Linguistic Analogies in Word Embeddings: Where Are They?

Riccardo Contessi, Paolo Fosci[a] and Giuseppe Psaila[b]
University of Bergamo, DIGIP, Viale Marconi 5, 24044 Dalmine (BG), Italy

Keywords: Word Embeddings, Linguistic Analogies, Vector Semantics, Semantic Similarity, NLP.

Abstract: *Word Embedding* has greatly improved Natural-Language Processing. In word-embedding models, words are represented as vectors in a multi-dimensional space; these vectors are trained through neural networks, by means of very large corpora of textual documents. Linguistic analogies are claimed to be encoded within word-embedding models, in such a way that they can be dealt with through simple vector-offset operations. This paper aims to give an answer to the following research question: *given a word-embedding model, are linguistic analogies really present?* It seems rather unrealistic that complex semantic relationships are encoded within a word-embedding model, which is trained to encode positional relationships between words. The investigation methodology is presented, and the results are discussed. This leads to the following question: "Linguistic analogies in *Word Embeddings*: where are they?".

1 INTRODUCTION

Natural-Language Processing (NLP) has achieved impressive results, which rely on "Neural Networks": a large corpus of textual documents is used to train the inner model of a neural network, following the idea that if a sequence of words $(w_1, \ldots, w_n)$ is submitted to the network, it should output the most probable next w_{n+1} word. The internal representation learned by a neural network trained in this way is a matrix: rows correspond to single words, while columns denote a pool of weights that are used to identify the most probable w_{n+1} word. In such a vision, a word is represented as a vector in a multi-dimensional space; the overall pool of vectors (or matrix) is named *Word Embedding*.

Modern *Word Embedding* was introduced by (Mikolov et al., 2013a), where the above-mentioned approach was presented. In particular, in the paper, the authors presented an interesting and remarkable, although unexpected, feature: the existence of semantic relationships in the form of "linguistic analogies". The most famous linguistic analogy can be illustrated as follows: `King` stands to `Man` as `Queen` stands to `Woman`. This analogy gives rise to a very famous vector operation:

`King - Man + Woman` $\approx$ `Queen`

that could let people think that there is a hidden vector (the one that results from `King - Man`) whose meaning should be "supreme chief".

As a consequence, many dissemination materials graphically show such a hypothetic behavior leading researchers to strongly believe in it. However, the authors of the present paper wanted to understand *Word Embedding*, in particular why a word-embedding model should be able to represent linguistic analogies and semantic relations. Indeed, the existence of linguistic analogies within word-embedding models seems quite strange, because an embedding model is, by nature, a statistical, not semantic, model. Indeed, many past research works pointed out criticisms about this vision; moreover, it is still a widely held opinion that linguistic analogies are encoded in word-embedding models as vector-offset operations.

The goal of this paper is to address the following research question: *given a word-embedding model, are linguistic analogies really present?*
Unfortunately, (Mikolov et al., 2013a) used a dataset that is no longer publicly available; furthermore, it empirically showed linguistic analogies by exploiting a vector operation named **MostSimilar**, which was not formally defined; although this operation is present in the Python library *Gensim*, which is considered the reference library for managing *Word Embeddings*, it is not clear if it was actually used in (Mikolov et al., 2013a).

The contribution of the paper is the following: (i) the concepts of *Word Embedding* and "linguistic

[a] https://orcid.org/0000-0001-9050-7873
[b] https://orcid.org/0000-0002-9228-560X

Contessi, R., Fosci, P. and Psaila, G.
Linguistic Analogies in Word Embeddings: Where Are They?.
DOI: 10.5220/0013830300003985

In *Proceedings of the 21st International Conference on Web Information Systems and Technologies (WEBIST 2025)*, pages 447-458
ISBN: 978-989-758-772-6; ISSN: 2184-3252

analogies" are introduced; (ii) the operation **MostSimilar** is formally defined, by interpreting the implementation in the *Gensim* library, and an alternative (more intuitive) operation named **SimilarByVector** (provided by *Gensim* as well) is considered; (iii) an investigation methodology is proposed and applied on a publicly available word-embedding model; (iv) finally, the results of the experiments are discussed, obtaining the question in the title (Linguistic analogies in *Word Embeddings*: where are they?).

The novelty of the contribution relies on a substantial criticism to the approaches followed by previous works: specifically, visual representations and evaluations have never considered the representational space actually used by word-embedding models.

The remainder of the paper is organized as follows. Section 2 briefly resumes the concepts of *Word Embedding* and "linguistic analogy", so as to further report about related work on the research question of this paper. Section 3 formally presents the various vector operations that could be used for managing linguistic analogies. Section 4 introduces the investigation methodology that was followed to answer the research question, and Section 5 presents the results of the experiments. Section 6 discusses the results of the investigation. Finally, Section 7 draws the conclusions and highlights possible directions for future works.

2 BACKGROUND AND RELATED WORK

The goal of this section is to introduce the background of the paper and, then, present the related work. Since all considerations are built around the concepts of *Word Embedding* and linguistic analogy, the first part of this section presents these concepts. The second part discusses the related work.

2.1 Word Embedding

Word Embedding is a technique for Natural-Language Processing (*NLP*) that represents a (possibly-large) vocabulary of words $V = \{w_1, w_2, \ldots, w_n\}$ in a distributed way: each term $w \in V$ is assigned to a vector in a d-dimensional space, i.e., given a word $w_i \in V$ it is represented by a vector $\mathbf{v_i} \in \mathbb{R}^d$. This continuous and dense representation should allow for a more effective capture of relationships among words in natural language; compared to previously developed techniques such as *bag-of-words* (Harris, 1954) or *TF-IDF* (Salton et al., 1975), *Word Embedding* has largely demonstrated its effectiveness. In recent years, *Word Embedding* has led to significant advancements in *NLP*, due to its ability to geometrically model similarities and relationships among words.

Formally speaking, given a set of words $W = \{w_1, w_2, \ldots, w_n\}$, an embedding is a function $f : W \rightarrow \mathbb{R}^d$ that maps each word w_i to a vector $\mathbf{v}_i \in \mathbb{R}^d$ (Mandelbaum and Shalev, 2016).

