Linear Compositional Distributional Semantics and Structural Kernels

Lorenzo Ferrone
University of Rome "Tor Vergata"
Via del Politecnico 1
00133 Roma, Italy
lorenzo.ferrone@gmail.com

Fabio Massimo Zanzotto
University of Rome "Tor Vergata"
Via del Politecnico 1
00133 Roma, Italy
fabio.massimo.zanzotto@uniroma2.it

Abstract

In this paper, we want to start the analysis of the models for compositional distributional semantics (CDS) with respect to the distributional similarity. We believe that this simple analysis of the properties of the similarity can help to better investigate new CDS models. We show that, looking at CDS models from this point of view, these models are strictly related with convolution kernels (Haussler, 1999), e.g.: tree kernels (Collins and Duffy, 2002). We will then examine how the distributed tree kernels (Zanzotto and Dell'Arciprete, 2012) are an interesting result to draw a stronger link between CDS models and convolution kernels.

1 Introduction

Distributional semantics (see (Turney and Pantel, 2010; Baroni and Lenci, 2010)) is an interesting way of "learning from corpora" meaning for words (Firth, 1957) and of comparing word meanings (Harris, 1964). A flourishing research area is compositional distributional semantics (CDS), which aims to leverage distributional semantics for accounting the meaning of word sequences and sentences (Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Zanzotto et al., 2010; Guevara, 2010; Grefenstette and Sadrzadeh, 2011; Clark et al., 2008; Socher et al., 2011). The area proposes compositional operations to derive the meaning of word sequences using the distributional meanings of the words in the sequences.

The first and more important feature of distributional semantics is to compare the meaning of different words, a way to compute their similarity. But, when focusing on compositional distributional semantics, methods are presented with respect to the compositional operation of the vectors. A scarce attention is given to how these operations affect the principal objective of the process of compositional distributional semantics: assessing the similarity between two word sequences. This analysis is important as the similarity is generally used even by machine learning models such as the kernel machines (Cristianini and Shawe-Taylor, 2000).

In this paper, we want to start the analysis of the models for compositional distributional semantics with respect to the similarity measure. We focus on linear CDS models. We believe that this simple analysis of the properties of the similarity can help to better investigate new CDS models. We show that, looking CDS models from this point of view, these models are strictly related with the convolution kernels (Haussler, 1999), e.g., the tree kernels (Collins and Duffy, 2002). We will then examine how the distributed tree kernels (Zanzotto and Dell'Arciprete, 2012) are an interesting result to draw a strongest link between CDS models and convolution kernels.

The rest of the paper is organized as follows. Section 2 focuses on the description of two basic binary operations for compositional distributional semantics, their recursive application to word sequences (or sentences) with a particular attention to their effect on the similarity measure. Section 3 describes the tree kernels (Collins and Duffy, 2002), the distributed tree kernels (Zanzotto and Dell'Arciprete, 2012), and the smoothed tree kernels (Mehdad et al., 2010; Croce et al., 2011) to introduce links with similarity measures applied over compositionally obtained distributional vectors. Section 4 draws sketches the future work.

2 Compositional distributional semantics over sentences

Generally, the proposal of a model for compositional distributional semantics stems from some basic vector combination operations and, then, these operations are recursively applied on the parse tree on the sequence of words of the sentences. In the rest of the section, we describe some simple basic operations along with their effects on the similarity between pairs of words and we describe some simple recursive models based on these operations. We finally describe how these simple operations and their recursive applications affect the similarity between sentences.

2.1 Two Basic Composition Operations

As we want to keep this analysis simple, we focus on two basic operations: the simple additive model, (presented in (Mitchell and Lapata, 2008) and cited as a comparative method in many research papers), and the full additive model (estimated in (Zanzotto et al., 2010; Guevara, 2010)).

We analyze these basic operations when resulting composed vectors are used to compute the similarity between two pairs of words. For simplicity, we use the dot product as the similarity measure. Let $a = a_1 a_2$ and $b = b_1 b_2$ be the two sequences of words and $\vec{a}_1$, $\vec{a}_2$, $\vec{b}_1$, and $\vec{b}_2$ be the related distributional vectors. Let $sim(a_1 a_2, b_1 b_2)$ be the similarity computed applying the dot product on the vectors $\vec{a}$ and $\vec{b}$ compositionally representing the distributional semantics of a and b.

