Encoding syntactic dependencies using Random Indexing and Wikipedia as a corpus

Pierpaolo Basile and Annalina Caputo

Dept. of Computer Science, University of Bari "Aldo Moro"
Via Orabona, 4, I-70125, Bari (ITALY)
{basilepp,acaputo}@di.uniba.it

Abstract. Distributional approaches are based on a simple hypothesis: the meaning of a word can be inferred from its usage. The application of that idea to the vector space model makes possible the construction of a WordSpace in which words are represented by mathematical points in a geometric space. Similar words are represented close in this space and the definition of "word usage" depends on the definition of the context used to build the space, which can be the whole document, the sentence in which the word occurs, a fixed window of words, or a specific syntactic context. However, in its original formulation WordSpace can take into account only one definition of context at a time. We propose an approach based on vector permutation and Random Indexing to encode several syntactic contexts in a single WordSpace. We adopt WaCkypedia_EN corpus to build our WordSpace that is a 2009 dump of the English Wikipedia (about 800 million tokens) annotated with syntactic information provided by a full dependency parser. The effectiveness of our approach is evaluated using the GEometrical Models of natural language Semantics (GEMS) 2011 Shared Evaluation data.

1 Background and motivation

Distributional approaches usually rely on the WordSpace model [20]. An overview can be found in [18]. This model is based on a vector space in which points are used to represent semantic concepts, such as words.

The core idea behind WordSpace is that words and concepts are represented by points in a mathematical space, and this representation is learned from text in such a way that concepts with similar or related meanings are near to one another in that space (geometric metaphor of meaning). The semantic similarity between concepts can be represented as proximity in an n-dimensional space. Therefore, the main feature of the geometric metaphor of meaning is not that meanings can be represented as locations in a semantic space, but rather that similarity between word meanings can be expressed in spatial terms, as proximity in a high-dimensional space.

One of the great virtues of WordSpaces is that they make very few language-specific assumptions, since just tokenized text is needed to build semantic spaces. Even more important is their independence from the quality (and the quantity) of available training material, since they can be built by exploiting an entirely unsupervised distributional analysis of free text. Indeed, the basis of the WordSpace model is the *distributional hypothesis* [10], according to which the meaning of a word is determined by the set of textual *contexts* in which it appears. As a consequence, in distributional models words can be represented as vectors built over the observable *contexts*. This means that words are semantically related as much as they are represented by similar vectors. For example, if "basketball" and "tennis" occur frequently in the same context, say after "play", they are semantically related or similar according to the distributional hypothesis.

Since co-occurrence is defined with respect to a context, co-occurring words can be stored into matrices whose rows represent the terms and columns represent contexts. More specifically, each row corresponds to a vector representation of a word. The strength of the semantic association between words can be computed by using cosine similarity.

A weak point of distributional approaches is that they are able to encode only one definition of context at a time. The type of semantics represented in a WordSpace depends on the context. If we choose documents as context we obtain a semantics different from the one we would obtain by selecting sentences as context. Several approaches have investigated the aforementioned problem: [2] use a representation based on third-order tensors and provide a general framework for distributional semantics in which it is possible to represent several aspects of meaning using a single data structure. [19] adopt vector permutations as a means to encode order in WordSpace, as described in Section 2. BEAGLE [12] is a very well-known method to encode word order and context information in WordSpace. The drawback of BEAGLE is that it relies on a complex model to build vectors which is computational expensive. This problem is solved by [9] in which the authors propose an approach similar to BEAGLE, but using a method based on Circular Holographic Reduced Representations to compute vectors.

All these methods tackle the problem of representing word order in WordSpace, but they do not take into account syntactic context. A valuable attempt in this direction is described in [17]. In this work, the authors propose a method to build WordSpace using information about syntactic dependencies. In particular, they consider syntactic dependencies as context and assign different weights to each kind of dependency. Moreover, they take into account the distance between two words into the graph of dependencies. The results obtained by the authors support our hypothesis that syntactic information can be useful to produce effective WordSpace. Nonetheless, their methods are not able to directly encode syntactic dependencies into the space.

