Creating semantic representations

Finn Årup Nielsen and Lars Kai Hansen

Cognitive Systems, DTU Compute, Technical University of Denmark

Abstract. In this chapter, we present the vector space model and a number of ways to further process such a represention: With feature hashing, random indexing, latent semantic analysis, non-negative matrix factorization, explicit semantic analysis and word embedding, a word or a text may be associated with a distributed semantic representation. Deep learning, explicit semantic networks and auxiliary non-linguistic information provide further means for creating distributed representations from linguistic data. We point to a few of the methods and datasets used to evaluate the many different algorithms that create semantic representation, and we also point to some of the problems associated with distributed representations.

- Individual words can be represented in a vector representation, but the space spanned by the words does not have any semantic interpretation.
- Both feature hashing and random indexing can reduce the size of the vector space.
- When latent semantic analysis and non-negative matrix factorization are applied on bag-of-words matrices, they created distributed semantic representations where the dimensions may have interpretations as topics for the corpus
- Word embedding are distributed semantic representations of words efficiently created from context windows.
- Several ways exist for evaluating semantic representations, e.g., by word similarities, word analogies and word intrusion tasks.

1 Vector space model

In the vector space model, a word or a text is represented in a numerical vector and each element is associated with a term that might be a word, a phrase and/or a character or word n-gram. If we have multiple instances of texts, the numerical row vectors of each individual text may be stacked, forming a numerical matrix. If the texts are documents, we can refer to it as a *document-term matrix* or *bag-of-words matrix*

This normal vector space alone does not represent individual words in a distributed way, and a such, any distance measured between words will be the same, so that we cannot use this representation to indicate semantic distance for individual words.

The vector can be expanded by pairs of (consecutive) words, i.e., bigrams, or longer sequences, trigrams and generally n-grams, but the distances between individual terms will still be equal. However, when the vector space model is used to represent a text with multiple words — a paragraph or a complete document — the text will be represented over multiple elements in the vector and then distances between two texts may have relation to semantics. For a text, vector elements may reflect the presence or the counts of each word in the text.

A number of methods have been suggested to improve the vector space representation of a document: Elements associated with so-called stop words may be exluded from the vector. Stop words are common words, such as *the*, *and* and *its*, that seldomly carry much information about the topic of the document and may hinder more than help in any further semantic processing or interpretation with the vector representation. If a term only occurs in a single document, then it often may be discarded because it does not affect any modeling based on co-occurence.

Apart from n-grams, there are various other methods to modify the terms: the individual word may be stemmed or a lemmatizer can identify it lemma, multiword expressions may be detected as named entities and aggregated into one term.

The document-term matrix can be scaled both row-wise and column-wise. Long documents have many words and its associated vector with raw word counts is long, i.e., its norm is large, and the vector representation of a short and a long document about the same topic may lie far from each other in the vector space if no scaling for length is performed. Furthermore, words that occur in most documents across the corpus may not be the words that separated topics well, and such words should have less weight. One common scaling scheme is referred to as *tf-idf* — term frequency (times) inverse document frequency. The specific form of the scaling varies, e.g., the scikit-learn machine learning package uses[1]

$$\mathrm{tfidf}(d,t) = \mathrm{tf}(d,t) \times \mathrm{idf}(d,t) \tag{1}$$

where d and t are the document row and term column index, respectively. $\mathrm{tf}(d,t)$ is the raw word counts, and the inverse document frequency is computed as $\mathrm{idf}(d,t) = \log[(1+n)/(1+\mathrm{df}(d,t))]+1$ with n as the number of documents and $\mathrm{df}(t)$ the number of documents with the term t.

2 Feature hashing

The representation of words and phrases, unigram, bigram or higher-order n-grams produces very large feature spaces which poses a challenge for resource constrained systems. The so-called "hashing trick" can limit the number of features by setting up a limited number of buckets and assign each word to a bucket. The method goes under the name *random feature mixing* or *feature*

[1] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfTransformer.html

hashing [12]. The approach leads to hash collisions where multiple words share the same bucket. From the further modeling point of view, it will look like a massive homograph problem and it comes as no surprise that the performance of a model using feature hashing may degrade. However, researchers have made the perhaps surprising observation that the performance does not degrade that much. Ganchev and Dredze found that on a range of supervised text classification tasks across different labeled corpora with four popular machine learning methods and binary unigram features a reduction in size of 10 would still yield a performance of between 96,7% and 97,4% of the original model [12].

As the hashing function Ganchev and Dredze suggested Java's hashCode function followed by a modulo operation with a division using the size of the intended number of buckets [12], while Gensim implements hashing via a adler32 function and a modulo operation.[2]

With a good hash function we han hope that the words are distributed with an equal probability among the buckets. The rate of hash collision has then the same probability as the birthday paradox:

$$P(c) = 1 - \prod_{x=B-N}^{B-1} x/B \tag{2}$$

Feature hashing is not a distributed representation of words. Each word is still represented—as in the normal vector space model—with one single element in the vector, and all distances between words are the same (except for the words that collide, bringing their distance to zero).

3 Random indexing

In random indexing, texts are projected using a random matrix [4]. The advantage with this approach is that no model parameters need to be estimated, and only one pass over the corpus is necessary to project the text into a low-dimensional space. If individual words are projected they will get a distributed representation. However, the distances between the projected words do not gain any semantic interpretation. Nevertheless, the dimension of the data is reduced and any subsequenct semantic modeling may benefit from that.