The Basic Additive model (ADD) (introduced in (Mitchell and Lapata, 2008)) computes the distibutional semantics of a pair of words $a = a_1 a_2$ as:

$$ADD(a_1, a_2) = (1 - \alpha)\vec{a_1} + \alpha\vec{a_2}$$

where $0 < \alpha < 1$ weigths the first and the second word of the pair. Then, the similarity between two pairs of words is:

$$sim(a, b) = ADD(a_1, a_2) \cdot ADD(b_1, b_2) =$$
$$(1 - \alpha)^2 \vec{a}_1 \cdot \vec{b}_1 + (1 - \alpha)\alpha \vec{a}_1 \cdot \vec{b}_2 +$$
$$(1 - \alpha)\alpha \vec{a}_2 \cdot \vec{b}_1 + \alpha^2 \vec{a}_2 \cdot \vec{b}_2$$

that is, basically, the linear combination of the similarities $\vec{a}_i \cdot \vec{b}_j$ between the words composing the sequences. For example, the similarity between $sim($animal extracts, beef extracts$)$ takes into consideration the similarity between *animal* (of the first pair) and *extracts* (of the second pair) that can be totally irrelevant.

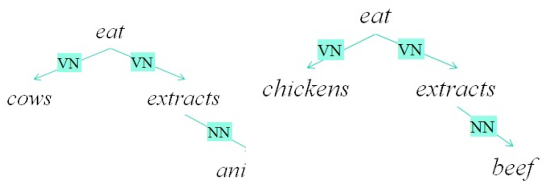

Figure 1: Two sentences and their simple dependency graphs

The Full Additive model (FADD) (used in (Guevara, 2010) for adjective-noun pairs and (Zanzotto et al., 2010) for three different syntactic relations) computes the compositional vector $\vec{a}$ of a pair using two linear tranformations A_R and B_R respectively applied to the vectors of the first and the second word. These matrices generally only depends on the syntactic relation R that links those two words. The operation follows:

$$FADD(a_1, a_2, R) = A_R \vec{a_1} + B_R \vec{a_2}$$

Then, the similariy between two pairs of words linked by the same relation R is:

$$sim(a, b) = FADD(a_1, a_2) \cdot FADD(b_1, b_2) =$$
$$= A_R \vec{a}_1 \cdot A_R \vec{b}_1 + A_R \vec{a}_1 \cdot B_R \vec{b}_2 +$$
$$B_R \vec{a}_2 \cdot A_R \vec{b}_1 + B_R \vec{a}_2 \cdot B_R \vec{b}_2$$

which is a linear combination of similarities between elements such as $A_R \vec{a}_i$, mixing a syntactic factor, the matrix A_R, and a single word of the sequence, a_i. In the above example, $sim($animal extracts, beef extracts$)$, (where we consider noun-noun (NN) as the syntactic relation) we also consider a factor (the similarity $B_{NN}\vec{extracts} \cdot A_{NN}\vec{beef}$) that may not be relevant in the similarity computation.

2.2 Recursive application on sentences and its effects on the similarity

The two linear models seems so simple that it is easy to think to recursively extend their application to whole sentences. To explain what is going on and what we are expecting, we will use two sentences as a driving example: *cows eat animal extracts* and *chickens eat beef extracts*. These are similar because both have ***animals*** *eating* ***animal*** *extracts*. This is what we expect the comparison between these two sentences should evaluate. Let

$$sim(T_a, T_b) = \begin{matrix} RFADD(T_a) \\ \cdot \\ RFADD(T_b) \end{matrix} = \begin{matrix} (2A_{VN}\vec{eat} + B_{VN}\vec{cows} + B_{VN}A_{NN}\vec{extracts} + B_{VN}B_{NN}\vec{animal}) \\ \cdot \\ (2A_{VN}\vec{eat} + B_{VN}\vec{chickens} + B_{VN}A_{NN}\vec{extracts} + B_{VN}B_{NN}\vec{beef}) \end{matrix}$$

Figure 2: Similarity using the recursive full additive model

$x = x_1 \dots x_n$ and $y = y_1 \dots y_m$ be two sentences (or word sequences).

2.2.1 Recursive basic additive model

The recursive basic additive model ($RADD$) is the first model we analyze. We can easily define the model as follows:

$$RADD(x_i x_{i+1} \dots x_n) = \\ = (1-\alpha)\vec{x}_i + \alpha RADD(x_{i+1} \dots x_n)$$

where $RADD(x_n) = \vec{x}_n$. Then, $RADD(x)$ is a weighted linear combination of the words in the sentence x, that is:

$$RADD(x) = \sum_{i=1}^{n} \lambda_i \vec{x_i}$$

where $\lambda_i = \alpha^{i-1}(1-\alpha)$ if $i < n$ and $\lambda_n = \alpha^{n-1}$ depends on α and the position of x_i in the sequence.

