 Expectation of the Closest Vector

In the preceding section, we have discussed different options for a distance to compare our word vectors. We will now look at the cosine distance in more depth. It is important to know which results are meaningful. Therefore, we will now analyze how close (with respect to the cosine distance) our vectors have to be to be able to call them 'related' or 'similar'.

Assume we have a set $F \subset S^{d-1}$ of $n$ points that are randomly distributed on the $(d-1)$-sphere according to the uniform distribution. This set corresponds to our word vectors. Since we are interested in the cosine distance, which only depends on the direction and not the length of a vector, we can normalize our word vectors. In the following section, we will compute the distribution and then the expectation of the minimum of the distances between a fixed point $p$ on the sphere and the $n$ random points on the sphere, that is

$$\mathbb{E}\left[\min_{x \in F} \left(D_C\left(p, x\right)\right)\right].$$

This value depends on the number of points $n$ and the dimension $d$. The goal of this section is to find a manageable formula for the expectation. Using this formula for $n = |V_W|$, the number of words, and the embedding dimension $d$, we have a rule of thumb for which results are 'meaningful': words with a distance less than this value are probably good results, words with a distance larger than this value are likely to have nothing in common with the query input.

The next lemma gives us the uniform distribution on the $(d-1)$-sphere.

**Lemma 6.2.** *Let $X_1, \ldots, X_d$ be independent, standard normally distributed random variables and let $X = (X_1, \ldots, X_d)$. Then, $Y := ||X||^{-1} \cdot X$ is uniformly distributed on the $(d-1)$-sphere.*

*Proof.* $X$ is a standard normally distributed random vector, that is a multivariate normal random variable with mean $\mu = (0, \ldots, 0)^T$ and variance $\Sigma = I_d$, the identity matrix. Hence, $X$ has the density function

$$f_X\left(x\right) = \frac{1}{\sqrt{\left(2\pi\right)^d}} \exp\left(-\frac{1}{2} x^T x\right).$$

For any orthogonal matrix $O \in O\left(d\right)$, we have:

$$\begin{aligned} f_X\left(Ox\right) &= \frac{1}{\sqrt{\left(2\pi\right)^d}} \exp\left(-\frac{1}{2}\left(Ox\right)^T \left(Ox\right)\right) \\ &= \frac{1}{\sqrt{\left(2\pi\right)^d}} \exp\left(-\frac{1}{2} x^T O^T O x\right) \\ &= \frac{1}{\sqrt{\left(2\pi\right)^d}} \exp\left(-\frac{1}{2} x^T x\right) \\ &= f_X\left(x\right). \end{aligned}$$

In particular, $f_X$ is invariant under rotations. Hence, the distribution of $Y$ is invariant under rotations as well. But this means that $Y$ is uniformly distributed on the $(d-1)$-sphere, because for each pair of points $p_1, p_2 \in S^{d-1}$, there is a rotation that maps $p_1$ onto $p_2$. □

Without loss of generality, we can assume that $p$, our fixed point on the sphere, is the north pole. In this case, we have

$$\begin{aligned} D_C(p,x) &= 1 - p \cdot x \\ &= 1 - x_1, \end{aligned}$$

where $x_1$ is the first coordinate of $x$. Thus, our next step is to compute the distribution function for the first coordinate of a randomly chosen point on the sphere.

**Theorem 6.3.** *Let $Y$ be a uniformly distributed random variable on the $(d-1)$-sphere. Then, the first entry of $Y$, $Y_1$, has the distribution function*

$$F_{Y_1}(y) = 1 - \frac{1}{2} \cdot \frac{\int_y^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds} \tag{6.1}$$

*for $y \in (-1,1)$.*

*Proof.* Let $X_1, \ldots, X_d$ be independent, standard normally distributed random variables and let $X = (X_1, \ldots, X_d)$. According to lemma 6.2, $Y$ and $||X||^{-1}X$ are identically distributed and we have

$$\mathrm{P}(Y_1 > y) = \mathrm{P}\left(||X||^{-1} X_1 > y\right).$$

First, we assume $y \in [0,1)$ is non-negative. We have

$$\begin{aligned} \mathrm{P}\left(||X||^{-1}X_1 > y\right) =& \mathrm{P}(X_1 > ||X|| \cdot y \mid X_1 > 0)\,\mathrm{P}(X_1 > 0) + \\ & \mathrm{P}(X_1 > ||X|| \cdot y \mid X_1 \leq 0)\,\mathrm{P}(X_1 \leq 0) \\ =& \frac{1}{2} \cdot \mathrm{P}(X_1 > ||X|| \cdot y \mid X_1 > 0) \\ =& \frac{1}{2} \cdot \mathrm{P}\left(X_1^2 > \left(X_1^2 + \cdots + X_d^2\right) \cdot y^2 \mid X_1 > 0\right) \\ =& \frac{1}{2} \cdot \mathrm{P}\left(X_1^2\left(1-y^2\right) > \left(X_2^2 + \cdots + X_d^2\right) \cdot y^2\right), \end{aligned}$$

where the last equation holds because $X_1$ is symmetric around 0, so $X_1^2$ is independent from $X_1 > 0$.