The vector representation is built in such a way that vectors that represent words that frequently appear in similar contexts are located nearby in space. This reflects the idea that the meaning of a word depends on the context in which it is used. Indeed, the idea is that the semantics of words is implicitly and indirectly captured: words with similar meaning or that are strongly correlated are represented by vectors that point to the same region of space; this is usually true for words that frequently occur in the same speech context, such as the relationship between "doctor" and "hospital", and, to some extent, syntactic relationships as well, such as the link between verbs and their inflected forms.

Several models propose different strategies to learn these vectors and they apply different techniques. *Word2vec* (Mikolov et al., 2013a) leverages the prediction of local contexts, and relies on a recurrent neural network to train the model (indeed, the embedding is the weight matrix of the inner linear layer). *FastText* (Bojanowski et al., 2017) enriches the representation by exploiting word morphology through character n-grams, but it still relies on a recurrent neural network. *GloVe* (Pennington et al., 2014) exploits global co-occurrences within the corpus; differently from *Word2vec* and *FastText*, the model is computed as a co-occurrence matrix of words within the training texts.

Apart from the specific technical approach, the idea is as follows: given a sequence of n words $(s_1, \ldots, s_n)$, through the word-embedding model it should be possible to predict the s_{n+1} word that is most likely to follow the given n words.

Although the training process can produce vectors of different length, models are usually normalized in such a way that all vectors have length 1; the claimed reason is to easily compute the cosine similarity between vectors. As a result, the actual representational space of a word-embedding model is a d-dimensional hyper-sphere with radius 1.

2.2 Linguistic Analogies

Linguistic analogies are relationships between pairs of words, which are based on semantic relationships between words. A typical example is: *a king stands to a man as a queen stands to a woman*. The implicit

semantic relationship is the concept of "person with absolute power". In a more formal way, the above analogy could be stated as:

`(king : man) = (queen : woman)`

In the paper (Mikolov et al., 2013a), the authors launch the idea that the difference between vectors representing the words in the left and in the right side of the previous analogy should be two more or less coincident vectors, i.e., the semantic relationship "person with absolute power" or "supreme chief" should be obtained by subtracting the vector "man" from the vector "king"; similarly, the same vector should be obtained by subtracting the vector "woman" from the vector "queen".

Consequently, given vectors that represent the words "king", "man", and "woman", the vector "queen" should be approximately obtained by the vector operation hereafter reported:

$$\texttt{king} - \texttt{man} + \texttt{woman} \approx \texttt{queen}$$

This property has attracted considerable interest, as it suggests that the vector space of *Word Embeddings* not only preserves some semantic similarity but also encodes more complex conceptual relationships. However, thinking that a representation of words, which is built to capture co-occurrences of words in the same speech contexts, is also able to encode semantic relationships within the same representational space raises several concerns. The purpose of this paper is to investigate this aspect.

2.3 Related Work

The fact that word-embedding models actually encode linguistic analogies and semantic relationships has been long debated. In particular, the fundamental question is about the way vector operations are performed. The reader can refer to Section 3.1, which reports a formal definition of vector operations, as intended by (Mikolov et al., 2013b), in which vectors representing input words are excluded from the potential result. In this respect, (Linzen, 2016) observed that if this exclusion is not performed, the correctness of vector operations drops very close to zero. On the same line, (Levy and Goldberg, 2014) proposed to consider the cosine similarity between offset vectors, but this proposal has not been further followed.

The paper (Rogers et al., 2017) presented a widespread analysis of critical issues about linguistic analogies in *Word Embeddings*. However, while (Rogers et al., 2017) focuses on the statistical perspective, the present paper continues on the same way, by considering the complementary perspective of the representational space.

In order to test the existence of linguistic analogies in word-embedding models, several authors defined corpora of categorized word pairs. The `Google set` was introduced by (Mikolov et al., 2013b) (the seminal paper about linguistic analogies); the "Better Analogy Test Set" (BATS) was introduced by (Gladkova et al., 2016), with the goal of enriching the "Google set" with a large number of categories, choosing a single category for each word (avoiding repetition of words across categories); furthermore, the older corpus of semantic analogies, named "SemEval-2012" (Jurgens et al., 2012), was used as a reference of semantic relationships between words to test analogies (Finley et al., 2017). However, these data sets are made by humans and can be biased, as far as the association of a word to a specific category is concerned (in case of polysemous words).

Issues related to linguistic analogies emerge for multi-lingual translations as well, typically managed through cross-language embedding (Garneau et al., 2021). Clearly, ways of saying complicate the task, and simple embedding limited to words could be insufficient.

3 OPERATIONS ON VECTORS

In the literature, a plethora of papers depict the above-reported example of linguistic analogy with figures that are very similar to Figure 1a. However, this representation is not accurate and is misleading, for several reasons.

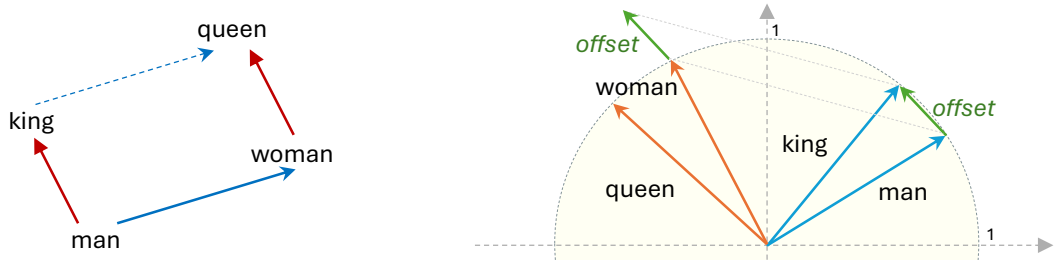
(i) It does not take into account that the representational space is a d-dimensional hyper-sphere.

(ii) Points represent words, but words are represented by vectors in the d-dimensional space, not by points on a plane or in a 3-dimensional space.

(iii) Furthermore, how vector operations are actually performed could significantly change the result and affect reproducibility.

Figure 1b provides the correct representation, for the simplified case of a 2-dimensional space. First of all, the vectors that represent words are all rooted in the origin of axes and reach the sphere (a circle in 2 dimensions) with radius 1; second, notice the effect of computing a vector offset and later applying it to another vector: the result is outside the representational space of the 2-dimensional sphere; furthermore, it is very far away from the expected resulting vector. Indeed, this is not how vector-offset operations are actually defined/performed.

