UPPSALA
UNIVERSITET

WILL SVENSKA AKADEMIENS ORDLISTA IMPROVE SWEDISH WORD EMBEDDINGS?

Submitted by
Ellen Ahlberg

A thesis submitted to the Department of Statistics
in partial fulfillment of the requirements
for a two-year Master of Science degree in Statistics
in the Faculty of Social Sciences

Supervisors
Måns Magnusson
Väinö Yrjänäinen

Spring, 2022

ABSTRACT

Unsupervised word embedding methods are frequently used for natural language processing applications. However, the unsupervised methods overlook known lexical relations that can be of value to capture accurate semantic word relations. This thesis aims to explore if Swedish word embeddings can benefit from prior known linguistic information. Four knowledge graphs extracted from Svenska Akademiens ordlista (SAOL) are incorporated during the training process using the Probabilistic Word Embeddings with Laplacian Priors (PELP) model. The four implemented PELP models are compared with baseline results to evaluate the use of side information. The results suggest that various lexical relations in SAOL are of interest to generate more accurate Swedish word embeddings.

Contents

1 Introduction

One major and fast-growing branch of artificial intelligence is Natural Language Processing (NLP). NLP refers to the concept of giving computers the ability to parse and generate natural language. As the internet grows exponentially, the need for automatically identifying texts grows with it. NLP combines the fields of linguistics and computer science to develop methods for understanding and analyzing large repositories of unstructured text data. Progression in machine learning techniques and improvements in computer performance have paved the way for state-of-the-art NLP models (Goldberg, 2017). The advanced methods can be applied in several areas, including machine translation, sentiment analysis, and text classification (B. Wang, Wang, Chen, Wang, & Kuo, 2019; Goldberg, 2017).

Since computers are only capable of assimilating numbers, one fundamental factor of NLP is to represent words in such a way that they can be processed by a computer. One approach is to convert each unique word in the vocabulary to a word embedding, which is a floating-point vector. The objective is to retain the meaning of a word in the embedding so that words with similar meanings have similar representations in the vector space (Goldberg, 2017). The complexity of NLP is that natural language is a multifaceted phenomenon that goes beyond sequences of characters, words, and sentences. Human language consists of extensive vocabularies where each word by itself does not have much significance in the vector space. In addition, human language also tends to be ambiguous and ever-changing. To derive meaningful information from natural language data, it is necessary to understand the overall structure and context of the text.

Most word embedding methods today are based on the distributional hypothesis. According to this hypothesis, the meaning of a word is determined by the context in which it is used (Harris, 1954; Firth, 1957). Large text corpora can be utilized by unsupervised learning algorithms to detect distributional properties, such as co-occurrence patterns and common neighbors between words (Goldberg, 2017). The distributional properties can be used to quantify semantic and syntactic meaning. However, by placing all attention on the distributional hypothesis, all previous lexical knowledge will be overlooked (Sahlgren, 2008). The main objective of this thesis is to integrate both distributional properties and prior lexical knowledge to refine Swedish word embeddings.

1.1 Background

Depending on the word embedding method, words will acquire different vector representations. One simple way of representing words as vectors is through one-hot encoding. Each unique word in a vocabulary of size V is associated with an integer index. The vectors are V-dimensional, with all zero entries except one non-zero entry at index i for word w_i (Chollet & Allaire, 2018). While the one-hot encoding method is straightforward, for a large vocabulary, the dimensionality of the vectors becomes cumbersome. Moreover, semantic qualities cannot be distinguished amongst the sparse vectors.

Ideally, the vectors should hold both semantic and syntactic features (Goldberg, 2017). A more functional representation of words is through dense continuous vectors, i.e., word embeddings. Word embeddings are often obtained from unsupervised machine learning approaches such as neural networks (Levy, Goldberg, & Dagan, 2015). During the learning phase of neural networks, words are mapped to mathematical objects in a low-dimensional space (Goldberg, 2017). Directions in the vector space should be relevant, and the distance between the learned vectors can be compared to the semantic distance between the corresponding words. Words or sentences from large text corpora are taken as input in the neural network model. In agreement with the distributional hypothesis, neighboring words will determine the representation of a word. Consider the following sequence:

the *lazy* **dog** *barked* *loudly*

The word **dog** is examined as the target word. Neighboring words within a specified context window contribute to semantic information regarding the target word. For the given example, the context window is set to two. This implies that two words before the target word (*the, lazy*) and two words after the target word (*barked, loudly*) are considered for each input sequence. When feeding the network with sequences of words from a large text corpus, co-occurrence patterns can be identified (Goldberg, 2017). For example, the network can recognize that neighboring words for the target word **dog** are similar to neighboring words for the word **canine** (Goldberg, 2017).

More recent machine learning techniques are exploring the effects of including side information in the training process. Side information implies data beyond the training data, which can be in the form of relational facts, logical rules, or knowledge graphs (Jonschkowski, Höfer, & Brock, 2015). By including prior knowledge, less data is required in the learning process, and known relations between entities can be employed (von Rueden et al., 2021; Diligenti, Roychowdhury, & Gori, 2017). For NLP methods, prior knowledge of word relations can be utilized to refine unsupervised embedding learning or to specialize the generated embeddings for a specific NLP task (Kiela, Hill, & Clark, 2015). Side information in the form of knowledge graphs can be added to establish a specific semantic relation between two or more words (von Rueden et al., 2021). The side information can be utilized using several different methods. One option is to refit trained word vectors onto knowledge graphs built from semantic lexicons (Faruqui et al., 2014). Another alternative is to change the objective of the training algorithm to incorporate prior knowledge during the training process (Tissier, Gravier, & Habrard, 2017; Yu & Dredze, 2014).

There are several online lexical databases for the English language that have been specifically annotated for machine-readability. The lexical resources provide semantic and syntactic generalizations that determine word definitions. The most frequently used lexical databases include WordNet (Miller, 1995), FrameNet (Baker, Fillmore, & Lowe, 1998), and the Paraphrase Database (Ganitkevitch, Van Durme, & Callison-Burch, 2013). However, it can be argued that the lexical databases can be restrictive in terms of vocabulary size and required maintenance work (Tissier et al., 2017). Instead, online natural language dictionaries can be utilized to refine the vector space.

For Swedish, the available lexical resources are slightly more limited. *Språkbanken* (the Swedish Language Bank) is currently collecting and publishing resources for Swedish modern language technology. Since training word embeddings can be both time-consuming and computationally complex, Språkbanken provides a list of pre-trained embeddings. Currently, all pre-trained Swedish word embeddings are trained solely on large text corpora. However, Språkbanken also provides lexical semantic knowledge sources similar to the WordNet (Miller, 1995) database, such as SALDO (Borin, Forsberg, & Lönngren, 2013). Another reliable source for Swedish lexical relations is Svenska Akademiens ordlista (SAOL), which is continuously updated to provide accurate information regarding the Swedish language (Svenska Akademien, 2015).

1.2 Aim

The Swedish word embeddings available today are trained on large amounts of text data and thus solely depend on the distributional hypothesis. However, there are several Swedish lexical resources available that can be utilized to refine unsupervised embedding learning. This thesis aims to evaluate if Swedish word embeddings can benefit from prior knowledge using Svenska Akademiens ordlista (SAOL). SAOL can be viewed as side information, supplementary to the training corpus, to incorporate prior known relations among words into the training process.