Now, $X_1^2 \sim \chi^2(1)$ and $Z := X_2^2 + \cdots + X_d^2 \sim \chi^2(d-1)$ are independent, chi-square distributed random variables and it is

$$\begin{aligned} \mathrm{P}(Y_1 > y) &= \frac{1}{2} \cdot \mathrm{P}\left(X_1^2\left(1-y^2\right) > Z \cdot y^2\right) \\ &= \frac{1}{2} \cdot \mathrm{P}\left(\frac{X_1^2}{Z} > \frac{y^2}{1-y^2}\right) \\ &= \frac{1}{2} \cdot \mathrm{P}\left(\frac{X_1^2/1}{Z/(d-1)} > \frac{(d-1)\,y^2}{1-y^2}\right). \end{aligned}$$

Since $\frac{X_1^2/1}{Z/(d-1)}$ is F-distributed with parameters 1 and $d-1$, we have

$$\begin{aligned}
1 - F_{Y_1}(y) &= \mathrm{P}(Y_1 > y) = \frac{1}{2} \cdot \mathrm{P}\left(\frac{X_1^2/1}{Z/(d-1)} > \frac{(d-1)\,y^2}{1-y^2}\right) \\
&= \frac{1}{2} \cdot \left(1 - F_F\left(\frac{(d-1)\,y^2}{1-y^2}; 1, d-1\right)\right) \\
&= \frac{1}{2} - \frac{1}{2} \cdot F_F\left(\frac{(d-1)\,y^2}{1-y^2}; 1, d-1\right),
\end{aligned}$$

or equivalently

$$F_{Y_1}(y) = \frac{1}{2} + \frac{1}{2} \cdot F_F\left(\frac{(d-1)\,y^2}{1-y^2}; 1, d-1\right).$$

Here $F_F(x; k, l) = I_{\frac{kx}{kx+l}}\left(\frac{k}{2}, \frac{l}{2}\right)$ is the cumulative distribution function (CDF) of an F-distributed random variable with parameters $k$ and $l$. Moreover, $I_z(a, b)$ is the regularized incomplete beta function, that is

$$I_z(a, b) = \frac{1}{B(a, b)} \int_0^z t^{a-1}(1-t)^{b-1}\,dt,$$

where $B(a, b) = \int_0^1 t^{a-1}(1-t)^{b-1}\,dt$ is the beta function. For more information on the F-distribution and the derivation of the formulas used, see chapter 16 in [16].

For negative $y \in (-1, 0)$, we can compute

$$\begin{aligned}
F_{Y_1}(y) &= 1 - F_{-Y_1}(-y) \\
&= 1 - F_{Y_1}(-y) \\
&= \frac{1}{2} - \frac{1}{2} \cdot F_F\left(\frac{(d-1)\,y^2}{1-y^2}; 1, d-1\right),
\end{aligned}$$

using the above result for $-y > 0$.

For $x = \frac{(d-1)y^2}{1-y^2}$, $k = 1$ and $l = d-1$, we have

$$\begin{aligned}
\frac{kx}{kx+l} &= \frac{\frac{(d-1)y^2}{1-y^2}}{\frac{(d-1)y^2}{1-y^2} + (d-1)} \\
&= \frac{y^2}{y^2 + (1-y^2)} \\
&= y^2.
\end{aligned}$$

Using the above results, we obtain for all $y \in (-1, 1)$

$$\begin{aligned}
F_F\left(\frac{(d-1)\,y^2}{1-y^2}; 1, d-1\right) &= I_{y^2}\left(\frac{1}{2}, \frac{d-1}{2}\right) \\
&= \left(\int_0^1 t^{-\frac{1}{2}}(1-t)^{\frac{d-3}{2}}\,dt\right)^{-1} \int_0^{y^2} t^{-\frac{1}{2}}(1-t)^{\frac{d-3}{2}}\,dt \\
&= \left(\int_0^1 s^{-1}\left(1-s^2\right)^{\frac{d-3}{2}} \cdot 2s\,ds\right)^{-1} \int_0^{|y|} s^{-1}\left(1-s^2\right)^{\frac{d-3}{2}} \cdot 2s\,ds \\
&= \left(\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds\right)^{-1} \int_0^{|y|} \left(1-s^2\right)^{\frac{d-3}{2}} ds
\end{aligned}$$

by the substitution $t = \phi(s) = s^2$. Since the integrand $t^{-\frac{1}{2}}(1-t)^{\frac{d-3}{2}}$ is not continuous at 0, we are not allowed to extend $\phi$ to negative number or 0 and therefore we get the absolute value of $y$. Hence, for negative $y$, we have

$$\begin{aligned} F_{Y_1}(y) &= \frac{1}{2} - \frac{1}{2} \cdot F_F\left(\frac{(d-1)y^2}{1-y^2}; 1, d-1\right) \\ &= \frac{1}{2} - \frac{1}{2} \cdot \left(\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds\right)^{-1} \int_0^{-y} \left(1-s^2\right)^{\frac{d-3}{2}} ds \\ &= \frac{1}{2} + \frac{1}{2} \cdot \left(\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds\right)^{-1} \int_0^{y} \left(1-s^2\right)^{\frac{d-3}{2}} ds, \end{aligned}$$

which is the same result we obtain for positive $y$. Lastly, it is

$$\begin{aligned} &\frac{1}{2} + \frac{1}{2} \cdot \left(\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds\right)^{-1} \int_0^{y} \left(1-s^2\right)^{\frac{d-3}{2}} ds \\ =&\frac{1}{2} + \frac{1}{2} \cdot \frac{\int_0^y \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds} \\ =&\frac{1}{2} + \frac{1}{2} \cdot \left(1 - \frac{\int_y^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}\right) \\ =&1 - \frac{1}{2} \cdot \frac{\int_y^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds} \end{aligned}$$

□

**Theorem 6.4.** *Let $Y^1, \ldots, Y^n$ be independent, uniformly distributed random variables on the $(d-1)$-sphere. Let $p$ be a fixed point on the $(d-1)$-sphere and let*

$$M = \min_{1 \le i \le n} \left(D_C\left(p, Y^i\right)\right).$$

*Then, the distribution function of $M$ is*

$$F_M(y) = 1 - F_{Y_1}(1-y)^n \tag{6.2}$$

$$= 1 - \left(1 - \frac{1}{2} \cdot \frac{\int_{1-y}^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}\right)^n \tag{6.3}$$

*for $y \in (0,2)$.*

*Proof.* Without loss of generality, we can assume that $p$ is the north pole. Thus, we have

$$D_C\left(p, Y^i\right) = 1 - Y_1^i,$$

where $Y_1^i$ denotes the first coordinate of $Y^i$ and it is

$$M = \min_{1 \le i \le n} \left(1 - Y_1^i\right) = 1 - \max_{1 \le i \le n} \left(Y_1^i\right) \in [0,2].$$