4 Latent semantic analysis

Latent semantic analysis (LSA) is a form of linear multivariate analysis on texts represented in matrix form [7]. It uses singular value decomposition (SVD) and typically works from a document-term matrix which may be weighted, e.g., with *tf-idf* before the SVD is applied. SVD factorizes a document-term matrix, $\mathbf{X}$,

$$\mathbf{X} = \mathbf{ULV}', \tag{3}$$

[2] https://radimrehurek.com/gensim/corpora/hashdictionary.html

into the orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ that consist of loadings over documents and terms, respectively. $\mathbf{L}$ is a diagonal matrix with positive singular values. It is usually the dimensions associated with the large singular values that are of interest.

An SVD algorithm is available for many programming languages and these standard implementations can work on small text data. There exists algorithms which can work with the corpus in batches, thus enabling the LSA to work over very large datasets that cannot fit in memory [47].

SVD may also be used on character n-grams [41].

5 Non-negative matrix factorization

Non-negative matrix factorization (NMF) factors a non-negative matrix into two non-negative matrices [20]

$$\mathbf{X} \approx \mathbf{WH}. \tag{4}$$

If the $\mathbf{X}$ is a document-term matrix then $\mathbf{W}$ will be a matrix of loading over documents and $\mathbf{H}$ a matrix of loading over terms. In its basic formulation, the only hyperparameter is the dimension of the factorized space k, i.e., the number of columns of $\mathbf{W}$ and the number of rows of $\mathbf{H}$. The dimension is usually chosen to be smaller than the dimension of the $\mathbf{X}$, thus the product $\mathbf{WH}$ will not be able to reconstruct the original matrix and a residual remains, i.e., "approximative NMF" in contrast to "exact NMF".

Lee and Seung presented simple iterative algorithms for two cost function [20]. In one case, the algorithm minimizes the residual as the Frobenius of the difference between the original matrix and the reconstructed matrix $||\mathbf{X}-\mathbf{WH}||_2$

$$\mathbf{H}_{kj} \leftarrow \mathbf{H}_{kj} \frac{(\mathbf{W}'\mathbf{X})_{kj}}{(\mathbf{W}'\mathbf{WH})_{kj}} \tag{5}$$

$$\mathbf{W}_{ik} \leftarrow \mathbf{W}_{ik} \frac{(\mathbf{XH}')_{ik}}{(\mathbf{WHH}')_{ik}} \tag{6}$$

where subscript means elementwise multiplications. This algorithm has an inherent non-uniqueness: The columns of $\mathbf{W}$ and the rows of $\mathbf{H}$ can be permuted and scalings can be moved between the two matrices. Furthermore, its convergence properties are not straightforward, e.g., if the denominator of equation is or becomes zero. A practical implementation may augment the denominator with a small positive constant.

Probabilistic latent semantic analysis (PLSA) considers the probabilities of co-occurences of documents and terms, $P(d,t)$, as a mixture distribution with k mixtures

$$P(d,t) = \sum_k P(d|k)\, P(k)\, P(t|k), \tag{7}$$

With an extension of the NMF model and suitable normalization of the matrices, NMF can be interpreted in the context of PLSA

$$\mathbf{X} \approx c\,\mathbf{WLH} \tag{8}$$

where $P(d,t) \equiv c\mathbf{X}$, $P(d|k) \equiv \mathbf{W}$, $P(t|k) \equiv \mathbf{H}$ and $P(k) \equiv \text{diag}(\mathbf{L})$.

There exists extension of NMF. For instance, the cost function can be extended with terms for the norms of the matrices $\mathbf{W}$ and $\mathbf{H}$. In contrast to LSA, NMF imposes no orthogonality constraints on any of the vector pairs in $\mathbf{W}$ or $\mathbf{H}$. And whereas principal component-based algorithms results in "holistic" representations, the non-negativity of NMF results in parts-based representation [19].

NMF has been applied in text mining and discovers semantic features [19,32]. The non-orthogonality and non-negative will usually make the interpretation of the factorization less problematic than the factorization from LSA. To overcome the issue of selecting the dimension of the factorized space, multiple independent NMFs can be run with different dimensions and a visualization technique can give an overview of the results [32].

6 Explicit Semantic Analysis

Explicit Semantic Analysis (ESA) creates a semantic representation of a word or a text with the use of an information retrieval technique. ESA represents the word or the text as a weighting over documents. The common method converts a corpus to a *tf-idf*-weighted document-term matrix. When a particular word is to be encoded in the ESA-distributed representation, the vector space representation of the word is multiplied on the document-term matrix. The original report used Wikipedia as the corpus [11]. The attractiveness of the ESA rest on the good performance in semantic relatedness tasks. Another attractive feature is that the dimensions of the space are directly interpretable as they represents documents of the corpus used to estimate the ESA representation, e.g., Wikipedia articles.

Wikipedias may have millions of articles (the English Wikipedia has over 5 million articles), so potentially the representation of each word would have a dimension of several millions. In practice, the number of documents selected is somewhat lower. The original study [11] used a selection of around 240'000 Wikipedia articles after excluding small articles and articles with few incoming and outgoing hyperlinks.

Some research has found that the type of corpus has less relevance for the performance of ESA in document similarity tasks [2].