In this section, we consider different definitions, as they were explored in the literature and beyond,



(a) Vector operations as usually presented.

(b) Actual vector operations.

Figure 1: Representations of vector-offset operations.

always considering the actual representation space of unit hyper-spheres.

3.1 Most Similar

The work (Mikolov et al., 2013a) explored various vector arithmetic operations to compute linguistic analogies. To evaluate these relationships, a method known as **MostSimilar**, later implemented in the *Gensim* library (Řehůřek and Sojka, 2010), was employed.

MostSimilar (GenSim Version). Given a set of *positive* words $P = \{w_1, w_2, \ldots, w_m\}$ and a set of *negative* words $N = \{w_{m+1}, w_{m+2}, \ldots, w_{m+n}\}$, each word w_i is represented by its embedding vector $\mathbf{v}_{w_i} \in \mathbb{R}^d$.

- **Step 1: Computing the query vector.** The "query vector" $\mathbf{v}_q$ is computed as a weighted average of the word vectors from both sets, where each word is associated with a weight $\alpha_i \in \{+1, -1\}$. Specifically, $\alpha_i = +1$ if $w_i \in P$, and $\alpha_i = -1$ if $w_i \in N$. The formula below reflects the implementation of the **MostSimilar** method in the *Gensim* library:

$$\mathbf{v}_q = \frac{1}{|P| + |N|} \sum_{w_i \in P \cup N} \alpha_i \times \mathbf{v}_{w_i} \tag{1}$$

  This formulation ensures that all the vectors, both positive and negative, contribute to a single weighted mean rather than computing separate averages.

- **Step 2: Normalizing the query vector.** The query vector $\mathbf{v}_q$ is named this way because it is later used to query the vector space, so as to find out the closest vector to $\mathbf{v}_q$. Since the vector space is normalized to unit vectors, $\mathbf{v}_q$ is normalized as well. Thus, in the following, the normalized version $\overline{\mathbf{v}}_q = norm(\mathbf{v}_q)$ is used.

- **Step 3: Querying the vector space.** Starting from the normalized query vector $\overline{\mathbf{v}}_q$, the top k words whose representing vectors $\mathbf{v}_w$ have the highest cosine similarity with $\overline{\mathbf{v}}_q$ are selected, excluding the input words in $P \cup N$:

$$\begin{aligned} &\texttt{MostSimilar}(P,N) = \\ &= \arg\max_{w \in V | w \notin P \cup N} \cos(\mathbf{v}_w, \overline{\mathbf{v}}_q) = \\ &= \arg\max_{w \in V | w \notin P \cup N} \frac{\mathbf{v}_w \cdot \overline{\mathbf{v}}_q}{\|\mathbf{v}_w\| \|\overline{\mathbf{v}}_q\|} \end{aligned} \tag{2}$$

Example. To illustrate, consider a simplified model in a 3-dimensional space ($d = 3$ is clearly insufficient to obtain an effective model, however it is worth for the sake of clarity), as illustrated in Figure 2. Specifically, Figure 2a illustrates Step 1, while Figure 2b illustrates Step 3.

The example of linguistic analogy is:

$$\texttt{(trains : train)} \approx \texttt{(dogs : dog)}$$

consequently, the following vector operation is performed:

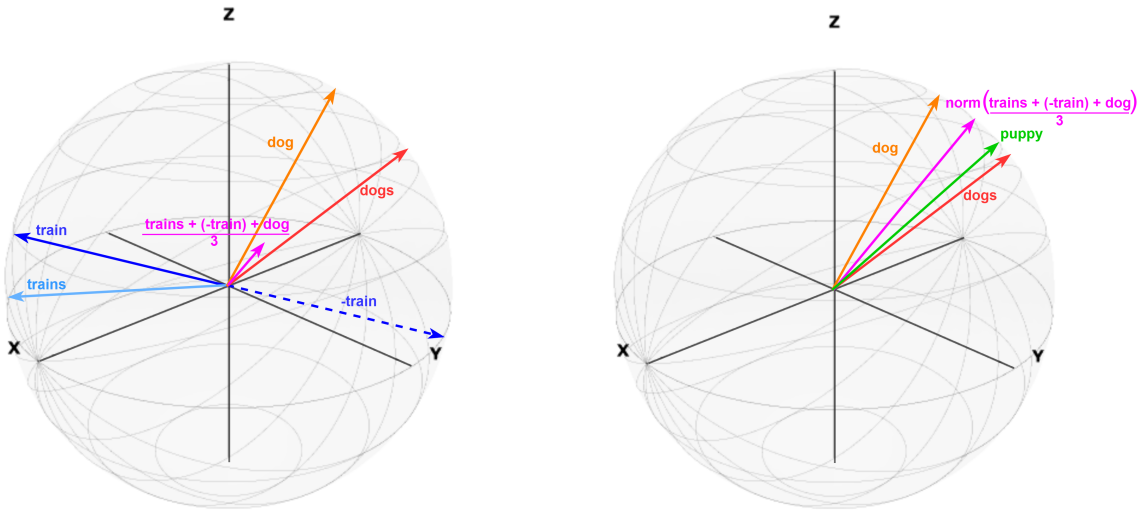
$$\texttt{trains - train + dog} \approx \texttt{dogs}$$

where the semantic relationship is "plural of".

In Figure 2a, the vectors for "train" and "trains" are the blue and light blue vectors, respectively; for the sake of clarity, the dashed blue vector depicts "-train". Similarly, the vectors for "dog" and "dogs" are the orange and red vectors, respectively. The resulting query vector $\mathbf{v}_q$ is the short fuchsia vector.
Figure 2b focuses on Step 3. For this reason, the normalized query vector $\overline{\mathbf{v}}_q$ is depicted (fuchsia vector), while the vectors for "trains", "train" and "-train" are not reported, for the sake of clarity. In addition, the green vector that represents "puppy" is added. Notice that in such a configuration, "dog" is still the closest vector to $\overline{\mathbf{v}}_q$: however, it is not returned because "dog" is discarded, since it is an input word. Consequently, usually "dogs" is returned.