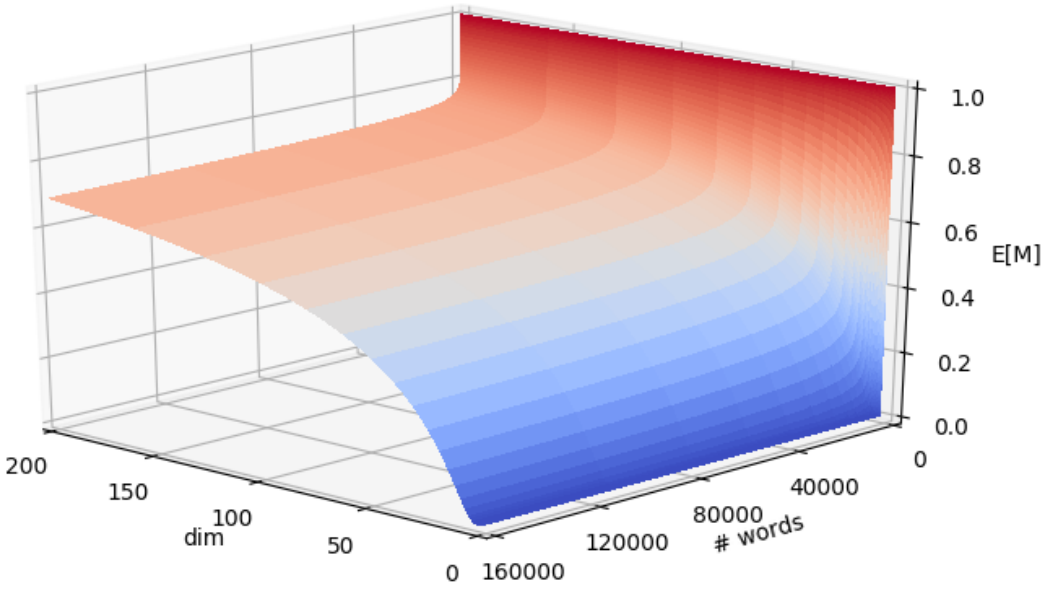



Figure 6.1: The expectation of the distance to the closest word depending on the embedding dimension and the number of words

Using Theorem 6.3 for the distribution of $Y_1^i$, we can compute the distribution function of $M$ as follows.

$$\begin{aligned}
F_M(y) &= \mathrm{P}\left(1 - \max_i \left(Y_1^i\right) \leq y\right) \\
&= \mathrm{P}\left(\max_i \left(Y_1^i\right) \geq 1 - y\right) \\
&= 1 - \mathrm{P}\left(\max_i \left(Y_1^i\right) \leq 1 - y\right) \\
&= 1 - \mathrm{P}\left(Y_1^1 \leq 1 - y\right) \cdot \ldots \cdot \mathrm{P}\left(Y_1^n \leq 1 - y\right) \\
&= 1 - F_{Y_1}(1 - y)^n \\
&= 1 - \left(1 - \frac{1}{2} \cdot \frac{\int_{1-y}^1 \left(1 - s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1 - s^2\right)^{\frac{d-3}{2}} ds}\right)^n
\end{aligned}$$

□

**Corollary 6.2.** *Let $Y^1, \ldots, Y^n$ be independent, uniformly distributed random variables on the $(d-1)$-sphere. Let $p$ be a fixed point on the $(d-1)$-sphere and let*

$$M = \min_{1 \leq i \leq n} \left(D_C\left(p, Y^i\right)\right).$$

*Then, we have*

$$\mathbb{E}[M] = \int_{-1}^1 \left[1 - \frac{1}{2} \cdot \frac{\int_y^1 \left(1 - s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1 - s^2\right)^{\frac{d-3}{2}} ds}\right]^n dy.$$

*Proof.* We use the derivative of the distribution function, integration by parts and integration by substitution to compute the expectation of $M$. We have

$$\begin{aligned}
\mathbb{E}[M] &= \int_0^2 y \cdot \frac{\partial}{\partial y} \left(1 - F_{Y_1}(1-y)^n\right) dy \\
&= \left[y \cdot \left(1 - F_{Y_1}(1-y)^n\right)\right]_0^2 - \int_0^2 1 - F_{Y_1}(1-y)^n \, dy \\
&= 2 \cdot \left(1 - F_{Y_1}(-1)^n\right) - \int_{-1}^1 1 - F_{Y_1}(y)^n \, dy \\
&= 2 - \int_{-1}^1 1 - F_{Y_1}(y)^n \, dy \\
&= \int_{-1}^1 1 \, dy - \int_{-1}^1 1 - F_{Y_1}(y)^n \, dy \\
&= \int_{-1}^1 F_{Y_1}(y)^n \, dy \\
&= \int_{-1}^1 \left[1 - \frac{1}{2} \cdot \frac{\int_y^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}{\int_0^1 \left(1-s^2\right)^{\frac{d-3}{2}} ds}\right]^n dy.
\end{aligned}$$

□

# 7 Evaluation

In this thesis, we have learned four different methods for finding word embeddings: two neural networks (SG and SGNS) and SVD based on the objective functions of each of the neural networks. However, the SG neural network was not feasible and we were not able to train it due to the large number of weights that the objective function depends on (for more details see the beginning of section 3.2).

Using the SGNS neural network and SVD on the matrices $M^{\mathrm{SG}+}$ and $M^{\mathrm{SGNS}+}$, we obtain three sets of word vectors. In this chapter, we will evaluate them and explore which of the methods worked best and why.