This work aims to provide a simple approach to encode syntactic relations dependencies directly into the WordSpace, dealing with both the scalability problem and the possibility to encode several context information. To achieve that goal, we developed a strategy based on Random Indexing and vector permutations. Moreover, this strategy opens new possibilities in the area of semantic composition as a result of the inherent capability of encoding relations between words.

The paper is structured as follows. Section 2 describes Random Indexing, the strategy for building our WordSpace, while details about the method used to encode syntactic dependencies are reported in Section 3. Section 4 describes a first attempt to define a model for semantic composition which relies on our WordSpace. Finally, the results of the evaluation performed using the GEMS 2011 Shared Evaluation data[1] is presented in Section 5, while conclusions are reported in Section 6.

2 Random Indexing

We exploit Random Indexing (RI), introduced by Kanerva [13], for creating a WordSpace. This technique allows us to build a WordSpace with no need for (either term-document or term-term) matrix factorization, because vectors are inferred by using an incremental strategy. Moreover, it allows to solve efficiently the problem of reducing dimensions, which is one of the key features used to uncover the "latent semantic dimensions" of a word distribution.

RI is based on the concept of Random Projection according to which high dimensional vectors chosen randomly are "nearly orthogonal".

Formally, given an $n \times m$ matrix A and an $m \times k$ matrix R made up of k m-dimensional random vectors, we define a new $n \times k$ matrix B as follows:

$$B^{n,k} = A^{n,m} \cdot R^{m,k} \quad k << m \tag{1}$$

The new matrix B has the property to preserve the distance between points. This property is known as Johnson-Lindenstrauss lemma: if the distance between two any points of A is d, then the distance d_r between the corresponding points in B will satisfy the property that $d_r = c \cdot d$. A proof of that property is reported in [8].

Specifically, RI creates a WordSpace in two steps (in this case we consider the document as context):

1. a context vector is assigned to each document. This vector is sparse, high-dimensional and ternary, which means that its elements can take values in {-1, 0, 1}. A context vector contains a small number of randomly distributed non-zero elements, and the structure of this vector follows the hypothesis behind the concept of Random Projection;
2. context vectors are accumulated by analyzing terms and documents in which terms occur. In particular, the semantic vector for a term is computed as the sum of the context vectors for the documents which contain that term. Context vectors are multiplied by term occurrences or other weighting functions, for example log-entropy.

Formally, given a collection of documents D whose vocabulary of terms is V (we denote with $dim(D)$ and $dim(V)$ the dimension of D and V, respectively) the above steps can be formalized as follows:

[1] Available on line:
http://sites.google.com/site/geometricalmodels/shared-evaluation

1. $\forall d_i \in D$, $i = 0, .., dim(D)$ we built the correspondent randomly generated context vector as:
$$\overrightarrow{r_j} = (r_{i1}, ..., r_{in}) \tag{2}$$
where $n \ll \dim(D)$, $r_{i*} \in \{-1, 0, 1\}$ and $\overrightarrow{r_j}$ contains only a small number of elements different from zero;
2. the WordSpace is made up of all term vectors $\overrightarrow{t_j}$ where:
$$\overrightarrow{t_j} = w_j \sum_{\substack{d_i \in D \\ t_j \in d_i}} \overrightarrow{r_i} \tag{3}$$
and w_j is the weight assigned to t_j in d_i.

By considering a fixed window W of terms as context, the WordSpace is built as follows:

1. a context vector is assigned to each term;
2. context vectors are accumulated by analyzing terms which co-occur in a window W. In particular, the semantic vector for each term is computed as the sum of the context vectors for terms which co-occur in W.

It is important to point out that the classical RI approach can handle only one context at a time, such as the whole document or the window W.