MostSimilar (Mikolov Version). In the work (Mikolov et al., 2013a), a slightly different version of the vector operation was presented. The difference concerns Step 1, in which the query vector is computed based on the following equation:

$$\mathbf{v}_q = \sum_{w_i \in P \cup N} \alpha_i \times \mathbf{v}_{w_i} \tag{3}$$



(a) Computing the query vectors.

(b) Searching for the most similar vector after normalization.

Figure 2: 3-D visual representation of **MostSimilar**.

The result is a longer vector compared to the previous formulation. However, since Step 2 still normalizes to unit the query vector, Step 3 obtains necessarily identical results.

3.2 Similar by Vector

A second operation that is provided by *Gensim* is named **SimilarByVector**: the idea is that, once a vector (that results from a classical vector operation) is provided, the operation looks for the closest vector in the model.

- **Step 1: Computing the query vector.** A simple sum of vector is performed. However, since the initial vector is rooted in the origin of the axes, the resulting vector is also rooted in the origin of the axes. As an effect, Equation 3 is used to compute $\mathbf{v}_q$.
- **Step 2: Normalizing the query vector.** The query vector is again normalized to unit, i.e., $\overline{\mathbf{v}}_q = norm(\mathbf{v}_q)$ is used.
- **Step 3: Querying the vector space.** The normalized query vector is used to query the vector space for the closest vector. This time, all vectors are considered.

$$\begin{aligned} &\texttt{SimilarByVector}(P,N) = \\ &= \arg\max_{w \in V} \cos(\mathbf{v}_w, \overline{\mathbf{v}}_q) = \\ &= \arg\max_{w \in V} \frac{\mathbf{v}_w \cdot \overline{\mathbf{v}}_q}{\|\mathbf{v}_w\| \|\overline{\mathbf{v}}_q\|} \end{aligned} \tag{4}$$

Equation 4 is very similar to Equation 2: the only difference is that all the vectors are considered, while in Equation 2 input vectors are discarded.

It is worth noting that, due to unit normalization it could be possible to directly derive **SimilarByVector** by simply relaxing the constraint that input vectors must be excluded from the final results.

Example. By exploiting the same example analogy that was illustrated in the example reported in Section 3.1, Figure 3 illustrates the operation **SimilarByVector**.

Figure 3a depicts the computation of the query vector. Notice that the resulting query vector (fuchsia vector) is now longer than the fuchsia vector in Figure 2a, because each coordinate is now computed as pure sum of the corresponding input coordinates.

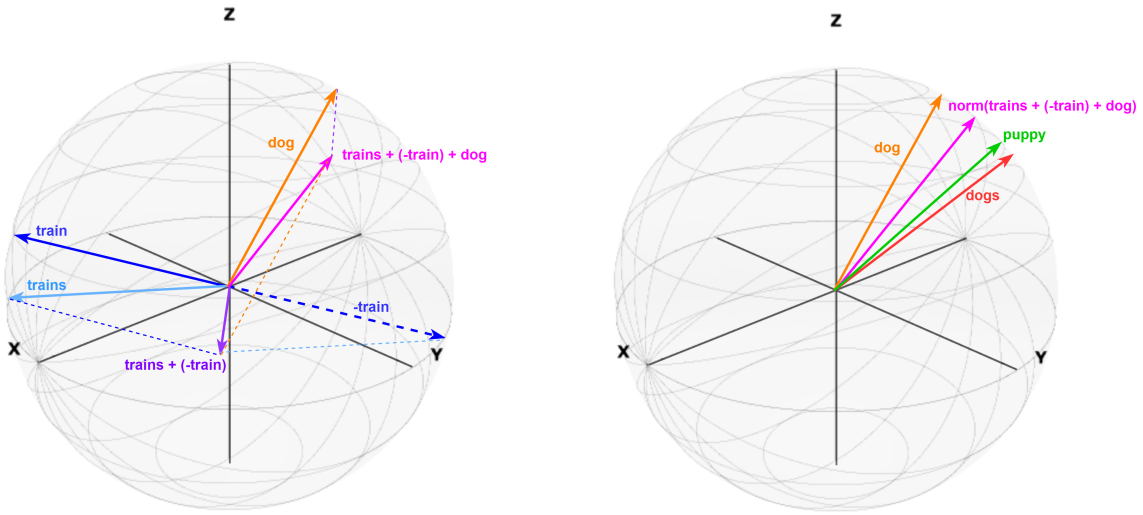
The fuchsia vector in Figure 3b is the corresponding normalized query vector, which is identical to the normalized query vector depicted in Figure 2b: thus, the nearby vectors are the same. However, this time the vector "dog" is obtained because it is the closest vector and is not discarded (as it is an input vector).

4 METHODOLOGY

This section presents the investigation methodology that was carried out to address the research question presented in Section 1.

4.1 Sources

Word Embedding based on the Skip-gram architecture with Negative Sampling was introduced by (Mikolov et al., 2013a), which demonstrated the effectiveness of such models on large text corpora: the model was



(a) Computing the query vectors.

(b) Searching for the most similar vector after normalization.

Figure 3: 3-D visual representation of **SimilarByVector**.

trained through the Google News dataset[1] and on the `Wikipedia Text8` corpus[2]; a 300-dimensional space was exploited for encoding words within the model.

The word-embedding model so far obtained was used for the seminal work (Mikolov et al., 2013b), so as to validate linguistic analogies; however, the word-embedding model is no longer publicly available at the link reported in the paper. Consequently, the present study utilizes a pre-trained model that should have similar characteristics (as far as training parameters and the Wikipedia training corpus are concerned), to remain as consistent as possible with (Mikolov et al., 2013b).
Specifically, it is the pre-trained *Word2vec* model, *GoogleNews-vectors-negative300-SLIM*.

This model constitutes an optimized and reduced-vocabulary version of the original *GoogleNews-vectors-negative300* released by Google. The original model was trained on approximately 100 billion words from the English-language Google News corpus, employing the Skip-gram architecture with Negative Sampling. The SLIM version restricts the vocabulary to the approximately 300,000 most frequent words and is not normalized to unit length by default[1].

So as to pursue the investigation, the *WordNet* lexical ontology (Miller, 1995) was downloaded[3]. In order to query the lexical ontology, the Python library *NLTK* was used[4].