7 Word embeddings

Word embedding algorithms create semantic representation of words by scanning large corpora, extracting a window of words and modeling the words in the window as a relation between the word and its context, so the final algorithm results in a model where each word gets represented as a point in a low-dimensional space of, say, 100 dimensions, for a schematic example in two dimensions, see Fig. 1. Researchers have suggested a number of algorithms. One early algorithm, hyperspace analogue to language (HAL), built the co-occurence matrix by scanning a corpus with a window size of 10 words with weighting

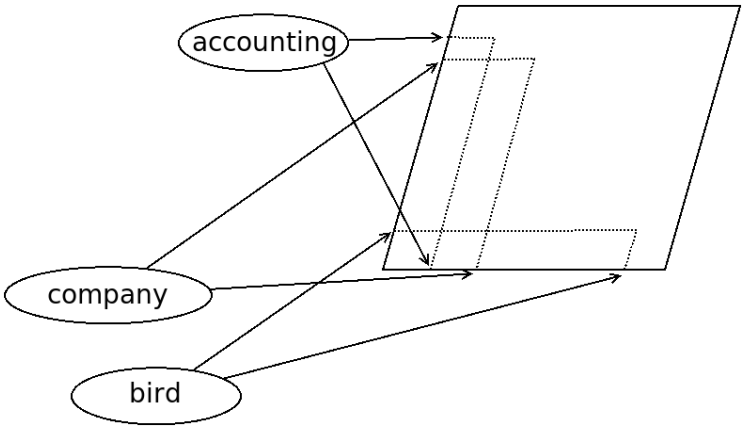

Fig. 1. Schematic representation of a two-dimensional word embedding. Here three words, *accounting*, *company* and *bird*, are embedded in the two-dimensional space. A good word embedding should place similar words close together, e.g., *accounting* and *company* should be closer together, than to *accounting*.

within the window based on the number of words separating the two words of interest [23]. From a low-dimensional representation, distances in the space of the co-occurence matrix yield distances with semantic interpretation and may separate words belonging to categories such as animal names, body parts and geographical locations.

Newer word embedding models build a neural network model between a center word and its context. Mikolov et al. simplified the neural network model to two layers [24]: A linear layer from the input with the dimension corresponding to the size of the vocabulary to the embedding space of 100 or more. From the embedding space a second layer projects to the output with the size of the vocabulary. In the parlance of Mikolov et al., the model that predicts the center word from the context is called continuous back of words (CBOW), while the reverse model where the context is predicted from the center word is termed skip-gram (SG). The output layer has a softmax layer. The optimization of the word embedding model through the softmax layer requires—in its common application—the normalization across the vocabulary. To avoid this computational costly step, methods use what is called noise-contrastive estimation or negative sampling, where a few samples of the vocabulary are used as a form of substitute for the normalization.

The word embedding has initialization parameters. The dimension of the embedding space can have sizes ranging, for instance, from 25 to 300,[3] while the

[3] See, e.g., https://nlp.stanford.edu/projects/glove/.

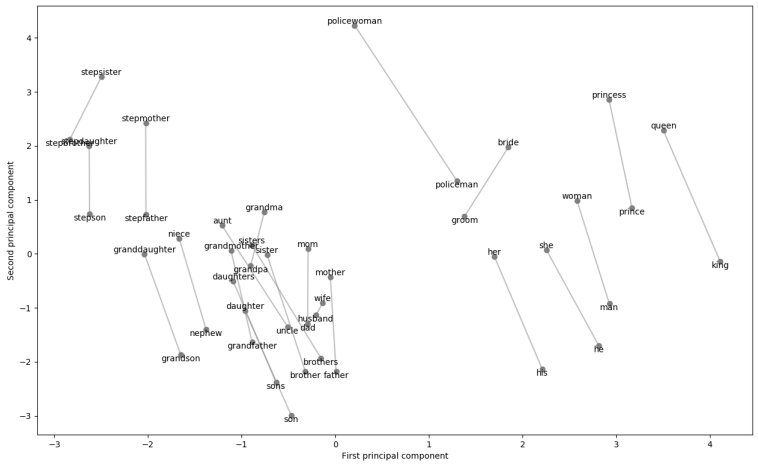

Fig. 2. Principal component projection of male/female words embedded with the 300-dimensional GloVe embedding [36]. Here the female words tend towards the positive part of the second principal component, while male words are projected in the opposite direction. All words have been taken from the word analogy dataset of Mikolov et al. [24].

window size has often the size $(2n + 1) = 5$ [1].[4] An evaluation of the HAL-based model among window sizes of 1, 2, 4, 8 and 10 found that 8 was the size with the largest correlation with a semantic distance meaure derived from a human reaction times experiment. An evaluation of word2vec word embedding for predicting various word analogies found that a short window of 2 words on either side mostly performed the best for syntactic analogies, while a wider window of 10 words on either sides performed the best for semantic tasks [22].

The quality of the word embedding usually depends much on the size of the corpus: the bigger the better [34]. Fig. 2 shows a projection of a few words from a word analogy dataset of Mikolov et al. [24] with the GloVe model trained on a very large corpora.[5] Only the two first principal component are shown, but the second principal component approximately encodes a femaleness/maleness semantics, and family relations, personal pronoun and royalty names seem to cluster: A qualitative evaluation of the word embedding would say that this is a good embedding.

[4] Note it is not always clear how the size is counted. One may count the window size as the total number of words or count it based on the number of words on each side of the word-of-interest.

[5] https://nlp.stanford.edu/projects/glove/

If word-based embeddings are to be used for a sentence, a paragraph or a longer text, then the individual embedded representations should be aggregated into one representation, — a procedure that might be referred to as *word embedding composition*. Different algorithms exist of varying complexity. In some cases simple additive composition will work well [39].