A method to add information about context (word order) in RI is proposed in [19]. The authors describe a strategy to encode word order in RI by permutation of coordinates in random vector. When the coordinates are shuffled using a random permutation, the resulting vector is nearly orthogonal to the original one. That operation corresponds to the generation of a new random vector. Moreover, by applying a predetermined mechanism to obtain random permutations, such as elements rotation, it is always possible to reconstruct the original vector using the reverse permutations. By exploiting this strategy it is possible to obtain different random vectors for each context[2] in which the term occurs. Let us consider the following example "The cat eats the mouse". To encode the word order for the word "cat" using a context window $W = 3$, we obtain:

$$\begin{aligned} < cat >= (\Pi^{-1} the) + (\Pi^{+1} eat) + \\ +(\Pi^{+2} the) + (\Pi^{+3} mouse) \end{aligned} \tag{4}$$

where $\Pi^n x$ indicates a rotation by n places of the elements in the vector x. Indeed, the rotation is performed by n right-shifting steps.

3 Encoding syntactic dependencies

Our idea is to encode syntactic dependencies, instead of words order, in the WordSpace using vector permutations.

[2] In the case in point the context corresponds to the word order

A syntactic dependency between two words is defined as:

$$dep(head, dependent) \tag{5}$$

where *dep* is the syntactic link which connects the *dependent* word to the *head* word. Generally speaking, *dependent* is the modifier, object or complement, while *head* plays a key role in determining the behavior of the link. For example, $subj(eat, cat)$ means that "cat" is the subject of "eat". In that case the *head* word is "eat", which plays the role of verb.

The key idea is to assign a permutation function to each kind of syntactic dependencies. Formally, let D be the set of all dependencies that we take into account. The function $f : D \rightarrow \Pi$ returns a schema of vector permutation for each $dep \in D$. Then, the method adopted to construct a semantic space that takes into account both syntactic dependencies and Random Indexing can be defined as follows:

1. a random context vector is assigned to each term, as described in Section 2 (Random Indexing);
2. random context vectors are accumulated by analyzing terms which are linked by a dependency. In particular the semantic vector for each term t_i is computed as the sum of the permuted context vectors for the terms t_j which are dependents of t_i and the inverse-permuted vectors for the terms t_j which are heads of t_i. The permutation is computed according to f. If $f(d) = \Pi^n$ the inverse-permutation is defined as $f^{-1}(d) = \Pi^{-n}$: the elements rotation is performed by n left-shifting steps.

Adding permuted vectors to the head word and inverse-permuted vectors to the corresponding dependent words allows to encode the information about both heads and dependents into the space. This approach is similar to the one investigated by [6] for encoding relations between medical terms.

To clarify, we provide an example. Given the following definition of f:

$$f(subj) = \Pi^{+3} \qquad f(obj) = \Pi^{+7} \tag{6}$$

and the sentence "The cat eats the mouse", we obtain the following dependencies:

$$\begin{array}{cc} det(the, cat) & subj(eat, cat) \\ obj(eat, mouse) & det(the, mouse) \end{array} \tag{7}$$

The semantic vector for each word is computed as:

– *eat*:
$$< eat >= (\Pi^{+3} cat) + (\Pi^{+7} mouse) \tag{8}$$
– *cat*:
$$< cat >= (\Pi^{-3} eat) \tag{9}$$
– *mouse*:
$$< mouse >= (\Pi^{-7} eat) \tag{10}$$

In the above examples, the function f does not consider the dependency *det*.

4 Compositional semantics

In this section we provide some initial ideas about semantic composition relying on our WordSpace. Distributional approaches represent words in isolation and they are typically used to compute similarities between words. They are not able to represent complex structures such as phrases or sentences. In some applications, such as Question Answering and Text Entailment, representing text by single words is not enough. These applications would benefit from the composition of words in more complex structures. The strength of our approach lies on the capability of codify syntactic relations between words overcoming the "word isolation" issue.