Only words confirmed as valid English lemmas were kept, while very technical or ambiguous terms were removed.

4.2 Generating Word Pairs

Generation of a significant number of word pairs for a specific category of linguistic analogy is the necessary starting point. Indeed, the idea is to couple all the word pairs that share the same type of linguistic analogy.

Types of Linguistic Analogies. A pool of categories was considered to automatically build the corpus of word pairs. This is the only human intervention, i.e., choosing the categories of interest; words and word pairs were automatically built by querying the *WordNet* only, in a totally agnostic way. Hereafter, the considered categories are presented.

- **Capital Cities** (e.g., `Norway:Oslo`). This group consists of 94 pairs composed of a country name and its corresponding capital city, illustrating a semantic relationship.
- **Currency** (e.g., `Mexico:peso`). Composed of 61 pairs, each formed by a country and its official currency, representing a semantic relationship.
- **Man-Woman** (e.g., `actor:actress`). This set includes 57 pairs reflecting gender-specific counterparts in professions, roles, or familial relations. Some pairs are identical due to the existence of

[1]https://code.google.com/archive/p/word2vec/, accessed on 15/07/2025.

[2]http://mattmahoney.net/dc/textdata.html, accessed on 15/07/2025.

[3]https://wordnet.princeton.edu/, accessed on 15/07/2025.

[4]https://www.nltk.org/, accessed on 15/07/2025.

gender-neutral terms in English (e.g., `teacher`). This exemplifies a semantic relationship.

- **Opposites** (e.g., `hot:cold`). Containing 50 pairs of antonyms, all adjectives, illustrating syntactic opposition.
- **Comparatives** (e.g., `tall:taller`). This group consists of 90 pairs of adjectives in their base and comparative forms, demonstrating syntactic morphological relationships.
- **Verb Forms** (e.g., `work:works`). Consisting of 65 pairs illustrating verb forms in the first-person singular and third-person singular present tense, exemplifying syntactic inflection (improperly referred to as "plural verbs" in (Mikolov et al., 2013a)).
- **Plural Nouns** (e.g., `chair:chairs`). Containing 344 pairs of singular and plural forms of common nouns. Note that some nouns share identical singular and plural forms, such as `furniture` and `species`. This represents a syntactic morphological relationship.
- **Plural Animals** (e.g., `cat:cats`). A subset of plural nouns focusing exclusively on animal names, containing 120 pairs. Some animal names have identical singular and plural forms, such as *sheep* and *fish*. This is a syntactic morphological relationship.
- **Plurals**. The two previous groups "Plural Nouns" and "Plural Animals" are fused together in a single group.

During the experiments, each pair in a group was compared with all other pairs in the same group. This explains why three different groups about "plurals" were considered: "Plural Nouns" were compared to each other, "Plural Animals" were compared to each other, and finally all plurals were compared to each other (thus, comparing animals with nouns).

Generation Process. Previous corpora of linguistic analogies (see Section 2) were not considered, in that since they are mostly created by human beings, they can be incomplete and can incorporate cognitive bias. Clearly, semantic relationships that are not encoded in *WordNet* were not considered; however, those encoded in *WordNet* are universally recognized and are not subject to cognitive bias.
The goal was to automate, as much as possible, the generation of word pairs for specific categories. Consequently, once categories of interest were defined, a pool of verified sets of words from which to start with the gathering process was established; then, the generation of word pairs was performed automatically.

The *WordNet* lexical ontology was the basic source. However, *WordNet* does not include factual relations such as country–currency pairs and country–capital pairs; for this reason, supplementary resources were needed[5]. A Python library as *pycountry*[6], actually acquired the factual word pairs.
Morphological and grammatical transformations, including pluralization and verb conjugations, were applied using inflection libraries such as *inflect*[7], which handle both regular and irregular forms.

To keep the experimental runtime manageable, not all possible word pairs were explored. However, the selected pairs were chosen to be as unbiased and domain-agnostic as possible.

4.3 Experimental Campaign

The investigation methodology encompassed an experimental campaign. It was structured as follows.

- For each category C_k of word pairs, generated as described in Section 4.2, each pair $p_i \in C_k$ was coupled with another pair $p_j \in C_k$ (both p_i and p_j belonged to the same category); p_i and p_j could coincide.
- Given two pairs $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$, the resulting vector $\mathbf{v}_R$ was computed.

$$\mathbf{v}_R = \mathbf{v}_{y_j} - \mathbf{v}_{x_j} + \mathbf{v}_{x_i}$$

  Vector operations were performed either by using the operation **MostSimilar** or the operation **SimilarByVector**. The word R, represented by $\mathbf{v}_R$, was the retrieved word from the operation.
- If the word R coincided with y_i in the pair $p_i = (x_i, y_i)$, than the retrieval succeeded, otherwise it failed.

As anticipated, a first round of experiments was done by performing vector operations through **MostSimilar**, while a second round was performed through **SimilarByVector**. These two rounds were necessary to compare the effects of using one operation in place of the other.

[5] https://en.wikipedia.org/wiki/List_of_circulating_currencies,
https://en.wikipedia.org/wiki/List_of_national_capitals,
accessed on 15/07/2025.

[6] https://pypi.org/project/pycountry/

[7] https://pypi.org/project/inflect/,
accessed on 15/07/2025.

Table 1: Correctness (in percentage) of linguistic analogy detection using different vector operations (**MS** stands for **MostSimilar**, while **SBV** stands for **SimilarByVector**).

Categories	Sample Word Pair		MS	SBV
Capital Cities	Norway	Oslo	78.78%	28.60%
Currency	Mexico	peso	0.75%	0.13%
Man-Woman	actor	actress	39.30%	10.40%
Opposites	hot	cold	12.08%	2.00%
Comparatives	tall	taller	81.35%	11.42%
Verb Forms	work	works	71.44%	13.06%
Plural Nouns	chair	chairs	67.80%	5.21%
Plural Animals	cat	cats	83.14%	9.44%
Plurals	box	boxes	59.38%	4.98%

5 RESULTS

This section presents the results of the experiments conducted using the methodology described in Section 4, comparing the performance of **MostSimilar** and **SimilarByVector**.