1.3 Literature Review

The following section provides an overview of four previously explored word embedding methods. The section also presents previous research on Swedish word embedding.

1.3.1 Word2Vec

The Word2Vec framework was introduced in a series of papers written by Mikolov et al. (2013a; 2013b). The core of the framework is a neural network with a single hidden layer. Word sequences from a large text corpus are fed into the network and the objective is to learn the weights of the projection layer, which will be the obtained word embeddings (McCormick, 2016).

Word2Vec consists of two different models: the Skip-gram model and the Continuous Bag-of-Words (CBOW) model. For the Skip-gram model, the network is trained to predict a context word using the target word, while the task for the CBOW network is to predict a target word given context words. Although the tasks for the neural networks differ between the two models, the objective of both models is to train high-quality word vectors that capture both syntactic and semantic similarities based on a large text corpus.

The training corpus consists of a series of N words, $w_1, ..., w_N$. For each target word, a context window of neighboring words is considered so that the input sequence contains c words before and c words after the target word. The main task of the Skip-gram model is to maximize the log probability of a context word, (w_{i+j}), given one target word (w_i). This task equates to maximizing Equation 1, where N is the total number of words in the training corpus and c is the selected size of the context window (Mikolov et al., 2013b).

$$\log p(w) = \sum_{i=1}^{N} \sum_{\substack{-c \leq j \leq c \\ j \neq 0}} \log\, p(w_{i+j}|w_i) \tag{1}$$

To compute the probability of a context word given a center word, each unique word in the vocabulary $(v_1, ..., v_V)$ is represented by two sets of vectors. One embedding vector $\rho_v \in \mathbb{R}^D$ and one context vector $\alpha_v \in \mathbb{R}^D$, where D denotes the dimensionality of the vector. The probability of the context word (w_{i+j}), given the target word (w_i) is given by Equation 2 (Mikolov et al., 2013b). The numerator is the dot product of the context vector for a neighboring word and the embedding vector for the target word. The product in the numerator quantifies the similarities between the two words. The denominator is a normalizing constant that spans the entire vocabulary (V).

$$p(w_{i+j}|w_i) = \frac{\exp({\alpha_{i+j}}^\top \rho_i)}{\sum_{v=1}^{V} \exp({\alpha_v}^\top \rho_i)} \tag{2}$$

Equation 1 is maximized using a gradient descent algorithm that updates the vectors, ρ and α, to obtain closer vector representations for words that appear in the same context. However, since the vocabulary is large, the normalizing constant in Equation 2 becomes computationally costly. To make the process more efficient, Mikolov et al. (2013b) suggest approximating Equation 2 using negative sampling.

By incorporating negative sampling in the objective function, only a small part of the vectors are updated at each iteration. For every target and context word pair, K negative samples are drawn from a noise distribution. Mikolov et al. (2013b) recommend using the unigram distribution raised to the power of 0.75 for optimal results. It is assumed that the negative samples do not appear in the same context as the target word. Instead of updating vectors for the complete vocabulary at each iteration, only the vectors for the target word, the context word, and the K negative samples are updated to optimize the objective. The ideal number of negative samples depends on the size of the training corpus, but generally ranges between 2 and 20 (Mikolov et al., 2013b). The approximation simplifies the calculations significantly, and the objective function for the Skip-gram model with negative sampling is given by Equation 3, where $\sigma(\cdot)$ is the sigmoid function, $\sigma(x) = \frac{1}{1+e^{-x}}$.

$$L(\rho, \alpha) = \frac{1}{N} \sum_{i=1}^{N} \sum_{\substack{-c \le j \le c \\ j \ne 0}} \left[\log\, \sigma({\alpha_{i+j}}^\top \rho_i) + \sum_{k=1}^{K} \log\, \sigma(-{\alpha_k}^\top \rho_i) \right] \tag{3}$$

In contrast to the Skip-gram network's training task, the CBOW model's network is trained to predict a target word based on its neighboring words. This task corresponds to maximizing Equation 4, i.e., maximizing the log probability for a target word given surrounding words.

$$\log p(w) = \sum_{i=1}^{N} \sum_{\substack{-c \le j \le c \\ j \ne 0}} \log\, p(w_i | w_{i-c}, w_{i-c+1}, ..., w_{i+c-1}, w_{i+c}) \tag{4}$$

The probability of a target word given its context words is calculated according to Equation 5, where $\hat{\alpha}_i$ denotes the sum of the context vectors conditional on a target word as described in Equation 6.

$$p(w_i | w_{i-c}, w_{i-c+1}, ..., w_{i+c-1}, w_{i+c}) = \frac{\exp({\rho_i}^\top \hat{\alpha}_i)}{\sum_{v=1}^{V} \exp({\rho_v}^\top \hat{\alpha}_i)} \tag{5}$$

$$\hat{\alpha}_i = \sum_{\substack{-c \le j \le c \\ j \ne 0}} \alpha_{i+j} \tag{6}$$

Similar to the Skip-gram model, the denominator in Equation 5 becomes costly for large vocabularies. To facilitate the training process while retaining the quality of the embeddings, Equation 5 can be approximated using negative sampling (Mikolov et al., 2013b). The objective function for the CBOW model with negative sampling is stated in Equation 7. The first term of the equation maximizes the probability of co-occurrence of the target word and the words within the context window. The second term of the equation minimizes the probability of co-occurrence between the words within the context window and a random word from the negative sample.

$$L(\rho, \alpha) = \sum_{i=1}^{N} \left[\log \sigma({\rho_i}^\top \hat{\alpha}_i) + \sum_{k=1}^{K} \log \sigma(-{\rho_k}^\top \hat{\alpha}_i) \right] \tag{7}$$

To further facilitate the computational cost for both the Skip-gram and the CBOW model, Mikolov et al. (2013b) suggest subsampling frequent words. Subsampling acknowledges the imbalance between rare and frequent words in the training set and is applied during the preprocessing of the data. The process implies that some instances of the most frequent words in the corpus are removed. The probability of discarding word w_i is given by Equation 8, where t is the chosen threshold frequency and $f(w_i)$ is the frequency of word w_i.

$$p(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{8}$$

1.3.2 Dict2vec

To improve the context-based Word2Vec framework, Tissier et al. (2017) proposed the Dict2vec method. The Dict2vec algorithm allows for prior knowledge to be included using dictionary definitions. The authors argue that there are two distinct limitations when solely relying on context when training the word vectors. Firstly, it is probable that words appearing in the same context window are not always related. The second limitation is that words with the same meaning, e.g. synonyms, hardly ever appear in the same context window. By adding side information from dictionaries to the objective function, the intention is that the language representation can be refined and clear semantic relations can be captured (Tissier et al., 2017).

The foundation of Dict2vec relies on identifying weak and strong pairs from dictionary definitions. The identified strong and weak pairs are incorporated into the objective function alongside a large text corpus. For a word pair to be classified as strong, two words need to occur in each other's definition. The word pairing is weak if one word is found in the others definition, but not the other way around. Table 1 illustrates an example of a strong pair collected from the Cambridge English Dictionary.

Table 1: Example of a strong pair, where the words appear in each other's definitions.