Recent work in compositional semantics argue that tensor product (⊗) could be useful to combine word vectors. In [21] some preliminary investigations about product and tensor product are provided, while an interesting work by Clark and Pulman [5] proposes an approach to combine symbolic and distributional models. The main idea is to use tensor product to combine these two aspects, but the authors do not describe a method to represent symbolic features, such as syntactic dependencies. Conversely, our approach deals with symbolic features by encoding syntactic information directly into the distributional model. The authors in [5] propose a strategy to represent a sentence like "man reads magazine" by tensor product:

$$man \otimes subj \otimes read \otimes obj \otimes magazine \tag{11}$$

They also propose a solid model for compositionality, but they do not provide a strategy to represent symbolic relations, such as *subj* and *obj*. Indeed, they state: "How to obtain vectors for the dependency relations - subj, obj, etc. - is an open question". We believe that our approach can tackle this problem by encoding the dependency directly in the space, because each semantic vector in our space contains information about syntactic roles.

The representation based on tensor product is useful to compute sentence similarity. For example, given the previous sentence and the following one: "woman browses newspaper", we want to compute the similarity between those two sentences. The sentence "woman browses newspaper", using the compositional model, is represented by:

$$woman \otimes subj \otimes browse \otimes obj \otimes newspaper \tag{12}$$

Finally, we can compute the similarity between the two sentences by inner product, as follows:

$$(man \otimes subj \otimes read \otimes obj \otimes magazine) \cdot (woman \otimes subj \otimes browse \otimes obj \otimes newspaper) \tag{13}$$

Computing the similarity requires to calculate the tensor product between each sentence element and then compute the inner product. This task is complex, but exploiting the following property of the tensor product:

$$(w_1 \otimes w_2) \cdot (w_3 \otimes w_4) = (w_1 \cdot w_3) \times (w_2 \cdot w_4) \tag{14}$$

the similarity between two sentences can be computed by taking into account the pairs in each dependency and multiplying the inner products as follows:

$$\begin{aligned} man \cdot woman \times read \cdot browse \times \\ \times magazine \cdot newspaper \end{aligned} \tag{15}$$

According to the property above mentioned, we can compute the similarity between sentences without using the tensor product. However, some open questions arise. This simple compositional strategy allows to compare sentences which have similar dependency trees. For example, the sentence “the dog bit the man” cannot can be compared to “the man was bitten by the dog”. This problem can be easily solved by identifying active and passive forms of a verb. When two sentences have different trees, Clark and Pulman [5] propose to adopt the *convolution kernel* [11]. This strategy identifies all the possible ways of decomposing the two trees, and sums up the similarities between all the pairwise decompositions. It is important to point out that, in a more recent work, Clark et al. [4] propose a model based on [5] combined with a compositional theory for grammatical types, known as Lambek’s pregroup semantics, which is able to take into account grammar structures. However, this strategy does not allow to encode grammatical roles into the WordSpace. This peculiarity makes our approach different. A more recent approach to distributional semantics and tree kernel can be found in [7] where authors propose a tree kernel that exploits distributional features to compute similarity between words.

5 Evaluation

The goal of the evaluation is to prove the capability of our approach in compositional semantics task exploiting the dataset proposed by Mitchell and Lapata [15], which is part of the “GEMS 2011 Shared Evaluation”. The dataset is a list of two pairs of adjective-noun/verb-object combinations or compound nouns. Humans rated pairs of combinations according to similarity. The dataset contains 5,833 rates which range from 1 to 7. Examples of pairs follow:

```
support offer help provide 7
old person right hand 1
```

where the similarity between offer-support and provide-help (verb-object) is higher than the one between old-person and right-hand (adjective-noun). As suggested by the authors, the goal of the evaluation is to compare the system performance against humans scores by Spearman correlation.

5.1 System setup

The system is implemented in Java and relies on some portions of code publicly available in the Semantic Vectors package [22]. For the evaluation of the system, we build our WordSpaces using the WaCkypedia_EN corpus[3].

