Towards Compositional Tree Kernels

Paolo Annesi, Danilo Croce, Roberto Basili
Department of Enterprise Engineering
University of Roma, Tor Vergata
00133 Roma, Italy
{annesi,croce,basili}@info.uniroma2.it

Abstract

Distributional Compositional Semantics (DCS) methods combine lexical vectors according to algebraic operators or functions to model the meaning of complex linguistic phrases. On the other hand, several textual inference tasks rely on supervised kernel-based learning, whereas Tree Kernels (TK) have been shown suitable to the modeling of syntactic and semantic similarity between linguistic instances. While the modeling of DCS for complex phrases is still an open research issue, TKs do not account for compositionality. In this paper, a novel kernel called Compositionally Smoothed Partial Tree Kernel is proposed integrating DCS operators into the TK estimation. Empirical results over Semantic Text Similarity and Question Classification tasks show the contribution of semantic compositions with respect to traditional TKs.

1 Introduction

Since the introduction of Landauer and Dumais in (Landauer and Dumais, 1997) and Schutze in (Schütze, 1998), Distributional Semantic Models (DMSs) have been an active area of research in computational linguistics and a promising technique for solving the lexical acquisition bottleneck by unsupervised learning. However, it is very difficult to reconcile these techniques with existing theories of meaning in language, which revolve around logical and ontological representations. According to logical theories (Kamp and Reyle, 1993; Blackburn and Bos, 2005), sentences should be translated to a logical form that can be interpreted as a description of the state of the world. On the contrary, vector-based techniques are closer to the philosophy of "meaning as context", relying on the Wittgenstein's (1953) "*meaning just is use*" and Firth's "*you shall know a word by the company it keeps*" and the distributional hypothesis of Harris (1968), that *words will occur in similar contexts if and only if they have similar meanings*. In these years attention has been focused on the question of how to combine word representations in order to characterize a model for sentence semantics. Since these models are typically directed at the representation of isolated words, a well formed theory on how to combine vectors and to represent complex phrases still represents a research topic. Distributional Compositional Semantic (DCS) models capture bi-gram semantics, but they are not sensitive to the syntactic structure yet. On the other hand, Convolution Kernels (Haussler, 1999) are well-known similarity functions among such complex structures. In particular, Tree Kernels (TKs) introduced in (Collins and Duffy, 2001), are largely used in NLP for their ability in capturing text grammatical information, directly from syntactic parse trees.

In this paper, we investigate the combination of DCS and Convolution Kernels. We extend a kernel function recently proposed in (Croce et al., 2011), called Smoothed Partial Tree Kernel (SPTK), that enriches the similarity between tree structures with a function of node similarity. As words are leaves in constituency syntactic trees, the lexical semantic similarity can be easily evaluated in term of similarity between their vector counterparts. In our DCS perspective, this lexical semantic information will be distributed across all the parse tree, as a carrier of the lexical composition, e.g. head/modifier relations, already explicit in dependency formalisms. The idea here is to propagate lexical semantic information over the entire parse tree, by building a Compositionally enriched Constituency Tree (CCT). By making non-terminal nodes dependent on both syntactic (e.g. the `VP` grammatical category) and lexical semantic information, it is possible to formulate a new kernel function based on this tree representation, that takes into account for each node a distributional compositional metrics. Thus, the idea is to i) use the SPTK formulation in order to exploit the lexical information of the leaves, ii) define a procedure to mark nodes of a constituency parse tree that allow to spread lexical bigrams across the non-terminal nodes, iii) apply smoothing metrics sensible to the compositionality between the non-terminal labels. The resulting model has been called Compositionally Smoothed Partial Tree Kernel (CSPTK).

In Section 2, a summary of approaches for DCS and TKs is presented. The entire process of marking parse trees is described in Section 3. Therefore in Section 4 the CSPTK similarity function is presented. Finally, in Section 5, the CSPTK model is investigated in Semantic Text Similarity (STS) and Question Classification tasks.