Word	**Definition**
money	*coins* or notes, that are used to buy things
coin	a small, round piece of metal [...] that is used as *money*

Equation 9 describes the global objective for the Dict2vec algorithm, where ρ_{w_t} is the embeddings vector for the target word and α is the context vector for a strong/weak word. The global objective includes the objective function for a Skip-gram model with negative sampling (Equation 3) and the cost of the strong and weak pairs. The added cost consists of a sample from the strong pairs (V_s) and a sample from the weak pairs (V_w). To optimize the objective function, the cost for all targets is minimized, indicating that the vectors move closed in the vector space (Tissier et al., 2017). The impact of the positive sample on the global objective depends on the hyperparameters β_s and β_w. For $\beta_s = \beta_w = 0$, the model becomes the Skip-gram model with negative sampling.

$$L(\rho, \alpha) = L_{SGNS}(\rho, \alpha) + \beta_s \sum_{w_i \in V_s(w_t)} \log \sigma({\alpha_{w_i}}^\top \rho_{w_t}) + \beta_w \sum_{w_j \in V_w(w_t)} \log \sigma({\alpha_{w_j}}^\top \rho_{w_t}) \quad (9)$$

The central principle of minimizing the cost of strong and weak pairs is that words that form strong or weak pairs get more similar representations. The linked relation is applied between the embedding vector for the target word and the context vector for the identified word pairs. In line with Equation 9, the embedding and the context vector move closer in the vector space. Using the example word pair in Table 1, if the words *money* and *coin* form a strong pair, the distance between ρ_{money} and α_{coin} is constrained in the objective function.

1.3.3 Bernoulli Embeddings

Relating to the Word2Vec framework, Bernoulli embeddings was presented as part of the exponential family embeddings. Under certain assumptions, the Bernoulli embeddings algorithm is similar to the Skip-gram model and identical to the CBOW model with negative sampling (M. R. Rudolph, Ruiz, Mandt, & Blei, 2016). However, for Bernoulli embeddings, the vector representations are used as latent variables in a probabilistic model (M. Rudolph & Blei, 2017).

The Bernoulli model consists of two parts, the embeddings and the conditional distribution of every data point. Each word in the corpus corresponds to a one-hot vector, $x_i \in \{0,1\}^V$, where V is the vocabulary size. The vector contains one non-zero entry for position i of the word in the vocabulary (M. Rudolph & Blei, 2017). A context window (c_i) for each target word is considered and holds the indices for the neighboring data points. Equation 10 depicts the conditional probability of a target word (x_{iv}) given its context words (x_{c_i}). The Bernoulli probability (p_{iv}) is expressed as a linear combination of the embedding (ρ) and context (α) vectors, as specified in Equation 11. Similar to the Word2Vec notation, $\sigma(\cdot)$ denotes the sigmoid function.

$$x_{iv}|x_{c_i} \sim \text{Bernoulli}(p_{iv}) \tag{10}$$

$$p_{iv} = \sigma\left(\rho_v^\top \sum_{s \in c_i} \alpha_s\right) \tag{11}$$

The embedding vectors and context vectors are shared across the corpus. This property ensures that the embedding for a specific word is identical regardless of the position in the training corpus (M. Rudolph & Blei, 2017). A priori, the vectors are modeled using a Gaussian random walk, as stated in Equation 12. The hyperparameter, $\lambda \in \mathbb{R}_+$, is the regularization constant that determines the strength of the prior.

$$\rho_v, \alpha_v \sim \mathcal{N}(0, \lambda^{-1}\boldsymbol{I}) \tag{12}$$

The global objective function is presented in Equation 13. The first part of the equation can be split into positive and negative samples, similar to the Skip-gram and CBOW model. The second part of the equation includes the log priors for the embeddings and context vectors, which regulate the objective function. Since the priors follow a Gaussian random walk, the system is transformed into an ℓ_2 regularized CBOW model (Yrjänäinen & Magnusson, 2022).

$$L(\rho, \alpha) = \sum_{i=1}^{N} \sum_{v=1}^{V} \log p(x_{iv}|x_{c_i}) + \log p(\rho) + \log p(\alpha) \tag{13}$$

The ℓ_2 regularization term is commonly used for machine learning algorithms (Phaisangittisagul, 2016). Since most machine learning algorithms aim to train models that produce accurate results for unseen input data, a penalty term is added to the objective function to prevent the model from overfitting the training data. The ℓ_2 penalty is applied to word embedding methods to provide a more general vector representation. This property becomes particularly important for words that do not appear frequently in the training data.

1.3.4 Probabilistic Embeddings with Laplacian Graph Priors

The probabilistic approach from Bernoulli embeddings and the inclusion of prior knowledge from the Dict2vec algorithm can be combined using Probabilistic Embeddings with Laplacian Priors, PELP (Yrjänäinen & Magnusson, 2022). The flexibility of the Laplacian prior ensures that any graph-structured side information can be incorporated into the objective function of the Skip-gram model or the CBOW model.

An undirected graph, $G = (V, E)$, consists of a set of vertices and a set of edges (Merris, 1994). Each word in the dictionary is represented in a graph by a vertex (V), where an edge (E) indicates a predefined semantic relationship between two words (Faruqui et al., 2014). The Laplacian matrix ($\boldsymbol{L}$) for the graph is given by Equation 14, where $\boldsymbol{D}$ is the degree matrix and $\boldsymbol{A}$ is the adjacency matrix. Each entry in $\boldsymbol{L}$ is given by Equation 15, where $d(i)$ is the degree of the ith vertex and $i \sim j$ represents a link between two vertices (Yrjänäinen & Magnusson, 2022).

$$\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A} \tag{14}$$

$$l_{i,j} = \begin{cases} d(i), & \text{if i = j} \\ -1, & \text{if i} \sim \text{j} \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

By definition, the Laplacian matrix is a positive semi-definite matrix (Merris, 1994). To employ the Laplacian as a prior in the PELP model, the Laplacian needs to be strictly positive definite. This requirement is satisfied by summing the Laplacian with any positive diagonal matrix (Yrjänäinen & Magnusson, 2022) according to Equation 16, where $\lambda_1, \lambda_0 \in \mathbb{R}_+$. The augmented Laplacian ($\mathbf{L}_+$) is a valid precision matrix for the multivariate Gaussian distribution (Yrjänäinen & Magnusson, 2022).

$$\boldsymbol{L}_+ = \lambda_1 \boldsymbol{L} + \lambda_0 \boldsymbol{D} \tag{16}$$

Using the augmented Laplacian as a precision matrix, the joint distribution for the embeddings and the context vectors is set according to Equation 17 (Yrjänäinen & Magnusson, 2022). The dimensionality of the embeddings is denoted by D, and $\theta = [\rho, \alpha]$ is the combined parameter vector.

$$p(\rho, \alpha) = \prod_{i=1}^{D} \mathcal{N}(\theta_{:i}|0, \boldsymbol{L}_+^{-1}) \tag{17}$$

The logarithmic form of Equation 17 is presented in Equation 18, where C is a normalizing constant. For two linked words, w_a and w_b, the Laplacian prior constrains the squared difference between their respective parameters θ_a and θ_b (Yrjänäinen & Magnusson, 2022). The strength of the prior is determined by the hyperparameters, λ_0 and λ_1.

$$\log p(\rho, \alpha) = -\frac{\lambda_1}{2} \sum_{(a,b)\in E} ||\theta_a - \theta_b||^2 - \frac{\lambda_0}{2} \sum_{a\in V} \left(||\rho_a||^2 + ||\alpha_a||^2\right) + C \tag{18}$$

The Laplacian prior in Equation 18 is applied to the global objective function. During the learning process, the combined objective is optimized through maximum a posteriori estimation. Equation 19 depicts the full posterior, where $\log p(x|\theta)$ is the likelihood of either the Skip-gram algorithm in Equation 3, or the CBOW algorithm in Equation 7 (Yrjänäinen & Magnusson, 2022).

$$\log p(\theta|x) = \log p(x|\theta) + \log p(\rho, \alpha) \tag{19}$$

Equivalent to the Dict2vec approach, the structure of the Laplacian allows for connections between related words. Unlike Dict2vec model, the side information included in PELP must be symmetric. This restriction follows that one-sided directed connections, i.e., weak pairs, are not applicable in the PELP model. However, due to the flexibility of the Laplacian priors, the assigned connections can be applied both between embedding-context vectors, embedding-embedding vectors, and context-context vectors. Subject to the included side information, the graph can be adjusted to reflect the desired lexical relation.