8 Deep learning representations

Deep learning can be used to train neural network-based language models or supervised classification models. Deep learning neural networks consist of multiple layers of parameters and non-linear units. For neural networks modeling sequential data, special recurrent units are also used, e.g., long short-term memory (LSTM) units [40]. Typically each layer will have a vectorial representation, and layers near the output of the neural network may gain some semantic representation after sufficient training.

Some of the deep learning models train a language model on very large datasets predicting a word, a character or a byte from previous parts of the text sequence. For instance, provided with large amount of text data, such as 82 million product reviews from Amazon, a UTF-8 encoded byte-level language model with a recurrent neural network may be trained for one month [38]. In a transfer learning scheme, the 4096-dimensional distributed representation of the deep learning model may be used in training a supervised machine learning model for various other prediction tasks, e.g., sentiment analysis with a specific dataset. Another related study trained a deep learning model on 1,246 million tweets to predict the type of emojis present in each of the tweets [9]. This pretrained model could be adapted with further training to related tasks.

9 Creating explicit semantic representations

Semantic representation can be explicit where relationships between words and concepts are stated with links for lexical or conceptual relations (hyponym, hypernym, synonym, antonym) and where the words and concepts can be associated with features. WordNet is such an example. Specialized graphical user interfaces are developed to set up these relations, see, e.g., [13].

Wikidata at https://www.wikidata.org is a knowledge graph grown out of the Wikipedia community. It describes the concepts corresponding to Wikipedia articles, but also a range of other topics such as scientific articles, lexemes and word forms. Wikidata is multilingual and identifies its items (concepts, lexemes, forms) not by words but by a non-descriptive integer identifier, e.g., the concept of a dog has the identifier 'Q144' while the Danish lexem *gentagelse* has the identifier 'L117'. Wikidata users can collaboratively edit the graph in an specialized online environment featuring revision control where the users can see changes to the graph. Users may also make explicit links to individual items in external resources such as the linguistic resources WordNet, DanNet, ILI and BabelNet, for the WordNet linkage, see [31].

There are various methods to convert lexical and conceptual items represented in a graph to a dense vectorial representation [30,35]. These methods come under the names graph embedding or knowledge graph embedding with specific names such as node2vec ad RDF2vec, An initial process generates 'pseudo-sentences' constructed from random walks in the graph, where the 'words' are nodes, — and possibly links. These pseudo-sentences are then submitted to standard word embedding software such as word2vec.

10 Creating semantic representations with non-linguistic information

In some cases, we have access to non-linguistic data that links to linguistic data: Words, sentences and documents might be associated with a location in space (e.g., geolocation), colors, images and brain activity. Modeling of the joint distribution between the linguistic and non-linguistic data can provide further and perhaps improved semantic representation. There are several large-scale datasets where images and texts are associated. Visual Genome[6] will associate a photo with region, attribute and relationship annotation, e.g., "man playing frisbee", "frisbee is white" and "building behind player". Similarly, the COCO dataset[7] will associate a photo with a full sentence, e.g., "a man in a red shirt throws an orange frisbee".

In a specialized application [33], short neuroanatomical labels associated with 3-dimensional stereotaxic coordinates from human brain mapping studies formed the basis for a statistical model. The resulting model connected the textual label, l, with the physical sites, $\mathbf{z}$, in the brain with a probability density estimate (PDE), $p(\mathbf{z}|l)$, $\mathbf{z} \in \mathbb{R}^3$. When this PDE is evaluated in discrete steps in 3-dimensional space, the PDE can be turned into a vector, $\mathbf{x}_l$, for each label, thus the neuroanatomical label has a distributed representation where each element is a non-negative value. In our specific application we modeled the probability density by kernel density estimation.

$$p(\mathbf{z}|l) = \frac{1}{|\mathcal{N}_l|} \sum_{n \in \mathcal{N}_l} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(\mathbf{z} - \mathbf{x_n})^2}{2\sigma^2}\right) \tag{9}$$

With the vector representation of the labels, neuroanatomical labels that partially overlap in brain space can be searched independently of the othographic representation of the word of its presence from neuroanatomical taxonomies.

11 Evaluating semantic representations

Various methods exist for evaluation of semantic representations of words. Some researchers distinguish between two modes of evaluations: intrinsic and extrinsic [6,8,46]. Intrinsic evaluation establishes dedicated test sets for the evaluation, while extrinsic evaluation uses the semantic representation as part of a

[6] http://visualgenome.org/
[7] http://cocodataset.org

model for a more complicated natural language processing task. Word similarity/relatedness, verbal analogy or word intrusion tasks may exemplify the former, while part-of-speech tagging, chunking, named entity extraction, dependency parsing and morphosyntactic disambiguation are examples of the latter [6,46]. Evaluations may find quite a difference in evaluation between the two modes [6]

In connection with evaluation of embedding models, intrinsic evaluations will typically use the cosine similiarity s_C between the two compared words represented in the embedding space

$$s_C(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}'\mathbf{y}}{\|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_2}. \tag{10}$$

There are other forms of similarity measures, e.g., the Minkowski family of distances [23],

$$d_M(\mathbf{x}, \mathbf{y}) = \sqrt[r]{\sum(|x_i - y_i|)^2}, \tag{11}$$

where a Minkowski distance with $r = 2$ is the common Euclidean distance and $r = 1$ is a city-block distance.