5.1 General Overview

Table 1 shows the correctness, expressed as the percentage of correctly predicted target words over the total number of evaluated queries, obtained by using **MostSimilar** and **SimilarByVector**, considering individually each category defined in Section 4.2. The table highlights that the two operations behave very differently, with **MostSimilar** consistently producing higher scores than **SimilarByVector**. This difference can be attributed to the exclusion of input words from the pool of potential results that is performed by **MostSimilar**. In contrast, **SimilarByVector** considers all vectors as candidate, without excluding the input terms, leading to markedly-lower scores.

The performance of **MostSimilar** aligns with the benchmarks reported by (Mikolov et al., 2013a). This way, it is possible to claim that the experimental campaign was performed in a way that is consistent with the experiments presented in the seminal paper.

5.2 Visual Analysis

A visual analysis was conducted to perform an in-depth study of results. The heatmaps in Figure 4 presents the results of vector operations considering word pairs of the category *Plural Animals* described in Section 4.2: Figure 4a shows results for **MostSimilar**; Figure 4b shows results for **SimilarByVector**.

Each heatmap must be read as follows.

- **Vertical axis:** Each point in the vertical axis corresponds to a word pair in the category; more precisely, it corresponds to a pair $p_j = (x_j, y_j)$; the label on the axis is $y_j - x_j$.
- **Horizontal axis:** A point denotes a pair $p_i = (x_i, y_i)$ in the same category, but the label is only x_i.
- **Point (i, j)**. A point corresponds to the couple of word pairs $p_i = (x_i, y_i)$ and $p_j = (x_j, y_j)$. The point (or cell) denotes the operation

$$\mathbf{v}_R = \mathbf{v}_{y_j} - \mathbf{v}_{x_j} + \mathbf{v}_{x_i}$$

in such a way that the point is blue if R and y_i coincides, otherwise the point is red.

As an example, given `cats - cat` on the vertical axis and `dog` on the horizontal axis, the intersection point denotes whether

```
cats - cat + dog
```

actually returned `dogs` (blue cell) or a different word (red cell).

To improve interpretability, the columns are sorted by retrieval success rate, and the rows are sorted accordingly. Thus, cells along the diagonal represents vector operations such as

```
cats - cat + cat
```

Figure 4a shows results obtained using **MostSimilar**. Notice that the heat map is mostly blue, due to the high rate of correct results; indeed, this is a confirmation of the retrieval rate of 83.14% reported in Table 1 for the analyzed category of "Plural Animals".

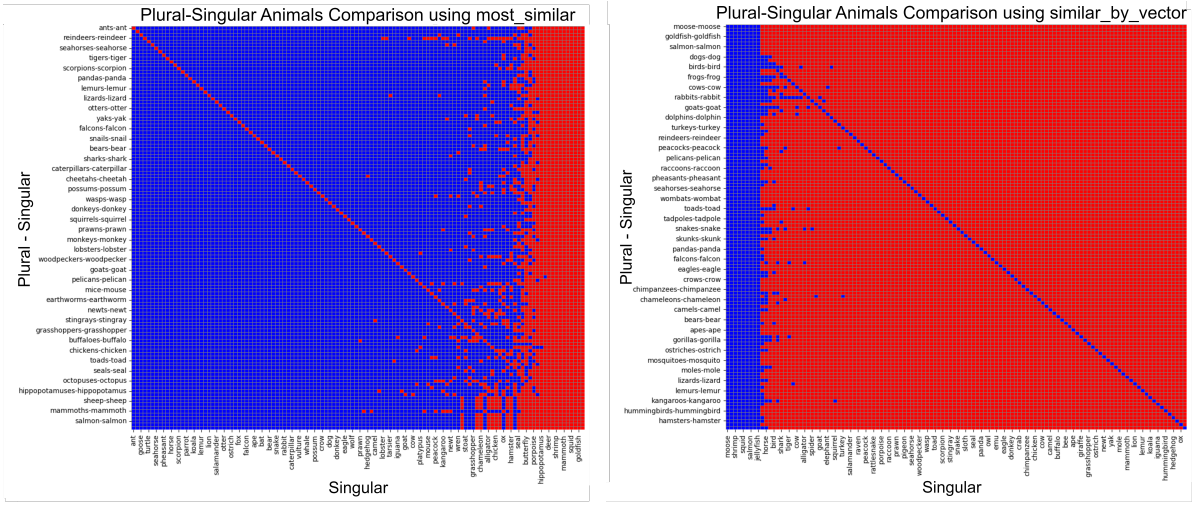
Nonetheless, notice that the cells in the diagonal are red: indeed, **MostSimilar** is not able to retrieve the starting word when a pair is coupled with itself; it should be

$$\mathbf{v}_{\texttt{cats}} - \mathbf{v}_{\texttt{cat}} + \mathbf{v}_{\texttt{cat}} = \mathbf{v}_{\texttt{cats}}$$

but **MostSimilar** does not return `cats`, because input words are excluded from the potential results. This is certainly an anomaly in the design of **MostSimilar**. Similar issues arise with words like `fish`, whose singular and plural forms are identical. Since these targets match the input and are filtered out, their retrieval is impossible with **MostSimilar**.

Figure 4b shows what happens when **SimilarByVector** is exploited. Recall that the main difference is that input vectors are not excluded; consequently, it is possible to expect that the cell on the diagonal are blue. This can be observed in the heatmap.

However, it performs worse overall, as denoted by the very large number of red cells. Indeed, this means



(a) Heat map obtained using the operation **MostSimilar**.

(b) Heat map obtained using the operation **SimilarByVector**.

Figure 4: Heat maps depicting the correctness obtained for the category "Plural Animals" of linguistic analogies.

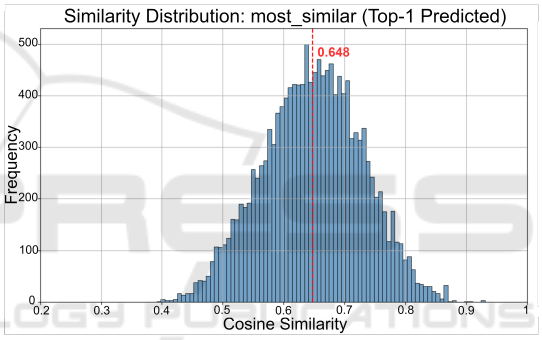
that without excluding the input vectors, **SimilarByVector** returns the input vectors as results, meaning that the input vectors are the closest to the query vector (see Section 3).