2 Related Work

Distributional Compositional Semantics. Vector-based models typically represent isolated words and ignore grammatical structure (Turney and Pantel, 2010). They have thus a limited capability to model compositional operations over phrases and sentences. In order to overcome these limitations, Distributional Compositional Semantics (DCS) models have been investigated. In (Smolensky, 1990) compositionality of two vector $\vec{u}$ and $\vec{v}$ is accounted by the tensor product $\vec{u} \otimes \vec{v}$, while in (Foltz et al., 1998) lexical vectors are summed, keeping the resulting vector with the same dimension of the input ones. In (Mitchell and Lapata, 2008) two general classes of compositional models have been defined: a linear additive model $\vec{p} = \mathbf{A}\vec{u} + \mathbf{B}\vec{v}$ and a multiplicative model $\vec{p} = \mathbf{C}\vec{u}\vec{v}$. $\mathbf{A}$ and $\mathbf{B}$ are weight matrices and $\mathbf{C}$ is a weight tensor that project lexical vectors $\vec{u}$ and $\vec{v}$ onto the space of $\vec{p}$, i.e. the vector resulting from the composition.

These models usually assume that composition is a symmetric function of the constituents. While this might be reasonable for certain structures, such as lists, a model of composition based on syntactic structure requires some way of differentiating the contributions of each constituent. In (Erk and Pado, 2008), the concept of a *structured vector space* is introduced, where each word is associated to a set of vectors corresponding to different syntactic dependencies. Noun component of a composition between verb and noun is here given by an average of verbs that the noun is typically object of. In (Guevara, 2010) a regressor is trained for adjective-noun (AN) compositionality: pairs of adjective-noun vector concatenations are used as input in training data, whilst corpus-derived AN vectors as output. A similar approach was previously undertaken by (Zanzotto et al., 2010).

A specific linear model of semantic composition based on the idea of space projection is proposed in (Annesi et al., 2012) for simple grammatical structures, i.e. syntactically typed bi-grams. Given a phrase such as "*buy car*" they project the source vectors $\vec{buy}$ and $\vec{car}$, into a so-called Support Subspace, that is a subset of the original feature space. Space Projection depends on both the two involved lexicals and selects only their "common" features: these *concurrently* constraint the suitable lexical interpretation *local* to the phrase.

Given two phases p_1 and p_2, semantic similarity can be computed by first projecting the two pairs in the suitable Support Subspace and then applying the traditional cosine metrics. Projection is expressed by a (filter) diagonal matrix $\mathbf{M}$ that projects each word into a subset of the original features. Different projections are discussed in (Annesi et al., 2012) aimed at identifying suitable semantic aspects of the underlying head/modifier relationships. The compositional similarity judgment between phrases $p_1 = (u, v)$ and $p_2 = (u', v')$ over the support subspace of p_1 is thus expressed as:

$$\Phi^{(\circ)}(p_1, p_2) = (\mathbf{M}\vec{u} \cdot \mathbf{M}\vec{u'}) \circ (\mathbf{M}\vec{v} \cdot \mathbf{M}\vec{v'}) \quad (1)$$

where first cosine similarity $(\cdot)$ between the vectors projected in the selected support subspaces is computed and then a composition function $\circ$, such as the sum or the product, is applied. Notice how projection $\mathbf{M}$ may depend on the involved pairs in complex ways. A Support Subspace can be derived from just one pair p_i and then being imposed to the other with a corresponding asymmetric behavior of the Φ metrics, denoted by Φ_i. Alternatively, $\mathbf{M}$ can be derived from projecting in two Support Subspaces, as derived for the two pairs, and then combining them by making again Φ symmetric. The symmetric composition function is thus obtained as the combination:

$$\Phi_{12}^{(\diamond)}(p_1, p_2) = \Phi_1^{(\circ)}(p_1, p_2) \diamond \Phi_2^{(\circ)}(p_1, p_2) \quad (2)$$

where Φ_1, as well as Φ_2, projects both p_1 and p_2

into the Support Subspace of p_1 (and p_2, respectively), and Φ_i are then combined via the $\diamond$ operator (e.g. sum). Although Support Subspaces cannot be applied to estimate similarity between complex linguistic structures, they seem very effective for simple syntactic structures. In (Annesi et al., 2012) experiments over different variants of Eq. 1 and 2, i.e. different choices for projections $\mathbf{M}$ and compositions $\circ$ and $\diamond$, are there discussed. Best results are obtained within the dataset introduced in (Mitchell and Lapata, 2010) when a multiplicative operator $\circ$ is used in Eq. 1.