The underlying dataset for the program is Wikimedia dump[1], which currently contains about 4.6 million English Wikipedia articles. After ignoring words that appeared less than 300 times in the corpus, we obtain a word and context vocabulary of 159,862 different words. Using window size 2 to each side results in about 9.7 billion word-context samples. For the embedding dimension we choose $d = 200$. For the negative sampling probability $\mathrm{P_D}(c)$, we actually used the probability that raises the word counts to the power of $\frac{3}{4}$, that is

$$\mathrm{P_D}(c) = \frac{\#(c)^{\frac{3}{4}}}{\sum_{c' \in V_C} \#(c')^{\frac{3}{4}}},$$

instead of the unigram distribution. In [11], Mikolov et al. explain about this probability that they "investigated a number of choices [...] and found that the unigram distribution [...] raised to the 3/4rd power [...] outperformed significantly the unigram and the uniform distributions."

Our models are available online[2]. For an input word or equation, the web application will output the closest words and their distances to the input. The expectation of the distance to the closest word we discussed in section 6.2 for dimension $d = 200$ and our vocabulary size is about 0.69. This means that words with distance less than 0.69 can be considered relevant to the query.

## 7.1 Optimizing the objective function

First, we want to evaluate how good each model is at maximizing the objective function. We will analyze the models that are based on the negative sampling method first and then consider the skip-gram models (no negative sampling).

[1] `https://dumps.wikimedia.org/enwiki/latest/`
[2] `http://word2vec.cs.uni-freiburg.de`

### 7.1.1 SGNS

In chapter 4, we discussed the optimum and found that SGNS's objective function has a maximum for

$$\begin{aligned}\vec{w} \cdot \vec{c} = x^*_{w,c} &= \log\left(\frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c)}\right) - \log k \\ &= \mathrm{PMI}(w,c) - \log k\end{aligned}$$

and we named the matrix containing these values $M^{\mathrm{SGNS}}$. Now, we consider the value of the objective function at these points to get the optimum value $\ell^{\mathrm{OPT}}$, that is

$$\ell^{\mathrm{OPT}} := \ell_{\mathrm{SGNS}}\left(M^{\mathrm{SGNS}}\right).$$

We evaluate the objective function at the non-negative values, that is

$$\ell^{+} := \ell_{\mathrm{SGNS}}\left(M^{\mathrm{SGNS+}}\right)$$

as discussed in section 4.3 and at the actual dot products computed by our models, that is

$$\ell_{\mathrm{SGNS}}(W,C) = \ell_{\mathrm{SGNS}}\left(W \cdot C^T\right),$$

where $W$ and $C$ are the representation matrices computed by the models (SVD and the neural network). Table 7.1 shows the percentage of deviation from $\ell^{\mathrm{OPT}}$, that is $\frac{\ell - \ell^{\mathrm{OPT}}}{\ell^{\mathrm{OPT}}}$, where $\ell$ is the objective value for a given model.

| $k$ | $\ell^{\mathrm{OPT}}$ | $\ell^{+}$ | SVD | SGNS |
|---|---|---|---|---|
| 1 | 0% | 29.3% | 38.8% | 22.7% |
| 5 | 0% | 120.9% | 124.7% | 9.5% |
| 15 | 0% | 309.0% | 310.4% | 8.9% |

Table 7.1: **Percentage of deviation from the optimal objective value** (lower values are better). We evaluated the objective function 3.5 with four different values for $\vec{w} \cdot \vec{c}$. SVD and SGNS use the computed word and context embeddings. See the beginning of section 7.1.1 for details.

The neural network is trained to maximize the objective function, while SVD is actually optimizing a different function (see equation 5.1). However, we still only evaluated the neural network's objective in table 7.1, since we derived all our models from this function.

The table shows that setting all negative values to 0 has a great influence on the objective function. This stems from the fact that a lot of information is discarded by doing so. Moreover, the values in this column increase quickly as $k$ increases, which can be understood when looking at the optimal dot product values $x^*_{w,c} = \mathrm{PMI}(w;c) - \log k$. As $k$ increases, more and more of these values will become negative and therefore will be set to 0 in the matrix $M^{\mathrm{SGNS+}}$, which leads to higher deviation from the optimal objective value.

SVD also performs rather poorly. However, we propose this deviation from the optimum does not stem from the SVD itself, but from the fact that this model is based on the matrix $M^{\mathrm{SGNS+}}$, which deviated from the optimum a lot already. Clearly, there is no reason why SVD should be better at optimizing the objective than the $M^{\mathrm{SGNS+}}$ values.

The neural network is by far the best model. This is because it is the only model whose main goal is to optimize the objective function. Be reminded, however, that the optimum cannot be achieved in general, since there are no matrices $W$ and $C$ such that $W \cdot C^T = M^{\text{SGNS}}$. The neural network seems to profit from increasing $k$, while SVD suffers from it.

**The objective values for the non-negative model** (that is $\ell^+$) differ significantly from the values that Levy and Goldberg obtain (Table 1 in [8]), which they call SPPMI ("Shifted Positive Pointwise Mutual Information"). While we already start with almost 30% for $k = 1$ and observe a considerable increase for increasing $k$ in our values, they obtain 0.00009% for $k = 1$ and their values decrease even more as $k$ increases. The following lemma identifies the deviation of the objective value when using $\max\left(x^*_{w,c}, 0\right)$ instead of $x^*_{w,c}$.