Methods that combine word embedding similarity with graph-based distances on the explicit semantic networks computes a common score. In one work [21], a semantic relatedness score, $\text{rel}(w_i, w_j)$, between two words, w_i and w_j, was defined as

$$\text{rel}(w_i, w_j) = s_c(\mathbf{x}_{w_i}, \mathbf{x}_{w_j}) + (1 - \lambda) \max_{m,n} \frac{1}{\text{dist}(S_{i,m}, S_{j,n})}, \tag{12}$$

where $\mathbf{x}_w$ are the word embeddings and $\text{dist}(S_{i,m}, S_{j,n})$ is the path distance in the semantic network (WordNet in their case) between the two senses $S_{i,m}$ and $S_{j,n}$, — the m'th sense of the i'th word and the n'th sense of the j'th word. The λ parameter mixes between the two parts of the combined score, and with the extra flexibility provided by this parameter, the researcher reported state-of-the-art results on a small range of standard similarity datasets [21].

11.1 Word similarity/relatedness

Datasets with human judgement of similarity/relatedness between pairs of words are popular means of evaluating semantic models. The website wordvectors.org lists 13 word similarity datasets. WordSim-353 (WS-353) with 353 word pairs [10] is probably the most popular and among the oldest.

Similarity and relatedness are similar concepts but often viewed as different with similarity more narrowly defined than relatedness. For instance, *car* and *steering wheel* are 'related' but would in many contexts not be regarded as 'similar'. Another example is *coffee* and *cup*, where in some contexts one word would point to a drink the other to a container [8]. Newer word similarity datasets, such as *SimLex-999*, are larger and may distinguish more stringently between relatedness and similarity [6].

The human scoring used in word similarity tests may be confounded by different effects [3]: A rescoring of a word pair list may not yield the same scores, the humans may differ widely in the similarity score they attribute to the individual word pairs, e.g., scores among 50 humans for the word pair *sun* and *planet* may have used the entire range of possible similarity scores. Furthermore, the human similarity scores may even be affected by the presentation ordering, so, e.g., *the similarity of baby to mother* results in different scores than *the similarity of mother to baby* [3].

With the similarity scored datasets, the models are typically ranked based on the Spearman correlation between the cosine similarity and human similarity judgement. The corpus-based ESA model maintained state-of-art performance on the WordSim-353 dataset for a few years.[8] In 2017, a combined corpus- and knowledge-graph based approach using ConceptNet set a new state-of-the-art [42].

Most of the datasets are in English, but word similarity datasets also exists for other languages.

11.2 Verbal analogies

Verbal analogies usually comes in the form of four words semantically connected, e.g., Germany, Berlin, France, Paris. The task is to predict the fourth word from the three others. Given that the fourth is to be selected from the entire vocabulary, the baseline accuracy for pure guessing would be close to zero.

The semantic testing with verbal analogies became popular after a demonstration with distributed word representations showed that the analogies could be estimated with simple algebra in the word embedding space and the cosine similarity [26,24]. This method is sometimes found under the names *vector difference* or *vector offset* [45]. The prototypical example is man is to king, what woman is to queen.

With the analogy on four words

$$a : a^* :: b : b^*, \tag{13}$$

the standard guess on the fourth word, b^*, is the analogy function:

$$b^* = \underset{b'}{\operatorname{argmax}}\, s_C(\mathbf{x}_{b'}, \mathbf{x}_{a*} - \mathbf{x}_a + \mathbf{x}_b), \tag{14}$$

where $\mathbf{x}$ is the word embedding and $s_C()$ is the cosine similarity. There are other suggested analogy functions. A variation with multiplication and a division instead of addition and substraction may perform slightly better [22].

Datasets with word analogies may focus on specific relations, e.g., agent-goal, object-typical action [15][9] or syntactic relations [26], see also the 15 relations in Table 2 in [45], e.g., hypernym, meronym, location or time association. As an

[8] https://aclweb.org/aclwiki/WordSimilarity-353_Test_Collection_(State_of_the_art)

[9] The dataset is available at https://sites.google.com/site/semeval2012task2/.

example, the first few lines of the popular dataset from Mikolov et al. from 2013 [24] display analogies between countries and capitals:

```
Athens Greece Baghdad Iraq
Athens Greece Bangkok Thailand
Athens Greece Beijing China
Athens Greece Berlin Germany
```

Apart from English, other languages with word analogy datasets are, e.g., Czech [44] and German [17].

Evaluations in 2017 with fastText trained for 3 days on either the very large Common Crawl data set or a combination of the English Wikipedia and news datasets set a new state-of-the-art on 88.5% for the accuracy on a popular word analogy dataset when fastText training included sub-word features [25]. Pre-trained fastText models are available.

11.3 Word intrusion

Word intrusion tasks present a series of words with one of the words being an outlier and should be detected. In one of our works to evaluate distributed semantic representation for Danish [34], we established a small dataset with 100 sets of words, where each set consisted of 4 words, e.g., corresponding to—in English—*apple*, *pear*, *cherry* and *chair* as one example and *grass*, *tree*, *flower* and *car* as another example.[10] In this case, we would expect random guessing to produce an accuracy of 25%.

The identification of the outlier with a distributed semantic representation projects each of the set of words to a low-dimensional space, e.g., a word2vec representation and then identifies the outlier based on the distances in this space. For instance, one may compute the overall average across words, compute the distance from each of the words to the average and select the one furthest away as the outlier. If the distributed semantic representation is good, we may expect the distance to represent semantic outlierness well.

In an evaluation on a Danish set of words [34], an ESA model was found to yield an accuracy on 73%, while a word2vec-based model trained on large corpora yielded almost as good a performance.