The similairity between two sentences x and y is then:

$$sim(x, y) = RADD(x) \cdot RADD(y) = \\ = \sum_{i=1}^{n}\sum_{j=1}^{m} \lambda_i \lambda_j \vec{x_i} \cdot \vec{y_j}$$

This is the weighted linear combination of the similariy among all the pairs of words taken from the sentences x and y. Given these two sample sentences, this similarity measure hardly captures the similarity in terms of the generalized sentence ***animals*** *eating* ***animal*** *extracts*. The measure also takes into consideration factors such as $\vec{chicken} \cdot \vec{beef}$ that have a high similarity score but that are not relevant for the similarity of the whole sentence.

2.2.2 Recursive Full Additive Model

For the recursive Full Additive Model, we need to introduce a structrured syntactic representation of the sentences. The full additive models (presented in Sec. 2.1) are defined on the syntactic dependency R between the words of the pair. We then use the dependency trees as syntactic representation. A dependency tree can be defined as a tree whose nodes are words and the typed links are the relations between two words. The root of the tree represents the word that governs the meaning of the sentence. A dependency tree T is then a word if it is a final node or it has a root r_T and links (r_T, Rel, C_i) where C_i is the i-th subtree of the node r_T and Rel is the relation that links the node r_T with C_i. The dependency trees of two example sentences are reported in Figure 1.

Stemming from the full additive models (FADD), the recursive FADD (RFADD) can be straightforwardly and recursively defined as follows:

$$RFADD(T) = \sum_i (A_{Rel}\vec{r_T} + B_{Rel} RFADD(C_i))$$

where (r_T, Rel, C_i) are the links originated in the root node r_T.

By recursively applying the model to the first sentence of the example (see Fig. 1), the resulting vector is:

$$RFADD(\text{cows eat animal extracts}) = \\ = A_{VN}\vec{eat} + B_{VN}\vec{cows} + A_{VN}\vec{eat} + \\ + B_{VN} RFADD(\text{animal extracts}) = \\ = A_{VN}\vec{eat} + B_{VN}\vec{cows} + A_{VN}\vec{eat} + \\ + B_{VN}A_{NN}\vec{extracts} + B_{VN}B_{NN}\vec{animal}$$

A first observation is that each term of the sum has a part that represents the structure and a part that represents the meaning, for example:

$$\overbrace{B_{VN}B_{NN}}^{structure} \underbrace{\vec{beef}}_{meaning}$$

It is possible to formally show that the function $RFADD(T)$ is a linear combination of elements $M_s\vec{w}_s$ where M_s is a product of matrices that represents the structure and $\vec{w}_s$ is the distributional meaning of one word in this structure, that is:

$$RFADD(T) = \sum_{s \in S(T)} M_s \vec{w}_s$$

where $S(T)$ are the relevant substructures of T. In this case, $S(T)$ contains the link chains.

Then, the similarity between two sentences in this case is:

$$sim(T_1, T_2) = RFADD(T_1) \cdot RFADD(T_2) = \\ = \sum_{s_1 \in S(T_1), s_2 \in S(T_2)} M_{s_1} \vec{w}_{s_1} \cdot M_{s_2} \vec{w}_{s_2}$$

The similarity between the two sentences T_a =*cows eat animal extracts* and T_b =*chickens eat beef extracts* in Figure 1 is represented in Figure 2. For the above dot product, $B_{VN}A_{NN}\vec{extracts} \cdot B_{VN}A_{NN}\vec{extracts} = 1$ as these addend represents the same piece of structure, $B_{VN}B_{NN}\vec{beef}) \cdot B_{VN}B_{NN}\vec{animal} < 1$ and should be strictly related to the value of $\vec{beef} \cdot \vec{animal}$ as these two parts are representing the branch of the tree describing the objects of the verb in the two sentences. The same should happen for $B_{VN}\vec{cows} \cdot B_{VN}\vec{chickens} < 1$. But, what do we expect for $B_{VN}\vec{cows} \cdot B_{VN}B_{NN}\vec{beef}$? We would like to have a similarity close to $\vec{beef} \cdot \vec{cows}$ or a similarity near 0, as these words appear in a different part of the structure? Going back to the overall goal of evaluating the similarity between the two sentences is clear that the second option should be preferred.

3 Tree Kernels

We here come to the last point we want to describe, the tree kernels (Collins and Duffy, 2002) and some strictly related recent results, the distributed tree kernels (Zanzotto and Dell'Arciprete, 2012) and the smoothed tree kernels (Mehdad et al., 2010; Croce et al., 2011). We want to show that what is computed by the $RADD$ and $RFADD$ is extremely similar to what is computed in tree kernels.