[3] Available on line: http://wacky.sslmit.unibo.it/doku.php?id=corpora

Dependency	Description	Permutation
OBJ	object of verbs	Π^{+7}
SBJ	subject of verbs	Π^{+3}
NMOD	the relationship between a noun and its adjunct modifier	Π^{+11}
COORD	coordination	Π^{+23}

Table 1. The set of dependencies used in the evaluation.

WaCkypedia_EN is based on a 2009 dump of the English Wikipedia (about 800 million tokens) and includes information about: PoS, lemma and a full dependency parse performed by MaltParser [16].

Our approach involves some parameters. We set the random vector dimension to 4,000 and the number of non-zero elements in the random vector equal to 10. We restrict the WordSpace to the 500,000 most frequent words. Another parameter is the set of dependencies that we take into account. In this preliminary investigation we consider the four dependencies described in Table 1 which reports also the kind of permutation[4] applied to each dependency.

5.2 Results

In this section, we provide the results of semantic composition. Table 2 reports the Spearman correlation between the output of our system and the scores given by the humans. Table 2 shows results for each type of combination: verb-object, adjective-noun and compound nouns. Moreover, Table 2 shows the results obtained when two other corpora were used for building the WordSpace: ukWaC [1] and TASA.

ukWaC contains 2 billion words and is constructed from the Web by limiting the crawling to the .uk domain and using medium-frequency words from the BNC corpus as seeds. We use only a portion of ukWaC corpus consisting of 7,025,587 sentences (about 220,000 documents).

The TASA corpus contains a collection of English texts that is approximately equivalent to what an average college-level student has read in his/her lifetime. More details about results on ukWaC and TASA corpora are reported in an our previous work [3].

It is important to underline that syntactic dependencies in ukWaC and TASA are extracted using MINIPAR[5] [14] instead of the MaltParser adopted by WaCkypedia_EN.

The results show that WaCkypedia_EN provides a significant improvement with respect to TASA and ukWaC. This result is mainly due to two factors: (1) the WordSpace built using WaCkypedia_EN contains more words and dependencies; (2) MaltParser produces more accurate dependencies than MINIPAR. However, considering adjective-noun relation, TASA corpus obtains the best re-

[4] The number of rotations is randomly chosen.

[5] MINIPAR is available at http://webdocs.cs.ualberta.ca/∼lindek/minipar.htm

Corpus	Combination	ρ
WaCkypedia_EN	**verb-object**	**0.257**
	adjective-noun	**0.346**
	compound nouns	**0.254**
	overall	**0.299**
TASA	verb-object	0.160
	adjective-noun	0.435
	compound nouns	0.243
	overall	0.186
ukWaC	verb-object	0.190
	adjective-noun	0.303
	compound nouns	0.159
	overall	0.179

Table 2. GEMS 2011 Shared Evaluation results.

sult and generally all corpora obtain their best performance in this relation. Probably, it is easier to discriminate this kind of relation than others.

Another important point, is that TASA corpus provides better results than ukWaC in spite of the huger number of relations encoded in ukWaC. We believe that texts in ukWaC contain more noise because they are extracted from the Web.

As future research, we plan to conduct an experiment similar to the one proposed in [15], which is based on the same dataset used in our evaluation. The idea is to use the composition functions proposed by the authors in our WordSpace, and compare them with our compositional model. In order to perform a fair evaluation, our WordSpace should be built from the BNC corpus. Nevertheless, the obtained results seem to be encouraging and the strength of our approach relies on the capability of capturing syntactic relations in a semantic space. We believe that the real advantage of our approach, that is the possibility to represent several syntactic relations, leaves some room for exploration.

6 Conclusions

In this work, we propose an approach to encode syntactic dependencies in WordSpace using vector permutations and Random Indexing. WordSpace is built relying on WaCkypedia_EN corpus extracted from English Wikipedia pages which contains information about syntactic dependencies. Moreover, we propose an early attempt to use that space for semantic composition of short phrases.

The evaluation using the GEMS 2011 shared dataset provides encouraging results, but we believe that there are open points which deserve more investigation. In future work, we have planned a deeper evaluation of our WordSpace and a more formal study about semantic composition.