The Test of English as a Foreign Language (TOEFL) synonym test is related to the word intrusion task. The TOEFL task also selects among 4 words, but for the TOEFL, the algorithm should select the most similar word to a given query word. TOEFL was originally used to test the latent semantic analysis model [18].

11.4 Sentiment analysis

Sentiment is one aspect of semantics. If the distributed semantics representation is good we should also expect that certain aspects of it can reveal the sentiment well. There exists many resources, where the sentiment or the emotion of

[10] https://github.com/fnielsen/dasem/blob/master/dasem/data/four_words.csv

a text have been assigned manually. This may be on levels from words, phrases, sentences or paragraphs to complete texts. For instance, AFINN assigns an integer between −5 to 5 to each word or small phrase [29], e.g., the English word *excellent* gets assigned the value 3.[11]

A distributed semantic representation pre-trained on an independent resource can represent the sentiment-labeled words, and if this representation is semantically accurate we may expect that a supervised learning algorithm trained to predict sentiment will easily be able to generalize, i.e., able to predict the sentiment of words, which have not been used during the supervised learning. For the Danish version of AFINN, we found that the scheme worked well and that the trained machine learning classifier could even point to words where the sentiment label may be less than optimal and should be reconsidered [34]. For instance, 'udsigtsløs' (futile) is labeled as positive in AFINN which can be considered an error. The analysis also pointed to words with 'implicit' negativity, e.g., benådet (pardoned), tilgiver (forgives) and præcisere (clarify) which could indicate a change from negativity.

12 Problems: polysemy, homograph, bias and compounds

Various problems exist with the semantic representations. For distributed semantic representation based on the orthographic representation of words, polysemy and homograph are problems: That two words with the same othographic representations can have different semantics, e.g., *jaguar* (disregarding case) can mean a feline, a car make or an operating system version. Maintaining the case of the word will help slightly on the polysemy/homograph problem, but will expand the vocabulary of the corpus, creating a problem with the estimation of the semantic model. For some of the pre-trained models based on a very large corpus, the case is maintained. Applying part-of-speech (POS) tagging before the word embedding training and using the POS-tag—together with its associated word—can disambiguate word classes (e.g., to *fly* vs. a *fly*), but does not help with polysemy/homograph within word classes. Another approach uses word sense disambiguation (WSD) before word embedding. A system called SensEmbed uses babelfy[12] for the disambiguation before training a CBOW word2vec model [14]. Rather than words, it is senses that are embedded, thus a word with multiple senses will be embedded at several different points in the embedding space. WSD may use a semantic network, such as WordNet or BabelNet, so embedded senses correspond to concepts in the semantic network. In this case, the corpus-based distributional sense embedding may be augmented with the information from the explicit semantic network. The SensEmbed researchers argue that the semantic network are particular helpful for rare words and gives as an example the highly related pair *orthodonist-dentist*, where *orthodonist* only

[11] https://github.com/fnielsen/afinn/blob/master/afinn/data/AFINN-en-165.txt

[12] http://babelfy.org/

occured 70 times in the corpus, making its distributed representation not accurate, and the corpus-based *orthodonist-dentist* similarity estimation poor. However, in BabelNet, the semantic network they used, the two words were directly linked and the combined distributional-semantic network similarity model can determine the synonymity of the word pair [14]. Other methods that handles polysemy/homographs, but without direct WSD, model the word context with shallow or deep learning models [43,37].

Bias in word embeddings should also be considered. These biases reflect the stereotypes in the corpus used to train the word embedding. Bias may be related to gender where words for occupation are associated with a specific gender, e.g., a reflected in the title "man is to computer programmer as woman is to homemaker" [5]. If one is aware of the bias, then it might be possible to correct it.

Compounds, that occur frequently in languages such as German and Danish, may pose a problem as they are often rare words. Decompounding methods exist [16], and may help if the individual compounded words are common. But it may also be worth considering fastText that with its modeling of n-grams are able to handle out-of-vocabulary words. For instance, consider the Danish compound noun *bogføringsvirksomhed* (*bookkeeping company*) which is out-of-vocabulary in a Dasem-trained[13] fastText model, but fastText is nevertheless able to identify semantically related within-vocabulary words when only the n-gram representation of the word is used:

forretningsvirksomhed (business)
rådgivningsvirksomhed (advisory/consultant business)
revisionsvirksomhed (auditing business)

Here the last compound, *virksomhed*, is shared.

Yet another issue is misspellings and optical character recognition (OCR) errors in the corpus used to train the distributed representation model. Such erroneous words will also get projected in the embedding space. Whether this is a problem depends on the task: If the task is just to project words or to select among a pre-specified number of words, the presence of misspellings may not matter, but in the case where the task is to select the closest word to a given query, misspelling and OCR errors will likely decrease the performance.

13 What does it mean?

It is unclear what form the semantic space should have to best represent semantics. In many cass, the semantic space is estimated with reference to an Euclidean space, but it need not be the best. Researchers have argued that other forms of spaces could yield a better representations than a Euclidean space for certain semantic structures. In a tree, the number of leaf nodes grows exponentially with

[13] Dasem is a Python package for Danish semantics available at https://github.com/fnielsen/dasem

the distance to the root node of the tree, if the tree is having a specific branching factor. A continuous space with a similar characteristics is hyperbolic geometry. Embedding in this space has been referred to as Poincaré embedding, and the researchers showed that this form of embedding may be better than Euclidean embedding for a hierarchical problem, the WordNet noun hierarchy [28].