Tree kernels are defined (Collins and Duffy, 2002) as convolution kernels (Haussler, 1999), thus, they are generally defined recursively. But, given two trees T_1 and T_2, these kernels are defined as to compute the dot product between vectors $\Phi(T_1), \Phi(T_2)$, representing the trees in the feature space $\mathbb{R}^n$. Each dimensions (or features) of this huge space $\mathbb{R}^n$ is a relevant subtree t and we can consider that each relevant subtree t has an associated vector $\vec{t}$. The vector $\vec{t}$ is a vector of the orthonormal basis of the space $\mathbb{R}^n$. Then, the function Φ, that maps trees into the space $\mathbb{R}^n$, can be defined as follows:

$$\Phi(T) = \sum_{t \in S(T)} \omega_t \vec{t}$$

where ω_t is a weight assigned to t in the tree T. The tree kernel functions $TK(T_1, T_2)$ then basically computes:

$$TK(T_1, T_2) = \sum_{t_1 \in S(T_1), t_2 \in S(T_2)} \omega_{t_1} \omega_{t_2} \vec{t_1} \cdot \vec{t_2}$$

where $\vec{t_1} \cdot \vec{t_2}$ is the Kronecker's delta between t_1 and t_2, that is: $\vec{t_1} \cdot \vec{t_2} = \delta(t_1, t_2)$.

The equation above is incredibly similar to equation 1 that computes the similarity with the recursive full additive model. There are however two limitations in using tree kernels to encode compositional distributional semantics model: first, standard tree kernels only encode the structure; second, standard tree kernels work in $\mathbb{R}^n$ where n is huge making it infeasible to use such a huge vectors. For the first issue, an interesting line of research are the smoothed tree kernels (Mehdad et al., 2010; Croce et al., 2011) that exploits distributional vectors to compute the similarity among nodes of the trees that contain words. For the second issue an interesting recent result is the distributed tree kernel (Zanzotto and Dell'Arciprete, 2012) that approximates tree kernels by encoding the huge space $\mathbb{R}^n$ in a smaller space $\mathbb{R}^d$, with $d << n$. This allows to encode structural information into small vectors.

4 Conclusions

This paper presents some simple observations on one of the current approaches to compositional distributional semantics, drawing the link with the deeply studied tree kernels and convolution kernels. With this analysis, we aim to show that these approaches are not radically different. Instead, (linear) compositional distributional models can be rephrased as a compact version of some existing convolution kernels.

This paper is not conclusive as it leave open two avenues: first, we need to prove that distributed tree kernel (Zanzotto and Dell'Arciprete, 2012) can also encode distributional informations as described in the smoothed tree kernels (Mehdad et al., 2010; Croce et al., 2011); second, it still leaves unexplored how the similarity between sentences is affected by the other compositional distributional models (Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Zanzotto et al., 2010; Guevara, 2010; Grefenstette and Sadrzadeh, 2011; Clark et al., 2008; Socher et al., 2011).

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Comput. Linguist.*, 36(4):673–721, December.

Marco Baroni and Roberto Zamparelli. 2010. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1183–1193, Cambridge, MA, October. Association for Computational Linguistics.

Stephen Clark, Bob Coecke, and Mehrnoosh Sadrzadeh. 2008. A compositional distributional model of meaning. *Proceedings of the Second Symposium on Quantum Interaction (QI-2008)*, pages 133–140.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL02*.

Nello Cristianini and John Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, March.

Danilo Croce, Alessandro Moschitti, and Roberto Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1034–1046, Stroudsburg, PA, USA. Association for Computational Linguistics.

John R. Firth. 1957. *Papers in Linguistics.* Oxford University Press., London.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1394–1404, Stroudsburg, PA, USA. Association for Computational Linguistics.

Emiliano Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, pages 33–37, Uppsala, Sweden, July. Association for Computational Linguistics.

Zellig Harris. 1964. Distributional structure. In Jerrold J. Katz and Jerry A. Fodor, editors, *The Philosophy of Linguistics*. Oxford University Press, New York.

David Haussler. 1999. Convolution kernels on discrete structures. Technical report, University of California at Santa Cruz.

Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. 2010. Syntactic/semantic structures for textual entailment recognition. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1020–1028, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June. Association for Computational Linguistics.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in Neural Information Processing Systems 24*.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res. (JAIR)*, 37:141–188.

F.M. Zanzotto and L. Dell'Arciprete. 2012. Distributed tree kernels. In *Proceedings of International Conference on Machine Learning*, pages 193–200.

Fabio Massimo Zanzotto, Ioannis Korkontzelos, Francesca Fallucchi, and Suresh Manandhar. 2010. Estimating linear models for compositional distributional semantics. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING)*, August,.