Adding an edge between the embedding and the context vector implies that the two connected words are similar but not necessarily identical. This edge is suitable for words that are semantically related. An edge between two embedding vectors suggests that the two words ought to share the exact vector space. Identical vector representations are suitable for words that can be used interchangeably, i.e., synonyms. Depending on the size of the hyperparameters, the strength of the prior relative to the training corpus can be modified.

1.3.5 Swedish Word Embeddings

In 2016, an attempt for Swedish word vector representations, Swectors, was presented (Fallgren, Segeblad, & Kuhlmann, 2016). Three different word embedding methods were trained and evaluated: Skip-gram with negative sampling (SGNS), CBOW, and GloVe. The models were trained using the large corpus *Göteborgsposten* from the years 2001 to 2013, provided by Språkbanken. The corpus contains approximately 220M tokens and consists of sentences sampled from news articles. The models were trained to explore different settings on window size, the dimensions of the embeddings, and the required number of iterations.

After evaluating the results, Fallgren et al. (2016) suggest that the CBOW algorithm performs best on the evaluation tasks, with the SGNS as a close second. Comparing the different settings, the optimal window size for the models is between 8 and 10 and it is concluded that a larger window size is preferred. However, the proportional improvement of a larger window size stagnates. The results also show that the 300-dimensional space outperforms the 50-dimensional space for all methods. Moreover, it is proposed that the trained vectors from the CBOW model can be used for future studies on Swedish natural language processing (Fallgren et al., 2016). The trained vectors are published at Språkbanken.

1.4 Research Question

In light of the literature presented above, it appears that the potential benefits of including prior knowledge in the training process have not been explored for Swedish word embedding. This thesis aims to explore the following research questions:

- *What linguistic information from SAOL can be extracted to generate more efficient word embeddings?*
- *In what ways can the use of SAOL as a knowledge graph help make Swedish word embeddings more accurate?*

The rest of the thesis is structured as follows. Section 2 presents the text corpus and the lexical resource that is used for training the models. In Section 3 the used methods are described. The obtained results are presented and discussed in Section 4. Finally, Section 5 concludes the thesis.

2 Data

The following section presents the training corpus and the lexical resource that is used to generate Swedish word embeddings with a Laplacian graph prior.

2.1 Swedish Wikipedia Corpus

The aforementioned Swedish word embeddings (Fallgren et al., 2016) were trained on the Swedish corpus *Göteborgsposten*, which consists of sentences sampled from longer articles. Since the current implementation of the PELP model requires sufficiently large sections of coherent text, it is not possible to use a sentence-based corpus. With this limitation in mind, the word embeddings are trained on the Swedish Wikipedia dump (2022), which contains full articles collected from Wikipedia. Wikipedia articles cover an extensive variety of topics and are continuously maintained, which is essential to generate general word embeddings for natural language (Gabrilovich & Markovitch, 2014).

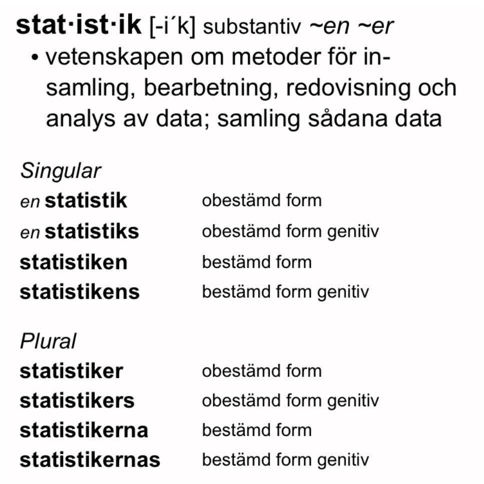
During the preprocessing phase, all non-alphabetical characters are removed and all numbers are spelled out. The full data contains over 500M words consisting of only lowercase letters (a-ö). To facilitate the training process, the full data is truncated to two smaller training sets. The first set contains the first 50M tokens and the second training set contains the first 200M tokens. A similar truncation method is used by Tissier et al. (2017) for the English Wikipedia dump.

In line with previously presented embedding methods, words with less than five entries are removed from the data (Mikolov et al., 2013b; Tissier et al., 2017). Moreover, following Mikolov et al. (2013a), some of the most frequent words are discarded to control for the imbalance between rare and frequent words in the data. Equation 20 shows the probability of which words are discarded from the data, where $f(w_i)$ is the frequency of word w_i and 10^{-5} is the chosen threshold.

$$p(w_i) = 1 - \sqrt{\frac{10^{-5}}{f(w_i)}} \tag{20}$$

2.2 SAOL

The most recent edition of SAOL, Svenska Akademiens ordlista (2015), is used to explore the effect of prior knowledge on a word embedding method. SAOL is a human language dictionary provided by the Swedish Academy and consists of approximately 126 000 words. It mainly offers a reference for spelling, inflectional endings, and pronunciation. Still, for most words, it also contains a short definition of the word. Figure 1 shows the dictionary entry for the word *statistik* (*statistics*) collected from SAOL (2015).



stat·ist·ik [-i´k] substantiv *~en ~er*
• vetenskapen om metoder för in-samling, bearbetning, redovisning och analys av data; samling sådana data

Singular	
en **statistik**	obestämd form
en **statistiks**	obestämd form genitiv
statistiken	bestämd form
statistikens	bestämd form genitiv
Plural	
statistiker	obestämd form
statistikers	obestämd form genitiv
statistikerna	bestämd form
statistikernas	bestämd form genitiv

Figure 1: An example of a dictionary entry from SAOL (2015).

3 Methods

To evaluate the impact of knowledge graphs on Swedish word embeddings, four different PELP models are trained and tested. For comparative purposes, a standard Word2Vec model and a Bernoulli model with a vague prior ($\lambda = 1$) are trained for baseline results. All explored models are based on the CBOW approach, i.e., the network task is to predict a target word given neighboring words. Furthermore, each model is trained on a 50M token training set and a 200M token training set to evaluate the impact of training data size.

For the baseline models, the Swedish Wikipedia corpus (2022) is used as the sole training data. To incorporate prior known lexical relations, side information from SAOL (2015) is included in the form of knowledge graphs for the four PELP models. The knowledge graphs contain different semantic relations to investigate what linguistic information from SAOL is useful to generate more efficient word embeddings.

3.1 Knowledge Graphs

The knowledge graphs are created using the 14th edition of SAOL (2015). Each entry in SAOL is connected to an HTML-file containing information regarding word morphology, word formation, and, for some words, a definition. As previously mentioned, for this thesis, four knowledge graphs are examined.