Thus, it is possible to interpret the results as follows: the linguistic category "Plurals Animal" is rather homogeneous, thus most vectors are in the same region of space; in particular, vectors representing singular and plural of the same word are close enough to each other, allowing **MostSimilar** to retrieve the correct one. However, they are not so close to the query vector, which in most cases remains close to the input vector(s), specifically $\mathbf{v}_{x_j}$. This is a clear clue that vector-offset operations do not manage linguistic analogies. The exclusion of input vectors performed by **MostSimilar** is just an attempt to achieve the desired result, not considering the intrinsic limitation of the approach, i.e., vector offsets cannot represent something (linguistic analogies or semantic relationships) that is outside the representational space (i.e., words).

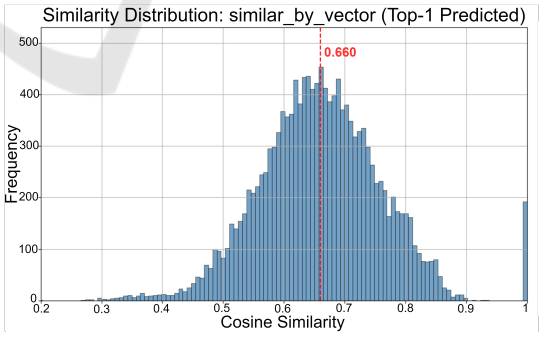
5.3 Distribution of Cosine Similarity

In order to further confirm the outcomes so far discussed, a third analysis was made. Specifically consider both definitions of **MostSimilar** and **SimilarByVector** in Section 3. Step 2 obtains the normalized query vector $\overline{\mathbf{v}}_q$, which is later used in Step 3 to look for the result vector $\overline{\mathbf{v}}_R$. During all the experiments, the cosine similarity between $\overline{\mathbf{v}}_q$ and $\overline{\mathbf{v}}_R$ was measured and collected.

Figure 5 depicts the collected cosine similarities. The charts were obtained as hereafter described.



(a) Cosine similarity distribution of the most similar predicted words using the **MostSimilar** method.



(b) Cosine similarity distribution of the most similar predicted words using the **SimilarByVector** method.

Figure 5: Target correctness matrices comparing singular and plural animal terms.

- Similarity scores were grouped in intervals of width 0.1.
- The horizontal axis reports the similarity scores.

- The vertical axis reports the number of pairs $(\overline{\mathbf{v}}_q, \overline{\mathbf{v}}_R)$ whose similarity score falls in the interval.

Figure 5a depicts the result for **MostSimilar**, while Figure 5b depicts the results for **SimilarByVector**.

For the **MostSimilar** method, it can be observed that the resulting vectors $\overline{\mathbf{v}}_R$ are not overly close to the query vectors $\overline{\mathbf{v}}_q$, as the average cosine similarity is 0.646, and in rare cases it is close to 0.9. This corresponds to an angle of about 49.8° between $\overline{\mathbf{v}}_q$ and $\overline{\mathbf{v}}_R$, meaning that they are somehow far away each other, they are not particularly close.

Figure 5b depicts the similarity scores for **SimilarByVector**. The reader can see that things do not change so much: the shape is slightly translated to the right, results in an average cosine similarity of 0.660. However, this still corresponds to an angle of approximately 48.5°, meaning that input vectors are just slightly close to $\overline{\mathbf{v}}_q$ so as to be retrieved as results, but they still remain distant. Furthermore, there are 195 similarity scores that are equal to 1, corresponding to the points seen in the diagonal of Figure 4b plus additional words whose singular and plural forms are identical.

Therefore, the very high correctness obtained by **MostSimilar** is apparent, in that it excludes the input vectors that, in most cases, are the closest ones to the query vector; consequently, it alters the search space and often retrieves the desired word by chance.

In contrast, **SimilarByVector**, is mathematically more adherent to the vector operations; as an effect, it gets the vector that is actually the closest to the query vector, which in most cases is not the wished one.

For the sake of space, the analysis is restricted to the category of "Plural Animals" only. Nonetheless, similar results were obtained for the other categories reported in Table 1. In particular, it is possible to say that the above considerations are further stressed when the correctness is lower. This confirms that the representational space of *Word Embedding* is not able to represent linguistic analogies and semantic relationships in the form of vector-offset operations; although the research tested only **MostSimilar** and **SimilarByVector**, it can be expected that this is true for any kind of vector-offset operation.

The fullresults of this experiment are available on a GitHub repository[8].

[8]GitHub repository: https://github.com/NLP-Studies-Group/NLP-Studies/

6 DISCUSSION

The results that were obtained by the investigation methodology ask for a discussion.

Vector operations are not suitable. The results show that linguistic analogies are not discovered by the tested vector operations. Both **MostSimilar** and **SimilarByVector** fail to obtain the expected results, even from a statistical point of view. In fact, assuming that they are somehow encoded within the word-embedding model, an operation that is actually able to deal with them should be very effective, with an correctness that is very close to 1 (considering imprecision in the training process). **MostSimilar** and **SimilarByVector** certainly are not the wanted operation.

Lack of Soundness. The idea that linguistic analogies and semantic relationships are encoded in the word-embedding model is certainly fascinating, however, all examined previous studies lack formal soundness. In other words, they move from the hypothesis that linguistic analogies are encoded, thus try to find out a way to let them emerge.
This approach is made evident by the exclusion of input vectors in **MostSimilar**: vectors that represent words that are somehow related to the same speech context point to the same region of space; consequently, since the difference between two very close vectors is a small vector (e.g., `trains - train`), the addition of a vector that points to a different region (e.g., `dog`) obtains a resulting vector that is necessarily very close to the last-added vector (e.g., `dog`); as an effect, the last-added vector is very likely to be the closest one to the normalized query vector.

In other words, it should not be expected that operations that are able to actually deal with semantic relationships (such as linguistic analogies) can be defined (as vector-offset operations): the reason is that there is no theoretical foundation that justifies the hypothesis; consequently, it is not reasonable to expect that they magically emerge.