Recently, Compositional Semantics has been used in syntactic parsing, as shown in (Socher et al., 2013) where Compositional Vector Grammars (CVGs) have been defined to extend small-state Probabilistic Context-Free Grammars, introducing distributional semantic constraints in constituency parsing: interestingly, CVGs allows to estimate the plausibility of the corresponding syntactic constituent within a Recursive Neural Network, by assigning scores to nodes in the parse tree. A similar integrated contribution of lexical information (i.e. word vectors) and syntactic constituency is proposed in semantic extensions of TKs, as introduced in (Croce et al., 2011). As they offer a framework to define similarity metrics strongly tied to the syntactic structure of entire sentences, they will be hereafter discussed.

Tree Kernels. Kernels are representationally efficient ways to encode similarity metrics able to support complex textual inferences (e.g. semantic role classification) in supervised learning models. Tree Kernels (TK) as they have been early introduced by (Collins and Duffy, 2001) correspond to Convolution Kernel (Haussler, 1999) over syntactic parse trees of sentence pairs. A TK computes the number of substructures (as well as their partial fragments) shared by two parse trees T_1 and T_2. For this purpose, let the set $\mathcal{F} = \{f_1, f_2, \dots, f_{|\mathcal{F}|}\}$ be a space of tree fragments and $\chi_i(n)$ be an indicator function: it is 1 if the target f_i is rooted at node n and 0 otherwise. A tree-kernel function is a function $TK(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2)$, where N_{T_1} and N_{T_2} are the sets of the T_1's and T_2's nodes, respectively and $\Delta(n_1, n_2) = \sum_{i=1}^{|\mathcal{F}|} \chi_i(n_1)\chi_i(n_2)$. The Δ function recursively compute the amount of similarity due to the similarity among substructures. The type of fragments allowed determine the expressiveness of the kernel space and different tree kernels are characterized by different choices. Lexical information has been early neglected in recursive matching, so that only exact matching between node labels were given a weight higher than 0, (Collins and Duffy, 2001): even when leaves are involved they must be equal, so that no lexical generalization was considered. An effective modeling of lexical information is proposed by (Croce et al., 2011), in the so called Smoothed Partial Tree Kernel (SPTK). In SPTK, the TK extends the similarity between tree structures allowing a smoothed function of node similarity. The aim of SPTK is to measure the similarity between syntactic tree structures, which are semantically related, i.e. partially similar, even when nodes, e.g. words at the leaves, differ. This is achieved with the following formulation of the function Δ over nodes $n_i \in T_i$:

$$\begin{aligned}
&\Delta_\sigma(n_1, n_2) = \mu\lambda\sigma(n_1, n_2), \text{ where } n_1 \text{ and } n_2 \\
&\qquad\qquad\qquad\qquad \text{are leaves, else} \\
&\Delta_\sigma(n_1, n_2) = \mu\sigma(n_1, n_2)\Big(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1) = l(\vec{I}_2)} \\
&\qquad \lambda^{d(\vec{I}_1) + d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta_\sigma(c_{n_1}(\vec{I}_{1j}), c_{n_2}(\vec{I}_{2j}))\Big)
\end{aligned} \tag{3}$$

In Eq. 4, $\vec{I}_{1j}$ represents the sequence of subtrees, dominated by node n_1, that are shared with the children of n_2 (i.e. $\vec{I}_{2j}$): as all other non-matching substructures are neglected. Parameter λ accounts for the decay factor penalizing embedded trees, whose contribution affects too many dominating structures towards the root. Moreover, σ is a similarity between two nodes: for non terminals it can be strict, such as the dot-product imposed to word vectors at the leaves. More details about SPTK as well as its efficient computation are discussed in (Croce et al., 2011). In constituency parse trees, the lexical similarity is only applied between leaves, which reflect words. One main limitation of SPTK is that lexical similarity does not consider compositional interaction between words. Given the following phrase pairs (*np* (*nn river*)(*nn bank*)) and (*np* (*nn savings*) (*nn bank*)), the SPTK estimates the similarity between *bank* without considering that they are compositionally modified with respect different meanings. Hereafter, the DCS operator of Eq. 1 and 2 will be adopted to model semantic similarity at the nodes in a parse tree, in general seen as head/modifier syntactic pairs. Notice that this is the role of the