The first knowledge graph includes strong pairs as defined by Tissier et al. (2017). A word pair is classified as strong if each word appears in the other's definition. For words with multiple meanings, all definitions are considered. Similar to the Dict2vec objective function, the edge is applied between the embedding and the context vector. Using the example from Table 1 (see Section 1.3.2), $money$ and $coin$ appear in each other's definitions, which implies that edges are applied between $\rho_{\text{money}} - \alpha_{\text{coin}}$ and $\rho_{\text{coin}} - \alpha_{\text{money}}$. The PELP model with the knowledge graph containing strong pairs from word definitions is referred to as *PELP (definition)*. A large portion of the identified strong pairs in the graph are synonyms or near-synonyms. The graph also links hyponyms and hypernyms, such as *regn-nederbörd* (*rain-precipitation*), and word pairs with opposite meanings, such as *våt-torr* (*wet-dry*).

The second knowledge graph consists of connecting compound words with their respective base words. A compound word is two or more words combined to yield a new meaning. Compounding is frequently used in the Swedish language, and a significant portion of the words in SAOL are closed compound words. SAOL separates compound words into their constituent parts. To incorporate compound words into the knowledge graph, all compound words are connected to their component words. The edge is applied between the embedding for the compound word and the context vector for the component words. For example, the word *kaffekopp* (*coffee cup*) will have an edge between $\rho_{\text{kaffekopp}} - \alpha_{\text{kaffe}}$, and between $\rho_{\text{kaffekopp}} - \alpha_{\text{kopp}}$.

The model with the included knowledge graph consisting of compound words is referred to as *PELP (compound)*.

The first and second graphs are combined to form the third knowledge graph, *PELP (def+comp)*. Similar to the implementation of the original graphs, the edges are applied between the embeddings and the context vectors.

Finally, a knowledge graph is created for all the inflectional forms of a word. Inflectional endings are added to the base form of a word to indicate a change in number or tense, among other things. In Swedish, definite articles are also generally added as an inflectional suffix to the base word. Using the example dictionary entry in Figure 1 as a reference, all forms of the word *statistik* are connected in the graph. The edge is added between the embeddings, e.g. $\rho_{\text{statistik}} - \rho_{\text{statistiks}}$, $\rho_{\text{statistik}} - \rho_{\text{statistiken}}$, etc. As discussed in Section 1.3.4, an edge between two embeddings indicates words that can be used interchangeably. Although inflectional forms of a word cannot be used as synonyms, they are highly semantically similar. In addition, it is not likely that multiple inflectional forms of the same word will appear within the same context window. The knowledge graph with inflectional forms of a word is referred to as *PELP (inflection)*.

The number of vertices and edges for all created graphs is presented in Table 2. As presented in the table, the identifies lexical relations differs between the smaller and larger test set. The reason for the difference is that the vocabulary is larger for the 200M token training set, which implies that more lexical relations can be identified.

Table 2: Size of the graphs obtained from SAOL.

Model	**Vertices 50M**	**Edges 50M**	**Vertices 200M**	**Edges 200M**
PELP (definition)	3 010	1 738	3 410	1 964
PELP (compound)	37 681	54 358	57 468	87 153
PELP (def+comp)	39 571	56 086	59 565	89 107
PELP (inflection)	93 310	196 044	151 103	313 979

3.2 Evaluation Method

The trained word embeddings are examined using two different evaluation methods: word similarity and QVEC-CCA. Firstly, in line with the evaluation of the Dict2vec model (Tissier et al., 2017) and the PELP model (Yrjänäinen & Magnusson, 2022), the vectors are evaluated with a similarity task. The similarity task consists of measuring the correlation between human perceived semantic similarity and the distance between word vectors (B. Wang et al., 2019). The distance between the word embeddings is estimated with cosine similarity. The cosine similarity is calculated according to Equation 21, where A and B are two word vectors and the denominator is the product of the ℓ_2 norm of the vectors.

$$similarity(A, B) = \frac{A \cdot B}{||A||\ ||B||} \tag{21}$$

The cosine similarity score is compared with human perceived semantic similarity using Spearman's rank correlation. The obtained correlation coefficient takes an absolute value between 0 and 1, where 1 indicates an embedding close to human judgment (Tissier et al., 2017). For English word embeddings, numerous test sets with human-annotated similarity ratings are available, such as SimLex-999 (Hill, Reichart, & Korhonen, 2014) and WordSim353 (Finkelstein et al., 2001). For Swedish word embeddings, the resources are more limited. Currently, Språkbanken provides a series of evaluation tasks for Swedish natural language understanding models called SuperLim (Adesam, Berdicevskis, & Morger, 2020). The SuperLim collection aims to translate the SuperGLUE benchmark (A. Wang et al., 2019), which was previously only applicable to English data. One of the evaluation tasks included in the collection is SuperSim. The SuperSim test set consists of more than 1 300 word pairs independently judged by five annotators on similarity and relatedness, and is created to resemble the SimLex-999 and WordSim353 test sets.

The SuperSim test set is used to evaluate the extent of correlation between the annotated human judgment and the cosine similarity of the trained word embeddings. Similar to the preprocessing of the training data, all words in the test set are converted to lowercase letters and non-alphabetical characters are removed. Entries in the test set with more than one word are omitted since the trained word embeddings consist of one vector per word. Word pairs in the test set that are not in the vocabulary for the embeddings are skipped during evaluation. Given that the embeddings are generated from the same training set and therefore share the vocabulary, the skipped word pairs will not affect the results. However, all models are trained on two different training set sizes, which implies that the vocabularies differ between the 50M token set and the 200M token set.

The SuperSim test set is split into a validation set and an evaluation set. The validation set contains 50 % of the complete test set and is used to find optimal values of the hyperparameters in the PELP model. The evaluation set is used to acquire the Spearman's rank correlation coefficients for the baseline models and the tuned PELP models. Since Spearman's rank correlation is a non-parametric statistic and the shape of the sampling distribution is unknown, the confidence intervals for the correlation coefficients are constructed using bootstrap (Ruscio, 2008). For each model, 10 000 bootstrap resamples are estimated. A 95 % confidence interval of the resampled correlations is computed using the bias-corrected and accelerated (BCA) bootstrap method. The used bootstrap method adjust for asymmetric sampling distribution (Ruscio, 2008).

The second used evaluation task is the QVEC-CCA method. The QVEC-CCA method was used to evaluate the aforementioned Swedish word embeddings created by Fallgren et al. (2016). QVEC is an intrinsic evaluation method, which is used as a quality measure for word embeddings (Tsvetkov, Faruqui, Ling, Lample, & Dyer, 2015). The evaluator estimates the correlation between the trained embeddings and a manually crafted set of linguistic vectors.

The linguistic resources for the Swedish QVEC test set are SALDO, the Swedish Association Lexicon (Borin et al., 2013), and the Stockholm-Umeå corpus (SUC). SALDO is used to categorize words into primary descriptors, also referred to as supersenses (Fallgren et al., 2016). All words are directly or indirectly categorized by a primary descriptor. SUC is annotated with primary descriptors from SALDO and is used to obtain sense frequencies for all words. The full test set contains approximately 13 000 words divided into 41 primary descriptors. Similar to the implementation of the similarity task, words in the QVEC test set that are not in the embedding vocabulary are omitted.