**Lemma 7.1.** *It is* $\ell^+ = \ell^{OPT} + R$, *where* $R = R_1 + R_2$ *with*

$$R_1 = - \sum_{\substack{(w,c) \text{ with} \\ \#(w,c)=0}} b_{w,c} \cdot \log 2,$$

$$R_2 = - \sum_{\substack{(w,c) \text{ with} \\ 0<\#(w,c)<b_{w,c}}} \left( b_{w,c} \cdot \log\left( \frac{2 \cdot b_{w,c}}{b_{w,c} + \#(w,c)} \right) + \#(w,c) \cdot \log\left( \frac{2 \cdot \#(w,c)}{b_{w,c} + \#(w,c)} \right) \right),$$

*where* $b_{w,c} = k \cdot \#(w) \cdot \frac{\#(c)}{|D|}$.

| $k$ | $\ell_{\text{OPT}}$ | $R_1$ | $R_2$ | $\ell_+$ |
|---|---|---|---|---|
| 1 | $-8.708$ | $-1.436$ | $-1.117$ | $-11.260$ |
| | 100% | 16.49% | 12.83% | 129.31% |
| 5 | $-17.927$ | $-7.208$ | $-14.486$ | $-39.592$ |
| | 100% | 40.21% | 80.81% | 220.85% |
| 15 | $-26.198$ | $-22.398$ | $-59,406$ | $-107.141$ |
| | 100% | 85.50% | 226.76% | 408.97% |

Table 7.2: $R_1$ and $R_2$, the two parts that make up the difference between $\ell^{\text{OPT}}$ and $\ell^+$. $\ell^{\text{OPT}} + R_1 + R_2$ should equal $\ell^+$. $R_2$ is an approximation. The absolute values are in billions. Percentage shows percentage of the optimum.

*Proof.* Recall from equation (4.3) that

$$\ell_{w,c}(x) = \#(w,c) \cdot \log \sigma(x) + b_{w,c} \cdot \log \sigma(-x)$$

and from Theorem 4.1 that $x^*_{w,c} = \arg\max_x \left(\ell_{w,c}(x)\right) = \log \frac{\#(w,c)}{b_{w,c}}$. Let

$$\ell^{\text{OPT}}_{w,c} = \ell_{w,c}\left(x^*_{w,c}\right),$$

which means that $\ell^{\text{OPT}} = \sum_{w,c} \ell^{\text{OPT}}_{w,c}$. Similarly, we define

$$\ell^+_{w,c} = \ell_{w,c}\left(\max\left(x^*_{w,c}, 0\right)\right)$$

so that $\ell^+ = \sum_{w,c} \ell^+_{w,c}$. Using this notation, it is

$$\begin{aligned} R &= \ell^+ - \ell^{\mathrm{OPT}} \\ &= \sum_{w,c} \ell^+_{w,c} - \sum_{w,c} \ell^{\mathrm{OPT}}_{w,c} \\ &= \sum_{w,c} \left( \ell^+_{w,c} - \ell^{\mathrm{OPT}}_{w,c} \right) \end{aligned}$$

and we have to compute the difference $\ell^+_{w,c} - \ell^{\mathrm{OPT}}_{w,c}$ for each word-context pair $(w, c)$. We distinguish three cases.

- If $\#(w,c) \geq b_{w,c}$, then $x^*_{w,c} \geq 0$, so $\ell^+_{w,c} = \ell_{w,c}\left(x^*_{w,c}\right) = \ell^{\mathrm{OPT}}_{w,c}$.
- Now, assume that $\#(w,c) = 0$. This means that $x^*_{w,c} = -\infty$ and it is

$$\begin{aligned} \ell^{\mathrm{OPT}}_{w,c} &= \lim_{x \to -\infty} \Big( 0 \cdot \log \sigma(x) + b_{w,c} \cdot \log \sigma(-x) \Big) \\ &= b_{w,c} \cdot \lim_{x \to -\infty} \log \sigma(-x) \\ &= 0. \end{aligned}$$

  Furthermore, it is

$$\begin{aligned} \ell^+_{w,c} &= \ell_{w,c}(0) \\ &= b_{w,c} \cdot \log(\sigma(0)) \\ &= -b_{w,c} \cdot \log 2 \\ &= \ell^{\mathrm{OPT}}_{w,c} - b_{w,c} \cdot \log 2. \end{aligned}$$

- For the last case, we consider word-context pairs $(w, c)$ with $0 < \#(w,c) < b_{w,c}$. This means that $x^*_{w,c}$ is negative again and we have

$$\begin{aligned} \ell^+_{w,c} &= \ell_{w,c}(0) \\ &= -\#(w,c) \cdot \log 2 - b_{w,c} \cdot \log 2 \\ &= -\#(w,c) \cdot \log 2 - b_{w,c} \cdot \log 2 + \ell^{\mathrm{OPT}}_{w,c} \\ &\qquad - \#(w,c) \cdot \log \sigma\left( \log\left( \frac{\#(w,c)}{b_{w,c}} \right) \right) - b_{w,c} \cdot \log \sigma\left( -\log\left( \frac{\#(w,c)}{b_{w,c}} \right) \right) \\ &= \ell^{\mathrm{OPT}}_{w,c} - b_{w,c} \cdot \log 2 - \#(w,c) \cdot \log 2 \\ &\qquad - \#(w,c) \cdot \log \frac{1}{1 + \frac{b_{w,c}}{\#(w,c)}} - b_{w,c} \cdot \log \frac{1}{1 + \frac{\#(w,c)}{b_{w,c}}} \\ &= \ell^{\mathrm{OPT}}_{w,c} - b_{w,c} \cdot \left( \log 2 + \log\left( \frac{b_{w,c}}{b_{w,c} + \#(w,c)} \right) \right) \\ &\qquad - \#(w,c) \cdot \left( \log 2 + \log\left( \frac{\#(w,c)}{\#(w,c) + b_{w,c}} \right) \right) \\ &= \ell^{\mathrm{OPT}}_{w,c} - b_{w,c} \cdot \log\left( \frac{2 \cdot b_{w,c}}{b_{w,c} + \#(w,c)} \right) - \#(w,c) \cdot \log\left( \frac{2 \cdot \#(w,c)}{b_{w,c} + \#(w,c)} \right). \end{aligned}$$