Indeed, some research work tried to define a formal framework for word-embedding models and related vector-offset operations for detecting linguistic analogies, such as (Allen and Hospedales, 2019). However, the result is a sufficient condition that is likely not to be met; furthermore, all formulas are still probabilistic and do not consider the actual definitions of vector operations.

Universality of the relationships. Linguistic analogies are "universal semantic relationships": they express a concept that applies to all pairs of words for which it makes sense, independently of the position

of the vector pairs in the space. Differences of vectors are not able to capture this universality, because vector pairs point to different regions of space. Consider, for example, $\mathbf{v}_1$ =`trains - train` and $\mathbf{v}_2$ =`dogs - dog`: $\mathbf{v}_1$ and $\mathbf{v}_2$ certainly point to two very different directions, because the two pairs of input vectors certainly point to two different regions. However, both $\mathbf{v}_1$ and $\mathbf{v}_2$ should represent the concept "plural of", so they are expected to be very very similar. The conclusion is that it is not possible to expect that the universal concept of "plural of" could emerge by a vector operator.

Outside the representational space. Definitely, *Word Embedding* exploits a multi-dimensional space to represent words in such a way that "positional relationships" are encoded too, so as to be able to generate the $n+1$ most likely word, given a sequence of n words. Clearly, positional relationships depends on syntactical relationships (that strongly depends on the language); however, these syntactical relationships are implicitly learned as positional relationships. As an effect, if semantic relationships are somehow within positional relationships it is because words whose meanings are strongly related to a given topic are often in the same speech context, i.e., they often appear close by. As an effect, it is possible to assume that words related to similar speech contexts point to the same region.
Semantic relationships, such as linguistic analogies, are something different. The analogy "person with absolute power" does not correspond to any single word in the vocabulary; thus, there is no vector that represents it. It is true that words often represent concepts, but it is also true that many concepts are not represented by simple words.

Consequently, the concept "person with absolute power" (as a representative of all complex concepts) cannot be represented within the same representational space as a single word.

Effectiveness of *Word Embedding*. Nonetheless, *Word Embedding* has been an incredible step towards effective techniques for NLP. The reason is simple: it effectively captures "positional relationships", which in turn are indirectly related to syntactical and semantic relationships. This is actually the strength of *Word Embedding*.

7 CONCLUSIONS

The motivation for this paper was the lasting belief that word-embedding models actually encode linguistic analogies and structured semantic relationships.

Consequently, the previous works that investigated the existence of linguistic analogies within word-embedding models were resumed, to reconsider the problem from a novel perspective: the representational space and the way vector-offset operations are defined are the focus of the contribution.

Indeed, after the discussion reported in Section 6, it appears that vector-offset operations are not the proper tool to manage structured semantic relationships, because they work in the representational space of words, i.e., they treat semantic relationships as they were words; on the contrary, the training phase considers only positional relationships and implicitly encodes them within the representational space of a word-embedding model, by construction.
Definitely, the results and the considerations that are presented in this paper confirm what was suggested by the work (Drozd et al., 2016):, which clearly states that the essence of *Word Embedding* is as follows: "all current distributional semantic approaches rely on the same basic principle of using similarity between co-occurrence frequency distributions as a way to infer the strength of association between words". Furthermore, the same work claims that "For many practical purposes, such as information indexing and retrieval and semantic clustering, these approaches work remarkably well". Indeed, since 2013, techniques for *Word Embedding* have made possible an incredible step forward for NLP.

As a future work, the experiments will be extended to models obtained with different and more sophisticated techniques, such as *BERT*. Furthermore, the authors plan to continue the investigation on the theme. In particular, a good starting point could be to move from considerations that were argued by (Ethayarajh, 2019), i.e., considering transformations in the vector space, instead of vector-offset operations.

ACKNOWLEDGEMENTS

This study was funded by the European Union - *NextGenerationEU, in the framework of the GRINS - Growing Resilient, INclusive and Sustainable project (GRINS PE00000018 – CUP F83C22001720001)*. The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, nor can the European Union be held responsible for them.

REFERENCES

Allen, C. and Hospedales, T. (2019). Analogies explained: Towards understanding word embeddings. In *International Conference on Machine Learning*, pages 223–231. PMLR.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Drozd, A., Gladkova, A., and Matsuoka, S. (2016). Word embeddings, analogies, and machine learning: Beyond king-man+ woman= queen. In *Proceedings of coling 2016, the 26th international conference on computational linguistics: Technical papers*, pages 3519–3530.

Ethayarajh, K. (2019). Rotate king to get queen: Word relationships as orthogonal transformations in embedding space. *arXiv preprint arXiv:1909.00504*.

Finley, G., Farmer, S., and Pakhomov, S. (2017). What analogies reveal about word vectors and their compositionality. In *Proceedings of the 6th joint conference on lexical and computational semantics (* SEM 2017)*, pages 1–11.

Garneau, N., Hartmann, M., Sandholm, A., Ruder, S., Vulić, I., and Søgaard, A. (2021). Analogy training multilingual encoders. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35(14), pages 12884–12892.

Gladkova, A., Drozd, A., and Matsuoka, S. (2016). Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. In *Proceedings of the NAACL Student Research Workshop*, pages 8–15.

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

Jurgens, D., Mohammad, S., Turney, P., and Holyoak, K. (2012). Semeval-2012 task 2: Measuring degrees of relational similarity. In ** 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)*, pages 356–364.

Levy, O. and Goldberg, Y. (2014). Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*, pages 171–180.

Linzen, T. (2016). Issues in evaluating semantic spaces using word analogies. *arXiv preprint arXiv:1606.07736*.

Mandelbaum, A. and Shalev, A. (2016). Word embeddings and their use in sentence classification tasks. *arXiv preprint arXiv:1610.08229*.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013b). Linguistic regularities in continuous space word representations. In Vanderwende, L., Daumé III, H., and Kirchhoff, K., editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Rogers, A., Drozd, A., and Li, B. (2017). The (too many) problems of analogical reasoning with word vectors. In *Proceedings of the 6th joint conference on lexical and computational semantics (* SEM 2017)*, pages 135–148.

Salton, G., Wong, A., and Yang, C. S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.