The original QVEC method does not support a comparison between embeddings with different dimensions. Since the generated score is not normalized, it follows that vectors with higher dimensions produce a higher score (Tsvetkov, Faruqui, & Dyer, 2016). This issue is solved with the QVEC-CCA, where canonical correlation analysis is applied to measure the correlation between the embeddings and the crafted linguistic vectors.

3.3 Training Settings

There are several hyperparameters to be determined before training the models. To ensure a fair comparison between the methods, the common hyperparameters are shared for the different models. The shared hyperparameters include the number of dimensions of the embeddings, the size of the context window, and the number of negative samples.

Firstly, the dimension of the embeddings usually varies from 50 to a few hundred (Goldberg, 2017). A larger vector dimension might capture more aspects of a word. However, the size of the embeddings affects the processing time and the memory requirements. Fallgren et al. (2016) compare two different vector dimensions, 50 and 300, and suggest that the larger-sized vectors improve the accuracy of the embeddings. When exploring different lengths of the vectors, Tissier et al. (2017) argue that increasing the dimensionality of the vectors increases the performance. However, the increase in performance stagnates at a dimension of around 100. To ease the computational cost that larger dimensional vectors entail, the dimension of the embeddings is set to 100. The chosen dimensionality is frequently used in previous research (Yrjänäinen & Magnusson, 2022; Tissier et al., 2017; M. Rudolph & Blei, 2017).

The size of the context window determines the number of neighboring words to include for each target word. A larger context window generally captures more topical relations and word relatedness, whereas a smaller context window usually reflects functional and syntactic similarities (Lison & Kutuzov, 2017; Levy & Goldberg, 2014). In line with window sizes settings commonly used in the literature (Tissier et al., 2017), the size of the context window for the trained models is set to five.

To approximate the objective of the models, negative sampling is applied during the learning process. As previously stated, Mikolov et al. (2013b) suggest using $2 - 20$ negative samples for each target word. For large training sets, the objective function is well approximated with a smaller number of negative samples. In this experiment, five negative samples are considered for each target word. Additionally, the number of epochs is set at 15, similar to the estimation of the original PELP model (Yrjänäinen & Magnusson, 2022). The number of epochs determines the number of times the entire training set is passed through the model.

The PELP models include the two hyperparameters λ_0 and λ_1 that need to be adjusted to find the optimal values for the given purpose. The hyperparameters determine the strength of the prior, i.e., the attention on the knowledge graph versus the text corpus. In the original PELP paper (Yrjänäinen & Magnusson, 2022), the size of the prior is selected using a hyperparameter search over $\lambda_0 \in \{1, 2.5, 5\}, \lambda_1 \in \{3.5, 7\}$. The same sets of values are used for this thesis to determine the strength of the prior for each model.

To avoid overfitting the evaluation set, the aforementioned similarity validation set is used to find optimal values of λ_0 and λ_1. The hyperparameters are chosen based on the average of the similarity and the relatedness rank correlation coefficients. A similar procedure is conducted for all PELP models and the identifies optimal hyperparameters are used for both the 50M token training set and the 200M token training set. The identified optimal values of the hyperparameters are presented in Table 3.

Table 3: Values of the hyperparameters for the PELP models.

Model	λ_0	λ_1
PELP (definition)	5.0	7.0
PELP (compound)	5.0	3.5
PELP (def+comp)	5.0	3.5
PELP (inflection)	1.0	3.5

4 Results and Discussion

In this section, the results from the six explored word embedding models are presented and discussed. Four PLEP models with different knowledge graphs are trained to evaluate what linguistic information from SAOL is useful for generating more efficient word embeddings. In addition, a Word2Vec CBOW model and a Bernoulli model with a vague prior are trained for baseline results. Using a word similarity task and the QVEC-CCA approach, the word embedding models are evaluated to determine how the incorporated knowledge graph produces more accurate Swedish word embeddings.

4.1 Similarity and Relatedness

Table 4 shows the Spearman's rank correlation between the human-judged similarity score and the calculated cosine similarity of the embeddings. The provided confidence intervals are constructed using a bias-corrected and accelerated bootstrap method.

Table 4: Spearman's rank correlation, with a corresponding 95% confidence interval, between the vector's cosine similarity and a human-judged similarity score.

Model	**50M**	**200M**
Word2Vec	0.458	0.494
	(0.413, 0.499)	*(0.450, 0.533)*
Bernoulli	0.470	0.493
	(0.424, 0.511)	*(0.450, 0.533)*
PELP (definition)	**0.478**	0.494
	(0.435, 0.519)	*(0.450, 0.534)*
PELP (compound)	0.472	**0.509**
	(0.426, 0.511)	*(0.467, 0.548)*
PELP (def+comp)	**0.478**	0.495
	(0.434, 0.520)	*(0.451, 0.537)*
PELP (inflection)	0.470	0.499
	(0.422, 0.508)	*(0.456, 0.538)*

By comparing the correlation coefficients for the two different training sets in Table 4, it is clear that the performance on the similarity task increases for all models trained on the larger training set. Since the confidence intervals overlap, the suggested difference is not statistically significant. Nevertheless, this result is consistent with the general intuition that machine learning algorithms typically achieve better results with large amounts of data. As previously discussed, one reason for including prior knowledge is that less data is required in the training process (von Rueden et al., 2021). However, the knowledge graphs used in this experiment could not compensate for the smaller training set since the performance of the models are consistently better when trained on the larger training set.

Almost all of the tested PELP models in Table 4 outperform the Word2Vec model and the Bernoulli model, which were used for baseline comparison. The same results hold for both the 50M and 200M training sets. Although the difference is not significant, these results imply that the knowledge graphs incorporated into the PELP models are informative and enable the embeddings to capture semantic information beyond the large text corpus. The one exception is the *PELP (inflection)* model, where the difference in correlation compared to the baseline models is marginal. One potential explanation for this is that the included lexical relations in the *PELP (inflection)* graph do not strictly capture semantic information. Since the similarity task explicitly examines semantic similarity, in all probability, any possible captured lexical relation in the embeddings is not evaluated with the used test.

The Spearman's rank correlation between the human-judged relatedness score and the calculated similarity of the embeddings is presented in Table 5. Similar to Table 4, the provided confidence intervals are constructed using a bias-corrected and accelerated bootstrap method.

Table 5: Spearman's rank correlation, with a corresponding 95% confidence interval, between the vector's cosine similarity and a human-judged relatedness score.

Model	**50M**	**200M**
Word2Vec	0.538	0.574
	(0.498, 0.578)	*(0.533, 0.611)*
Bernoulli	0.543	0.579
	(0.503, 0.582)	*(0.539, 0.615)*
PELP (definition)	**0.559**	**0.591**
	(0.519, 0.597)	*(0.552, 0.627)*
PELP (compound)	0.552	0.577
	(0.512, 0.590)	*(0.536, 0.614)*
PELP (def+comp)	0.556	0.584
	(0.516, 0.593)	*(0.544, 0.621)*
PELP (inflection)	0.544	0.579
	(0.504, 0.584)	*(0.539, 0.616)*

The tested model with the best overall performance in Table 5 is the PELP model with the knowledge graph containing strong word pairs from word definitions. Similar to the similarity score in Table 4, the confidence intervals overlap, which implies that the difference in performance is not significant. Still, since the graph contains semantic relations, this result confirms the previously mentioned statement that prior known semantic relations can be used to specialize word embeddings for specific NLP tasks (Kiela et al., 2015). Another explanation for the improved performance for the *PELP (definition)* model is the possibility of adding an edge between two words that are included in the test set. If the distance between a word pair is constrained by the graph prior, and the same word pair is included in the test set with a high human-annotated relatedness score, the correlation coefficient might be affected in favor of the model with the included graph. However, since the test set contains both words that are highly semantically related and words that are completely unrelated, the effect on the presented correlation coefficient is probably negligible.