Combining the three cases yields the desired result. □

Now, the $\ell^+$ value in Table 7.1 is equal to $\frac{R_2}{\ell^{\mathrm{OPT}}} + \frac{R_1}{\ell^{\mathrm{OPT}}}$. Table 7.2 presents these values (in percent). The table shows that for increasing $k$, $R_1$ and $R_2$ do not only decrease, but they do so faster than $\ell^{\mathrm{OPT}}$, i.e. the fractions $\frac{R_1}{\ell^{\mathrm{OPT}}}$ and $\frac{R_2}{\ell^{\mathrm{OPT}}}$ decrease as well. The formula for $R_1$ shows that the decrease of $R_1$ with respect to $k$ is linear. Observe that Table 7.2 agrees with this fact. Moreover, we can observe that $R_2$ decreases even faster than $R_1$. We suppose that this is due to the fact that an increase in $k$ affects $R_2$ in two ways: Firstly, as $k$ increases, we have more word-context pairs with $0 < \#(w,c) < b_{w,c}$ (see Table 7.3). This means that more items are added to the sum $R_2$. Secondly, a simple derivative shows that each addend of $R_2$ is decreasing for $0 < \#(w,c) < b_{w,c}$, which we will show in the following lemma.

| $k$ | $\#(w,c) = 0$ | $0 < \#(w,c) < b_{w,c}$ | $\#(w,c) < b_{w,c}$ |
|---|---|---|---|
| 1 | 98.2471 | 0.7052 | 1.0477 |
| 5 | 98.2471 | 1.2547 | 0.4982 |
| 15 | 98.2471 | 1.4854 | 0.2675 |

Table 7.3: Proportion of word-context-pairs with different values of $\#(w,c)$ in percent.

**Lemma 7.2.** *Each addend of $R_2$ is strictly decreasing for increasing $k$.*

*Proof.* Let

$$f(b) = -b \cdot \log\left(\frac{2b}{b + \#(w,c)}\right) - \#(w,c) \cdot \log\left(\frac{2 \cdot \#(w,c)}{b + \#(w,c)}\right).$$

We prove the lemma by deriving $f$ with respect to $b$. This is sufficient since $\frac{\partial}{\partial k} b_{w,c} = 1$. It is

$$\begin{aligned}
f'(b) &= -\log\left(\frac{2b}{b + \#(w,c)}\right) - b \cdot \frac{b + \#(w,c)}{2b} \cdot \frac{2\left(b + \#(w,c)\right) - 2b}{\left(b + \#(w,c)\right)^2} \\
&\quad - \#(w,c) \cdot \frac{b + \#(w,c)}{2 \cdot \#(w,c)} \cdot \frac{-2 \cdot \#(w,c)}{\left(b + \#(w,c)\right)^2} \\
&= -\log\left(\frac{2b}{b + \#(w,c)}\right) - \frac{\#(w,c)}{b + \#(w,c)} + \#(w,c) \cdot \frac{1}{b + \#(w,c)} \\
&= -\log\left(\frac{2b}{b + \#(w,c)}\right) \\
&< 0
\end{aligned}$$

for $b > \#(w,c)$ and therefore, $f$ is decreasing for the corresponding values of $b$. □

### 7.1.2 Skip-Gram

We evaluated the objective function of the SG model similar to the SGNS model. Since the SG neural network is not feasible, we only evaluated the SVD model. Let $\ell^{\mathrm{OPT}} = \ell_{\mathrm{SG}}\left(M^{\mathrm{SG}}\right)$ be the optimal value of the objective function and let $\ell^+ = \ell_{\mathrm{SG}}\left(M^{\mathrm{SG+}}\right)$ be the value for the non-negative dot products as explained in section 4.3.

Table 7.4 shows that the error that arises from setting negative values to 0 is considerably smaller than in the SGNS model. One reason for this is that we only change dot products

| $\ell^{\text{OPT}}$ | $\ell^+$ | SVD |
|---|---|---|
| 0% | 5.7% | 25.1% |

Table 7.4: **Percentage of deviation from the optimal objective value** (that is $\ell^{\text{OPT}}$). We evaluated the objective function 4.5 with three different values for the matrix $X$. SVD uses the computed word and context embeddings ($W \cdot C^T$).

of word-context pairs that never appeared in the text (i.e. with $\#(w, c) = 0$). While the SVD values for the SGNS model were fairly close to $\ell^+$ (see Table 7.1), the deviation from the optimum of the SVD here does not stem from the loss of information by setting too many values to 0. It is possible that there are no representation matrices $W$ and $C$ for dimension 200 that perform as well as $\ell^+$, which would explain the deviation in the SVD model.

## 7.2 Word similarity tasks

**Experiment.** We use two datasets to evaluate the models on word similarity tasks. Both datasets contain word pairs and a human-assigned similarity score for each pair. For instance, the pair 'sun' and 'sunlight' got a very high score, while the pair 'bakery' and 'zebra' got a very low score. We then measure the strength of association between the human-assigned scores and the cosine similarity of our vectors using the *Spearman's rank correlation coefficient* (or Spearman's rho). Let $X_i$ and $Y_i$ be the human and computer assigned scores respectively. Then, the Spearman correlation is defined as

$$\rho_S = \frac{\operatorname{cov}(\operatorname{rg}(X), \operatorname{rg}(Y))}{\sigma(\operatorname{rg}(X))\,\sigma(\operatorname{rg}(Y))},$$

where $\operatorname{rg}(X)$ and $\operatorname{rg}(Y)$ are the rank variables, cov denotes the covariance and $\sigma$ is the standard deviation. The Spearman's rho takes values between $-1$ and $+1$, which only depend on the rank (i.e. the order) of the scores rather than the values of the scores themselves. A positive value indicates that as values of one variable increase, the other value tends to increase as well, while a negative value indicates that as values of one variable increase, the other variable tends to decrease. Values around 0 indicate that there is no such correlation.

**Results.** Table 7.5 shows the performance of our models for the WordSim353 dataset [5], which contains 353 word pairs, and the MEN dataset [3], which contains 3000 word pairs.

| | **WordSim353** | | **MEN** | |
|---|---|---|---|---|
| k | Neural Network | SVD | Neural Network | SVD |
| 0 | - | 0.601 | - | 0.655 |
| 1 | 0.524 | 0.613 | 0.588 | 0.700 |
| 5 | 0.658 | 0.536 | 0.712 | 0.669 |
| 15 | 0.644 | 0.400 | 0.681 | 0.606 |

Table 7.5: Spearman's coefficient of different word representations on two datasets. The row for $k = 0$ represents skip-gram (without negative sampling).