Due to polysemy/homographs a semantic word space could be regarded as violating the triangle inequality [27]. Consider the two word pairs (*bee*, *fly*) and (*fly*, *travel*). Here the semantic similarity within the pairs are high and *fly* is a homograph, so the semantic distance from *bee* to *travel* is short when via *fly*, but longer when the pair (*bee*, *travel*) is viewed in isolation.

Semantic spaces may need to be of a certain size to faithfully represent some concept relations. Consider a small semantic star network with one super concept, A, and 4 subconcepts, $B_i,\ i \in [1, 2, 3, 4]$, where the semantic distances should be $\text{dist}(A, B_i) = 1$ and $\text{dist}(B_i, B_j) = c,\ c > 1$. In 2 dimensions, it would not be possible to place the subconcepts B_i in a configuration so all 6 distances between them are equal. Here the addition of a further dimension would enable the four subconcepts to be places in a 3-dimensional tetraedic configuration where the subconcepts all have similar distances to each other.[14]

When spaces have many dimensions certain effects usually arise: distance concentration and hubness. The latter means that, e.g., for a *tf-idf*-weighted corpus some documents of the corpus will very often appear as nearest neighbors to other documents.

14 Acknowledgment

We would like to thank Innovation Fund Denmark for funding through the DABAI project.

References

1. Al-Rfou, R., Perozzi, B., Skiena, S.: Polyglot: Distributed Word Representations for Multilingual NLP. Proceedings of the Seventeenth Conference on Computational Natural Language Learning pp. 183–192 (June 2014), https://arxiv.org/pdf/1307.1662.pdf
2. Anderka, M., Stein, B.: The ESA retrieval model revisited. Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval pp. 670–671 (2009)
3. Bhattacharyya, M., Suhara, Y., Rahman, M.M., Krause, M.: Possible Confounds in Word-based Semantic Similarity Test Data. Companion of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing - CSCW '17 Companion pp. 147–150 (2017)

[14] We can place the super concept at $A = (0, 0, 0)$ and the subconcepts at $(1, 1, 1)$, $(-1, -1, 1)$, $(-1, 1, -1)$ and $(1, -1, -1)$. All subconcepts have the same distance to the super concept and the same distance to all other subconcepts.

4. Bingham, E., Mannila, H.: Random projection in dimensionality reduction: applications to image and text data. Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining pp. 245–250 (August 2001)
5. Bolukbasi, T., Chang, K.W., Zou, J.Y., Saligrama, V., Kalai, A.T.: Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings. Advances in Neural Information Processing Systems 29 (July 2016), https://papers.nips.cc/paper/6228-man-is-to-computer-programmer-as-woman-is-to-homemaker-debiasing-word-embeddings.pdf
6. Chiu, B., Korhonen, A., Pyysalo, S.: Intrinsic Evaluation of Word Vectors Fails to Predict Extrinsic Performance. Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP (August 2016), https://sites.google.com/site/repevalacl16/26_Paper.pdf
7. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science 41, 391–407 (September 1990)
8. Faruqui, M., Tsvetkov, Y., Rastogi, P., Dyer, C.: Problems With Evaluation of Word Embeddings Using Word Similarity Tasks. Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP (May 2016), https://sites.google.com/site/repevalacl16/11_Paper.pdf
9. Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., Lehmann, S.: Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing pp. 1616–1626 (August 2017), http://aclweb.org/anthology/D17-1169
10. Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., Ruppin, E.: Placing search in context: the concept revisited. ACM Transactions on Information Systems 20, 116–131 (January 2002)
11. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. Proceedings of the 20th international joint conference on Artifical intelligence pp. 1606–1611 (January 2007), http://www.aaai.org/Papers/IJCAI/2007/IJCAI07-259.pdf
12. Ganchev, K., Dredze, M.: Small Statistical Models by Random Feature Mixing (2008)
13. Henrich, V., Hinrichs, E.: GernEdiT: A Graphical Tool for GermaNet Development. Proceedings of the ACL 2010 System Demonstrations pp. 19–24 (July 2010)
14. Iacobacci, I., Pilehvar, M.T., Navigli, R.: SensEmbed: Learning Sense Embeddings for Word and Relational Similarity. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing pp. 95–105 (July 2015), http://aclweb.org/anthology/P15-1010
15. Jurgens, D.A., Turney, P.D., Mohammad, S.M., Holyoak, K.J.: SemEval-2012 task 2: measuring degrees of relational similarity. *SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012) pp. 356–364 (June 2012), http://www.aclweb.org/anthology/S12-1047
16. Koehn, P., Knight, K.: Empirical Methods for Compound Splitting. Proceedings of the tenth conference on European chapter of the Association for Computational