The results in Table 5 are interesting since the *PELP (definition)* graph contains the least number of edges of all the tested graphs (see Table 2). This property suggests that, although the provided information in SAOL is limited in terms of the number of definitions, the linked strong pairs seem to provide some effective information. For English word embeddings, several lexical resources are often combined to collect additional semantic information for most words in the vocabulary (Tissier et al., 2017; Yrjänäinen & Magnusson, 2022). A similar procedure could be explored for the Swedish word embeddings. Probably, the results from this experiment could see improvement with word definitions from more than one dictionary.

Comparing Table 4 and Table 5, all models receive a higher correlation coefficient on the relatedness test compared to the similarity test. This result implies that, generally, the embeddings are better at capturing word relatedness rather than semantic similarity. A possible explanation for this is that the used knowledge graphs do not distinguish between related words and similar words. For example, the *PELP (definition)* graph contains both words that are semantically similar, such as synonyms, and words that can be considered to be related, such as opposite words (*wet, dry*). Word similarity and relatedness scores are subjective and difficult to generally quantify, which affects the effectiveness of the similarity task (Faruqui, Tsvetkov, Rastogi, & Dyer, 2016). However, similar words are often considered related, but relatedness among words does not necessarily imply semantic similarity. While adding an edge between opposite words could potentially increase the relatedness score, it would most likely affect the word similarity score negatively. Depending on the purpose of the trained embeddings, it can be necessary to make more distinctions in the knowledge graph between relatedness and similarity.

Although the differences between some of the PELP models and the baseline models are marginal, the results suggest that the obtained distance between the embeddings is stable. Since the vectors are initially randomized and the objective function is approximated using negative sampling, the embeddings will not yield the same representations when running the model multiple times. Nonetheless, for reliable results, the spacing between the embeddings should be constant (B. Wang et al., 2019), which appears to be the case for this experiment.

Another aspect worth mentioning is the optimization of the hyperparameters. The hyperparameters are chosen in reference to the model performance on the validation set. The obtained correlation coefficient is compared with the correlation coefficient of the baseline models to get an indication of the value of the information in the prior. If the performance of the tested PELP model is inferior to the baseline results, the model is placing too much attention on the prior, indicating that lower values of the hyperparameters are desired. The ultimate extension of this approach would be to set both hyperparameters to zero. By setting the hyperparameters to zero, the effect of the prior vanishes and the system becomes a standard Word2Vec CBOW model. This interrelation between the model performance and the adjustment of the prior follows that, for this experiment setup, the performance can only be improved with the PELP model.

One limitation of the similarity task is that, currently, the only available evaluation data for Swedish embeddings is the SuperSim test set (Adesam et al., 2020). The complete SuperSim test set contains approximately 1 300 word pairs. The limited access to the desired data implies that only a fraction of the trained embeddings is evaluated with the SuperSim test set. For English semantic similarity tasks, there are several evaluation sets available. Commonly, multiple test sets are combined (Yrjänäinen & Magnusson, 2022; Tissier et al., 2017) to gain a broader perspective on the semantic similarities captured by the embeddings.

4.2 QVEC-CCA

Table 6 presents the results from the QVEC-CCA evaluation metric.

Table 6: Impact of different knowledge graphs on QVEC-CCA score.

Model	**50M**	**200M**
Word2Vec	0.225	0.242
Bernoulli	0.227	0.242
PELP (definition)	0.227	0.244
PELP (compound)	0.228	0.245
PELP (def+comp)	0.228	0.243
PELP (inflection)	**0.239**	**0.253**

In line with the similarity evaluator, all models receive a higher QVEC-CCA score when trained on the 200M token training set compared to the 50M training set. Comparing the performance of the different models, the only PELP model that distinctly differs from the two baseline models is the *PELP (inflection)*. One possible explanation for the increased performance is the structure of the used test set, SALDO. Since SALDO contains several inflectional forms of the same word, it is probable that those words are classified under the same primary descriptor in the test set. For the *PELP (inflection)* model, the distance between inflectional words is constrained. This constraint means that the vector representations for those words will be forced to be more similar. This property would likely benefit the QVEC-CCA performance for the *PELP (inflection)* model.

In addition, it is worth noting that the hyperparameters are adjusted in reference to the similarity task. To obtain the best possible results for the QVEC-CCA metric, the hyperparameters need to be optimized against the QVEC-CCA score. Since the difference between the tested models is relatively small in Table 6, it is not likely that fine-tuning the hyperparameters would considerably change the results.

In contrast to the mentioned limitations of the similarity task, the number of constructed linguistic word vectors for the Swedish evaluation set by Fallgren et al. (2016) and the English evaluation set by Tsvetkov et al. (2016) is comparable. However, one uncertainty with the implemented Swedish QVEC-CCA test set is the lack of knowledge regarding its correlation with downstream tasks. The English evaluation set is reported to correlate well with several downstream tasks (Tsvetkov et al., 2016), but similar experiments have not been constructed for the Swedish evaluation set.

5 Conclusions

The purpose of this thesis has been to explore if Swedish word embeddings can benefit from prior lexical knowledge. Four knowledge graphs, containing different linguistic relations collected from Svenska Akademiens ordlista (SAOL), were created to be used as side information. The knowledge graphs were incorporated into the objective function during the training process using the PELP model. In addition to the four PELP models, a standard Word2Vec model and a Bernoulli model with a vague prior were estimated for baseline results. All models were trained on a 50M token set and a 200M token set collected from the Swedish Wikipedia dump to evaluate the impact of the size of the training data. The obtained embeddings were evaluated with a word similarity task and the QVEC-CCA metric.

The results suggest that the performance increases for all models trained on the larger training set. The tested model with the best overall performance on the similarity task is the PELP model with the knowledge graph connecting strong pairs from word definitions. However, the difference between the evaluated models is not statistically significant. The results from the similarity task also suggest that the obtained embeddings capture word relatedness rather than word similarity. One explanation for these results is that the created graphs do not distinguish between word similarity and word relatedness, which might affect the similarity score negatively.

Additionally, the results from the QVEC-CCA evaluation method showed that the highest-scoring method is the PELP model with the graph connecting all the inflectional forms of a word. One discussed explanation for these results is the construction of the evaluation set, SALDO. Since different inflectional forms of the same word likely share primary descriptor in the evaluation set and the best-performing model limits the distance between inflectional words, the QVEC-CCA metric probably takes this relationship into account.

Acknowledgements

First and foremost, I would like to thank my supervisors Måns Magnusson and Väinö Yrjänäinen for their constructive advice and support through all stages of this project. I would also like to give special thanks to my classmates, Douglas Ekstedt and Jesper Mortensen Blomquist, for their helpful comments and for keeping me company in the library.

References

Adesam, Y., Berdicevskis, A., & Morger, F. (2020). *Swedishglue – towards a Swedish test set for evaluating natural language understanding models* (Tech. Rep.).