**Interpretation.** This experiment provides more evidence that SVD suffers from increasing $k$, a result we also discovered in the last section when it comes to optimizing the objective function. While the neural network outperforms the SVD model for $k = 5$ and $k = 15$, SVD performs better for $k = 1$. The table suggests that as $k$ gets too large, the quality of the vectors will start declining, even for the neural model. Choosing too many negative samples results in less emphasis on the positive samples, which might affect the quality of the vectors in a negative way after a certain point.

## 7.3 Analogy tasks

**Experiment.** We evaluate our models on two datasets containing analogy tasks of the form "$a$ is to $a^*$ as $b$ is to $b^*$". This means that in our model, the vector between $a$ and $a^*$ should be the same as the vector between $b$ and $b^*$, in other words: $a^* - a = b^* - b$. The word $b^*$ is hidden and the models try to find the correct answer by computing the vector addition $\tilde{b} = a^* - a + b$. We measure the percentage of questions for which the closest word to $\tilde{b}$ is $b^*$. In the course of this, we ignore the three words $a$, $a^*$ and $b$.

The *Syntactic* dataset [12] contains 8000 syntactic analogy questions, such as "*good* is to *best* as *smart* is to *smartest*". The second dataset, the *Mixed* dataset [10], contains about 19500 analogy questions of both semantic and syntactic kind. An example for a semantic analogy is "*Paris* is to *France* as *Berlin* is to *Germany*". After ignoring instances with out-of-vocabulary words, our models were tested on the remaining 5732 questions[3] in the *Syntactic* dataset and 19308 questions in the *Mixed* dataset.

**Results.** Table 7.6 shows precision at 1, which is the percentage of questions for which the model gave the correct answer ($b^*$).

| | **Mixed** | | **Syntactic** | |
|---|---|---|---|---|
| k | Neural Network | SVD | Neural Network | SVD |
| 0 | - | 26.84 | - | 28.70 |
| 1 | 27.28 | 30.55 | 32.29 | 19.59 |
| 5 | 50.95 | 12.03 | 51.03 | 5.67 |
| 15 | 53.20 | 5.86 | 47.89 | 1.36 |

Table 7.6: Percentage of correct answers on two word analogy tasks. The row for $k = 0$ represents skip-gram (without negative sampling).

**Interpretation.** The neural networks outperform the SVD models by a lot for all but one category. For the *Mixed* dataset, SVD performs slightly better for $k = 1$. Again, there is an evident decrease in precision for the SVD model for increasing $k$. Moreover, we can observe a difference in performance between the two datasets. While the neural networks achieve similar results for both datasets, the SVD model performs a lot better on the *Mixed* dataset than on the *Syntactic* dataset, with an exception for the model without

[3]The large amount of out-of-vocabulary words mainly comes from the category *Possessive/Non-possessive nouns*. Because of the apostrophes, none of the possessive nouns is in the vocabulary. Thus, all 1000 questions in this category are being ignored. In addition, many of the comparative and superlative verbs are ouf-of-vocabulary.

negative sampling ($k = 0$). This could mean that SVD is better at semantic tasks than syntactic ones.

Figure 7.1 shows the performance of each model for the different categories of the *Mixed* dataset. There are vast discrepancies between the different categories. A possible explanation for the low values in the categories *currency* (C03), *opposite* (C07) and *superlative* (C09) is that they contain many rare words. In fact, these are the only categories with out-of-vocabulary words, which points to the fact that many of the words that were found in the vocabulary are rare words as well. This clearly affects the quality of the vectors. The figure shows that the neural networks for $k = 5$ and $k = 15$ perform similarly well and they are superior to the other models in almost all categories. The opposite is true for the corresponding SVD models. They perform more poorly than the other models in most categories.