Linguistics pp. 187–193 (February 2003), https://arxiv.org/pdf/cs/0302032.pdf
17. Köper, M., Scheible, C., im Walde, S.S.: Multilingual Reliability and "Semantic" Structure of Continuous Word Spaces. Proceedings of the 11th International Conference on Computational Semantics pp. 40–45 (April 2015), http://www.aclweb.org/anthology/W15-0105
18. Landauer, T.K., Dumais, S.T.: A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review 104, 211–240 (1997)
19. Lee, D.D., Seung, S.: Learning the parts of objects by non-negative matrix factorization. Nature 401, 788–791 (October 1999)
20. Lee, D.D., Seung, S.: Algorithms for non-negative matrix factorization. Advances in Neural Information Processing Systems 13 pp. 556–562 (2001), http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf
21. Lee, Y.Y., Ke, H., Huang, H.H., Chen, H.H.: Combining Word Embedding and Lexical Database for Semantic Relatedness Measurement. Proceedings of the 25th International Conference Companion on World Wide Web pp. 73–74 (2016)
22. Linzen, T.: Issues in evaluating semantic spaces using word analogies. Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP pp. 13–18 (August 2016), http://www.aclweb.org/anthology/W16-2503
23. Lund, K., Burgess, C.: Producing high-dimensional semantic spaces from lexical co-occurrence. Behavior Research Methods 28, 203–208 (June 1996)
24. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient Estimation of Word Representations in Vector Space (January 2013), https://arxiv.org/pdf/1301.3781v3
25. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in Pre-Training Distributed Word Representations (December 2017), https://arxiv.org/pdf/1712.09405.pdf
26. Mikolov, T., tau Yih, W., Zweig, G.: Linguistic Regularities in Continuous Space Word Representations. Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies pp. 746–751 (June 2013), http://www.aclweb.org/anthology/N13-1090
27. Neelakantan, A., Shankar, J., Passos, A., McCallum, A.: Efficient Non-parametric Estimation of Multiple Embeddings per Word in Vector Space. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) pp. 1059–10699 (2014), http://www.aclweb.org/anthology/D14-1113
28. Nickel, M., Kiela, D., Kiela, D.: Poincaré Embeddings for Learning Hierarchical Representations. Advances in Neural Information Processing Systems 30 (May 2017), https://arxiv.org/pdf/1705.08039.pdf
29. Nielsen, F.Å.: A New ANEW: Evaluation of a Word List for Sentiment Analysis in Microblogs. Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages pp. 93–98 (May 2011), http://ceur-ws.org/Vol-718/paper_16.pdf
30. Nielsen, F.Å.: Wembedder: Wikidata entity embedding web service (October 2017), https://arxiv.org/pdf/1710.04099
31. Nielsen, F.Å.: Linking ImageNet WordNet Synsets with Wikidata. WWW '18 Companion: The 2018 Web Conference Companion, April 23–27, 2018, Lyon, France pp. 1809–1814 (April 2018), http://www2.compute.dtu.dk/pubdb/views/edoc_download.php/7090/pdf/imm7090.pdf

32. Nielsen, F.Å., Balslev, D., Hansen, L.K.: Mining the posterior cingulate: segregation between memory and pain components. NeuroImage 27, 520–532 (September 2005)
33. Nielsen, F.Å., Hansen, L.K.: Modeling of activation data in the BrainMap database: detection of outliers. Human Brain Mapping 15, 146–156 (March 2002)
34. Nielsen, F.Å., Hansen, L.K.: Open semantic analysis: The case of word level semantics in Danish. Human Language Technologies as a Challenge for Computer Science and Linguistics pp. 415–419 (October 2017), http://www2.compute.dtu.dk/pubdb/views/edoc_download.php/7029/pdf/imm7029.pdf
35. Nielsen, F.Å., Hansen, L.K.: Inferring visual semantic similarity with deep learning and Wikidata: Introducing imagesim-353. Proceedings of the First Workshop on Deep Learning for Knowledge Graphs and Semantic Technologies pp. 56–61 (April 2018), http://www2.compute.dtu.dk/pubdb/views/edoc_download.php/7102/pdf/imm7102.pdf
36. Pennington, J., Socher, R., Manning, C.D.: GloVe: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) p. 1532–1543 (2014), http://www.emnlp2014.org/papers/pdf/EMNLP2014162.pdf
37. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. Proceedings of NAACL-HLT 2018 pp. 2227–2237 (March 2018), https://arxiv.org/pdf/1802.05365.pdf
38. Radford, A., Józefowicz, R., Sutskever, I.: Learning to Generate Reviews and Discovering Sentiment (April 2017), https://arxiv.org/pdf/1704.01444.pdf
39. Scheepers, T., Kanoulas, E., Gavves, E.: Improving Word Embedding Compositionality using Lexicographic Definitions. Proceedings of the 2018 World Wide Web Conference (2018)
40. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Networks 61, 85–117 (October 2014)
41. Soboroff, I.M., Nicholas, C.K., Kukla, J.M., Ebert, D.S.: Visualizing document authorship using n-grams and latent semantic indexing. Proceedings of the 1997 workshop on New paradigms in information visualization and manipulation (1997)
42. Speer, R., Chin, J., Havasi, C.: ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence pp. 4444–4451 (December 2016), https://arxiv.org/pdf/1612.03975.pdf
43. Sun, Y., Rao, N., Ding, W.: A Simple Approach to Learn Polysemous Word Embeddings (July 2017), https://arxiv.org/pdf/1707.01793.pdf
44. Svoboda, L., Brychcín, T.: New Word Analogy Corpus for Exploring Embeddings of Czech Words. Computational Linguistics and Intelligent Text Processing pp. 103–114 (2018)
45. Vylomova, E., Rimell, L., Cohn, T., Baldwin, T.: Take and Took, Gaggle and Goose, Book and Read: Evaluating the Utility of Vector Differences for Lexical Relation Learning. Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics pp. 1671–1682 (2016), http://www.aclweb.org/anthology/P16-1158
46. Wróblewska, A., Krasnowska-Kieraś, K., Rybak, P.: Towards the evaluation of feature embedding models of the fusional languages. Human Language Technologies as a Challenge for Computer Science and Linguistics pp. 420–424 (November 2017)
47. Řehůřek, R.: Fast and Faster: A Comparison of Two Streamed Matrix Decomposition Algorithms (February 2011), https://arxiv.org/pdf/1102.5597.pdf