Baker, C. F., Fillmore, C. J., & Lowe, J. B. (1998, August). The Berkeley FrameNet project. In *36th annual meeting of the association for computational linguistics and 17th international conference on computational linguistics, volume 1* (pp. 86–90). Montreal, Quebec, Canada: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P98-1013` doi: 10.3115/980845.980860

Borin, L., Forsberg, M., & Lönngren, L. (2013). SALDO: a touch of yin to wordnet's yang. *Language resources and evaluation*, *47*(4), 1191–1211.

Chollet, F., & Allaire, J. J. (2018). Deep learning with R. Manning Publications Co.

Diligenti, M., Roychowdhury, S., & Gori, M. (2017). Integrating prior knowledge into deep learning. In *2017 16th ieee international conference on machine learning and applications (icmla)* (p. 920-923). doi: 10.1109/ICMLA.2017.00-37

Fallgren, P., Segeblad, J., & Kuhlmann, M. (2016). Towards a standard dataset of Swedish word vectors. In *Proceedings of the sixth swedish language technology conference (SLTC).* Umeå, Sweden.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., & Smith, N. A. (2014). Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Faruqui, M., Tsvetkov, Y., Rastogi, P., & Dyer, C. (2016, August). Problems with evaluation of word embeddings using word similarity tasks. In *Proceedings of the 1st workshop on evaluating vector-space representations for NLP* (pp. 30–35). Berlin, Germany: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W16-2506` doi: 10.18653/v1/W16-2506

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppin, E. (2001). Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on world wide web* (pp. 406–414).

Firth, J. R. (1957). A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis*.

Gabrilovich, E., & Markovitch, S. (2014). Wikipedia-based semantic interpretation for natural language processing. *CoRR*, *abs/1401.5697*. Retrieved from `http://arxiv.org/abs/1401.5697`

Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013, June). PPDB: The paraphrase database. In *Proceedings of the 2013 conference of the north American chapter of the association for computational linguistics: Human language technologies* (pp. 758–764). Atlanta, Georgia: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/N13-1092`

Goldberg, Y. (2017). Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, *10*(1), 1–309.

Harris, Z. S. (1954). Distributional structure. *Word*, *10*(2-3), 146–162.

Hill, F., Reichart, R., & Korhonen, A. (2014). *Simlex-999: Evaluating semantic models with (genuine) similarity estimation.* arXiv. Retrieved from `https://arxiv.org/abs/1408.3456` doi: 10.48550/ARXIV.1408.3456

Jonschkowski, R., Höfer, S., & Brock, O. (2015). *Patterns for learning with side information.* arXiv. Retrieved from `https://arxiv.org/abs/1511.06429` doi: 10.48550/ARXIV.1511.06429

Kiela, D., Hill, F., & Clark, S. (2015, September). Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 conference on empirical methods in natural language processing* (pp. 2044–2048). Lisbon, Portugal: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/D15-1242` doi: 10.18653/v1/D15-1242

Levy, O., & Goldberg, Y. (2014, June). Dependency-based word embeddings. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)* (pp. 302–308). Baltimore, Maryland: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/P14-2050` doi: 10.3115/v1/P14-2050

Levy, O., Goldberg, Y., & Dagan, I. (2015). Improving distributional similarity with lessons learned from word embeddings. *Transactions of the association for computational linguistics*, *3*, 211–225.

Lison, P., & Kutuzov, A. (2017, May). Redefining context windows for word embedding models: An experimental study. In *Proceedings of the 21st nordic conference on computational linguistics* (pp. 284–288). Gothenburg, Sweden: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W17-0239`

McCormick, C. (2016). Word2vec tutorial-the skip-gram model. *Apr-2016.[Online]. Available: http://mccormickml. com/2016/04/19/word2vec-tutorial-the-skip-gram-model.*

Merris, R. (1994). Laplacian matrices of graphs: a survey. *Linear Algebra and its Applications*, 143-176.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). *Efficient estimation of word representations in vector space.*

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Miller, G. A. (1995, nov). Wordnet: A lexical database for english. *Commun. ACM*, *38*(11), 39–41. Retrieved from `https://doi.org/10.1145/219717.219748` doi: 10.1145/219717.219748

Phaisangittisagul, E. (2016). An analysis of the regularization between l2 and dropout in single hidden layer neural network. In *2016 7th international conference on intelligent systems, modelling and simulation (isms)* (p. 174-179). doi: 10.1109/ISMS.2016.14

Rudolph, M., & Blei, D. (2017). *Dynamic bernoulli embeddings for language evolution.*

Rudolph, M. R., Ruiz, F. J. R., Mandt, S., & Blei, D. M. (2016). *Exponential family embeddings.*

Ruscio, J. (2008). Constructing confidence intervals for spearman's rank correlation with ordinal data: a simulation study comparing analytic and bootstrap methods. *Journal of Modern Applied Statistical Methods*, *7*(2), 7.

Sahlgren, M. (2008). The distributional hypothesis. *Italian Journal of Disability Studies*, *20*, 33–53.

Svenska Akademien. (2015). *Svenska Akademiens ordlista (SAOL)* (14th ed.). Stockholm, Sweden: Nordstedts förlag.

Tissier, J., Gravier, C., & Habrard, A. (2017). Dict2vec : Learning word embeddings using lexical dictionaries. In *Emnlp.*

Tsvetkov, Y., Faruqui, M., & Dyer, C. (2016, August). Correlation-based intrinsic evaluation of word vector representations. In *Proceedings of the 1st workshop on evaluating vector-space representations for NLP* (pp. 111–115). Berlin, Germany: Association for Computational Linguistics. Retrieved from `https://aclanthology.org/W16-2520` doi: 10.18653/v1/W16-2520

Tsvetkov, Y., Faruqui, M., Ling, W., Lample, G., & Dyer, C. (2015, September). Evaluation of word vector representations by subspace alignment. In *Proceedings of the 2015* *conference on empirical methods in natural language processing* (pp. 2049–2054). Lisbon, Portugal: Association for Computational Linguistics. Retrieved from https://aclanthology.org/D15-1243 doi: 10.18653/v1/D15-1243

von Rueden, L., Mayer, S., Beckh, K., Georgiev, B., Giesselbach, S., Heese, R., ... Schuecker, J. (2021). Informed machine learning - a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 1–1. Retrieved from https://doi.org/10.1109%2Ftkde.2021.3079836 doi: 10.1109/tkde.2021.3079836

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., ... Bowman, S. R. (2019). *Superglue: A stickier benchmark for general-purpose language understanding systems.* arXiv. Retrieved from https://arxiv.org/abs/1905.00537 doi: 10.48550/ARXIV.1905.00537

Wang, B., Wang, A., Chen, F., Wang, Y., & Kuo, C.-C. J. (2019). Evaluating word embedding models: methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, *8*(1). Retrieved from https://doi.org/10.1017%2Fatsip.2019.12 doi: 10.1017/atsip.2019.12

Wikipedia contributors. (2022). *Wikimedia incremental dump files for the Swedish wikipedia on 20220204.* Retrieved from https://archive.org/details/incr-svwiki-20220204

Yrjänäinen, V., & Magnusson, M. (2022). *Probabilistic embeddings with laplacian graph priors.* arXiv. Retrieved from https://arxiv.org/abs/2204.01846 doi: 10.48550/ARXIV.2204.01846

Yu, M., & Dredze, M. (2014, June). Improving lexical embeddings with semantic knowledge. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)* (pp. 545–550). Baltimore, Maryland: Association for Computational Linguistics. Retrieved from https://aclanthology.org/P14-2089 doi: 10.3115/v1/P14-2089