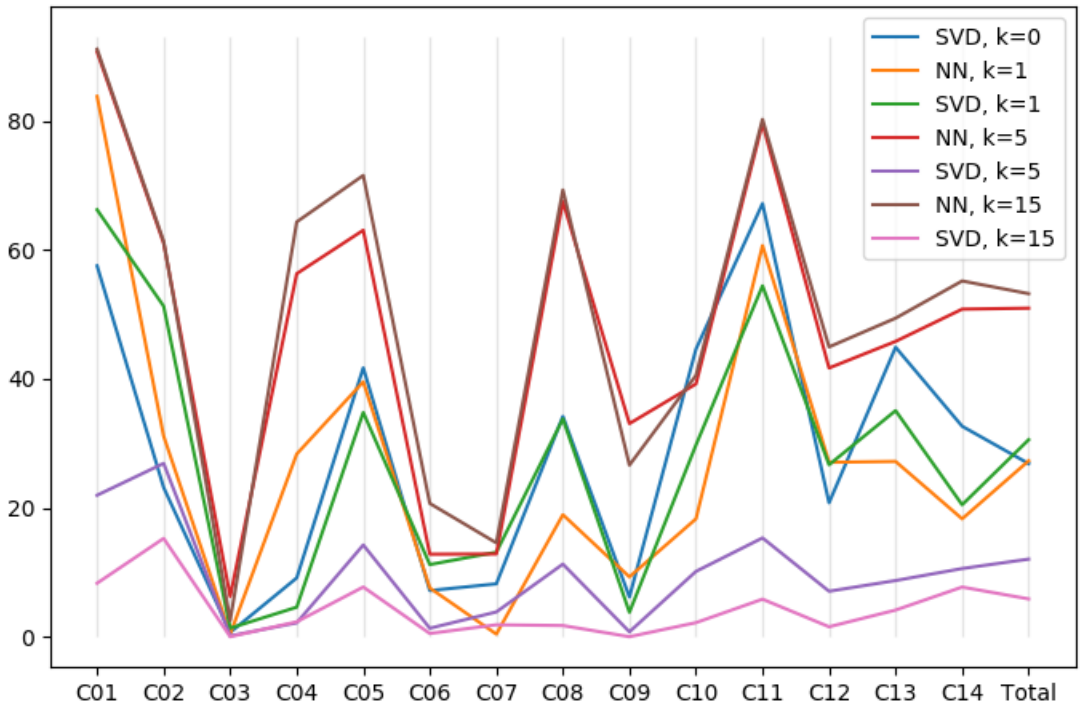



Figure 7.1: Percentage of correct answers that different models gave; question categories on the x-axis. For more information about the dataset see [10]. The categories are: **C01**: Common capital city, **C02**: All capital cities, **C03**: Currency, **C04**: City-in-state, **C05**: Man-Woman, **C06**: Adjective to adverb, **C07**: Opposite, **C08**: Comparative, **C09**: Superlative, **C10**: Present Participle, **C11**: Nationality adjective, **C12**: Past tense, **C13**: Plural nouns, **C14**: Plural verbs

# 8 Conclusion and Future Work

We analyzed and evaluated two approaches to retrieve word vectors, neural networks and SVD. We showed that, if the embedding dimension is high enough, both models maximize the same objective function. However, we ran into a problem with word-context pairs which never appear in the same context together (this was the case for more than 98% of our word-context pairs). In order to meet this problem, we adjusted some values by setting them to 0. Our data shows that these changes caused a downward change in the quality of the vectors. This loss of information, which stems from setting values to 0, increases as $k$, the number of negative samples, increases.

Motivated by these findings, we suggest applying SVD on a different matrix. Let $s$ be the smallest value in $M^{\text{SGNS}}$ (apart from those that are $-\infty$), that is

$$s = \min_{(w,c)\in D} \left( \log \left( \frac{\#(w,c) \cdot |D|}{\#(w) \cdot \#(c) \cdot k} \right) \right).$$

Then, shifting the matrix by $-s$ and setting the $-\infty$-values to 0 results in a sparse, non-negative matrix, which we will denote by $S$. This way, we do not lose information for increasing $k$. Table 8.1 shows that factorizing $S$ produces quite good results, especially compared to the the old SVD model for $k = 15$. As future work, we suggest further investigation of the characteristics, the advantages and disadvantages of this matrix.

| factorizing | k | WordSim353 | MEN | Mixed | Syntactic |
|---|---|---|---|---|---|
| $M^{\text{SGNS+}}$ | 1 | 0.613 | 0.700 | 30.55% | 19.59% |
| $M^{\text{SGNS+}}$ | 15 | 0.400 | 0.606 | 5.86% | 1.36% |
| $S$ | 15 | 0.566 | 0.654 | 32.83% | 26.08% |

Table 8.1: Results of the newly proposed SVD method (factorizing $S$) set against the SVD method described in this thesis (factorizing $M^{\text{SGNS+}}$).

Another path to explore further is the skip-gram neural network. It would be interesting to see if negative sampling improves not just the running time, but also the quality of the vectors and how well SVD is doing compared to the network without negative sampling.

Lastly, we suggest exploring hyperparameter optimization more. We suppose that this has a great effect on the quality of the vectors and more investigation might lead to much better results. For instance, we did not discuss the window size in this thesis. Likewise, we would like to know the qualitative behavior of our models for different embedding dimensions.

# 9 Declaration

I hereby declare that I am the sole author and composer of my thesis and that no other sources or learning aids, other than those listed, have been used. Furthermore, I declare that I have acknowledged the work of others by providing detailed references of said work. I hereby also declare that my thesis has not been prepared for another examination or assignment, either wholly or excerpts thereof.

| | |
|---|---|
| Place, Date | Theresa Klumpp |

# 10 Acknowledgments

First and foremost, I give praise to god. I thank him for providing the following people who where crucial to finishing this thesis.

I would like to thank Prof. Dr. Hannah Bast for supervising this thesis – even though I am a student of mathematics – and providing a delightful topic. She had a lot of patience when there was a lack of computer skills on my side. Thank you for the fantastic guidance and taking a lot of time to work through the main challenges with me.

Further, I am grateful to Prof. Dr. Peter Pfaffelhuber, who willingly agreed to examine this thesis and who supervised the mathematical parts with excellent insight.

I also want to express my sincere gratitude to Johannes Müller for proof-reading and giving me deeply insightful advice and to Lukas Weith for his amazingly thorough proof-reading on language. I know this has not been the most interesting piece of literature for you...

Last, but certainly not least, my gratitude goes to my parents, whose moral and financial support and continuous encouragement and trust have enabled me to pursue my studies and finish this thesis.

# Bibliography


[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

[2] Avrim Blum, John Hopcroft, and Ravindran Kannan. Foundations of data science. *Vorabversion eines Lehrbuchs*, 2016.

[3] Elia Bruni, Gemma Boleda, Marco Baroni, and Nam-Khanh Tran. Distributional semantics in technicolor. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 136–145. Association for Computational Linguistics, 2012.

[4] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[5] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. *ACM Transactions on information systems*, 20(1):116–131, 2002.

[6] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

[7] Rémi Lebret and Ronan Collobert. Word emdeddings through hellinger pca. *arXiv preprint arXiv:1312.5542*, 2013.

[8] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185, 2014.

[9] Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208, 1996.

[10] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[11] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[12] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, 2013.

[13] Andriy Mnih and Geoffrey E Hinton. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088, 2009.

[14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[15] Douglas LT Rohde, Laura M Gonnerman, and David C Plaut. An improved model of semantic similarity based on lexical co-occurrence. *Communications of the ACM*, 8(627-633):116, 2006.

[16] Christian Walck. Statistical distributions for experimentalists. *Particle Physics Group*, 2007.