References

1. Baroni, M., Bernardini, S., Ferraresi, A., Zanchetta, E.: The WaCky Wide Web: A collection of very large linguistically processed Web-crawled corpora. Language Resources and Evaluation 43(3), 209–226 (2009)
2. Baroni, M., Lenci, A.: Distributional memory: A general framework for corpus-based semantics. Computational Linguistics 36(4), 673–721 (2010)
3. Basile, P., Caputo, A., Semeraro, G.: Encoding syntactic dependencies by vector permutation. In: Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics. pp. 43–51. Association for Computational Linguistics, Edinburgh, UK (July 2011)
4. Clark, S., Coecke, B., Sadrzadeh, M.: A compositional distributional model of meaning. In: Proceedings of the Second Quantum Interaction Symposium (QI-2008). pp. 133–140 (2008)
5. Clark, S., Pulman, S.: Combining symbolic and distributional models of meaning. In: Proceedings of the AAAI Spring Symposium on Quantum Interaction. pp. 52–55 (2007)
6. Cohen, T., Widdows, D., Schvaneveldt, R., Rindflesch, T.: Logical leaps and quantum connectives: Forging paths through predication space. In: AAAI-Fall 2010 Symposium on Quantum Informatics for Cognitive, Social, and Semantic Processes. pp. 11–13 (2010)
7. Croce, D., Moschitti, A., Basili, R.: Structured lexical similarity via convolution kernels on dependency trees. In: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing. pp. 1034–1046. Association for Computational Linguistics, Edinburgh, Scotland, UK. (July 2011)
8. Dasgupta, S., Gupta, A.: An elementary proof of the Johnson-Lindenstrauss lemma. Tech. rep., Technical Report TR-99-006, International Computer Science Institute, Berkeley, California, USA (1999)
9. De Vine, L., Bruza, P.: Semantic Oscillations: Encoding Context and Structure in Complex Valued Holographic Vectors. Quantum Informatics for Cognitive, Social, and Semantic Processes (QI 2010) (2010)
10. Harris, Z.: Mathematical Structures of Language. New York: Interscience (1968)
11. Haussler, D.: Convolution kernels on discrete structures. Technical Report UCSC-CRL-99-10 (1999)
12. Jones, M., Mewhort, D.: Representing word meaning and order information in a composite holographic lexicon. Psychological review 114(1), 1–37 (2007)
13. Kanerva, P.: Sparse Distributed Memory. MIT Press (1988)
14. Lin, D.: Dependency-based evaluation of MINIPAR. Treebanks: building and using parsed corpora (2003)
15. Mitchell, J., Lapata, M.: Composition in distributional models of semantics. Cognitive Science 34(8), 1388–1429 (2010)
16. Nivre, J., Hall, J., Nilsson, J.: Maltparser: A data-driven parser-generator for dependency parsing. In: Proceedings of LREC. vol. 6, pp. 2216–2219 (2006)
17. Padó, S., Lapata, M.: Dependency-based construction of semantic space models. Computational Linguistics 33(2), 161–199 (2007)
18. Sahlgren, M.: The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces. Ph.D. thesis, Stockholm: Stockholm University, Faculty of Humanities, Department of Linguistics (2006)

19. Sahlgren, M., Holst, A., Kanerva, P.: Permutations as a means to encode order in word space. In: Proceedings of the 30th Annual Meeting of the Cognitive Science Society (CogSci'08) (2008)
20. Schütze, H.: Word space. In: Hanson, S.J., Cowan, J.D., Giles, C.L. (eds.) Advances in Neural Information Processing Systems. pp. 895–902. Morgan Kaufmann Publishers (1993)
21. Widdows, D.: Semantic vector products: Some initial investigations. In: The Second AAAI Symposium on Quantum Interaction (2008)
22. Widdows, D., Ferraro, K.: Semantic Vectors: A Scalable Open Source Package and Online Technology Management Application. In: Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008) (2008)