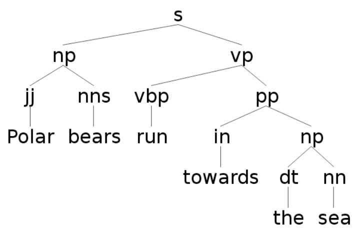

Figure 1: Constituency tree of the sentence "*Polar bears run towards the sea*"

function σ in Eq. 4.

3 Explicit compositions in Parse Tree

In order to consider compositional semantic constraints during a Tree Kernel computation, parse trees are here enriched to enable the definition of a Compositionally Smoothed Partial Tree Kernel, i.e. CSPTK. The input structures are thus tree pairs whose nodes are enriched with lexical information needed for the recursive compositional matching foreseen by the adopted convolution model. The syntactic structure of an individual sentence s is the constituency-based parse tree, as shown in Figure 1. Nodes can be partitioned into: *terminal nodes* n ($\mathcal{T}$), i.e. leaves representing lexical information, in terms of $\langle l_n$*::*$pos_n\rangle$, such as *polar::j* or *bear::n*, where *l* is the lemma of the token and pos its part-of-speech[1]; *Pre-terminal nodes* ($\mathcal{PT}$) are the direct ancestors of terminals and they are marked through the pos of their unique corresponding leaf; Non Pre-terminal nodes ($\mathcal{NPT}$), i.e. nodes that are neither terminal nor pre-terminal and reflect the phrase type, e.g. nominal phrase (*np*) or verb phrase (*vp*). Notice that all nodes in a tree express either lexical information (e.g. terminal $n \in \mathcal{T}$) or the compositional information between one head and a modifier corresponding to subtrees, such as the non pre-terminal in $\mathcal{NPT}$. In order to model this information, we need to associate each node with different types of information able to express every aspect of compositionality, i.e. lexical as well as grammatical properties. We model this information in a form of a complex mark-up of generic non pre-terminal nodes $n \in \mathcal{NPT}$, in order to exploit them in a compositional extension of a tree kernel (such as in Eq. 4). Compositionality operators acting on the subtrees depend on at least the following types of syntactic as well as lexical information:

[1]General POS tags are obtained from the PennTreebank standard by truncating at their first char (as in $bear :: n$).

Grammatical Types, denoted by $\mathcal{GT}$, that express the grammatical category of the constituent corresponding to the root of a subtree. Example of these types are the np or vp traditional categories of context-free grammars.

Lexical Information. Non pre-terminal nodes in general express binary grammatical relations between a varying number of dominated subtrees (i.e. direct descendants). Each node can be expressed in terms of an head/modifier pair, denoted by (h, m). In order to emphasize the semantic contribution that a subtree (compositionally) expresses, the lexical information about the involved head (l_h) and modifier (l_m) lexicals must be expressed: we denote this information through the 4-tuple $\langle \text{l}_h\text{::pos}_h,\text{l}_m\text{::pos}_m\rangle$. Notice that this information can be used as an index to a distributional semantic model where lexical entries are expressed by unique vectors for individual lemma and POS tag pairs.

Syntactic Relations. Usually each node expresses a specific syntactical relation between the head and its modifier. Depending on linguistic theories several system of types have been proposed. As in this work, syntactic relations are only used to constrain structural analogies between two trees, the reference relationship system adopted is not here discussed. We denote the set of syntactic relations $\mathcal{SR}$, and they are usually derived by simply juxtaposing grammatical labels of the involved head and modifier, h and m, subtrees. Every relation is denoted by $rel_{h,m} \in \mathcal{SR}$. Examples of the adopted syntactic relations are: vp/np for verb object relation or nn/nn for noun compounds.

Therefore, according to the definitions above, every non pre-terminal $n \in \mathcal{NPT}$ is marked with the following triple

$$\langle gT, rel_{h,m}, \langle l_h :: pos_h, l_m :: pos_m \rangle \rangle$$

where $gT \in \mathcal{GT}, rel_{h,m} \in \mathcal{SR}$, and l_i and pos_i are lexical entries, and POS tags. This triple enables the definition of a similarity function between sentence pairs through the recursive comparison of the marked subtrees. Given the recursive nature of a Convolution Kernel, we will show how the similarity estimation of two (sub)trees is made dependent on the semantic equivalence between the triples assigned to their roots.

A shallow compositional function, that ignores any syntactic information of gT and $rel_{h,m}$ for the head/modifier structure (h, m), can be straightforwardly defined by adopting the DCS model discussed in Section 2, and in particular Eq. 2. Given two subtrees in T_1, T_2, rooted at n_1, n_2, the corresponding head-modifier pairs $(h_1, m_1), (h_2, m_2)$ are defined. This similarity metrics, based on the geometric projection into Support Subspace (Eq. 2), can be applied as follows:

$$\sigma_{Comp}\big((h_1, m_1), (h_2, m_2)\big) = \Phi_{12}^{(\diamond)}((h_1, m_1), (h_2, m_2)) \tag{4}$$

In particular, σ_{Comp} is evaluated through the *Symmetric model* introduced in (Annesi et al., 2012). This model is characterized by a projection $\mathbf{M}$ that selects the 50 dimensions of the space that maximize the component-wise product between compounds, and by the operator *diamond* that combines the similarity scores with the product function (Eq. 2). Moreover, similarity scores in each subspace are obtained by defining $\circ$ as the sum of cosine similarities in Eq. 1.

3.1 Mark-Up Rules for Constituency Trees

While marking terminal $\mathcal{T}$ nodes and pre-terminal $\mathcal{PT}$ nodes is quite simple, the labeling of non pre-terminal $\mathcal{NPT}$ nodes is complex. Grammar specific notations and rules are needed and mainly differ with respect to (1) the type of the children nodes, i.e. if they are all pre-terminal or not, and (2) the arity of the branching at the root of a subtree: n-ary , with $n > 2$, can be found indeed.

$\mathcal{NPT}$ nodes with binary branches correspond to simple labeling, since exactly two subtrees are always involved. On the basis of the underlying CFG rule, the head and the modifier are determined and labeled. In particular, the treatment of binary trees whose binary branches only involve pre-terminal nodes depends exclusively on lexical nodes $n \in \mathcal{T}$. Given two terminal nodes (n_1, n_2), described by $\langle \mathrm{l}_1\text{::}\mathrm{pos}_1, \mathrm{l}_2\text{::}\mathrm{pos}_2 \rangle$, the mark-up rule for their direct (non pre-terminal) ancestor is

$$pos_2/pos_1[h = n_2, m = n_1] \leftarrow (n_1, n_2) \tag{5}$$

where $[h = p_2, m = p_1]$ denotes that the second leaf node has been selected as the *head*, and the first as the *modifier*, while the relation is rel_{pos_2,pos_1}. Figure 2 shows a fully labeled tree for the sentence whose unlabeled version has been shown in Figure 1. The np node spanning the *polar bear* phrase is labeled s $\langle np, nns/jj, (bear :: n, polar :: j)\rangle$.

Notice how usually the grammatical type assigned to a node does not change the assignment

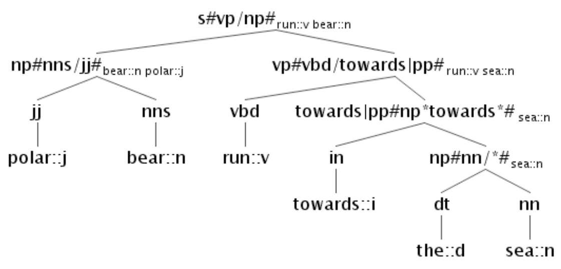

Figure 2: Marking of a Compositional constituency tree of the example in Figure 1

already provided by the tree. In some particular cases, the pair head h and modifier m and the syntactic relation $rel_{h,m}$ implied by a node in the tree are not fully defined either because some null semantic content is encountered (e.g. a missing modifier) or because it is not possible to model some lexical as it is not present in distributional semantic representation. Some relations exist where the modifier seems not to carry relevant lexical information, as the case of the relation between a determiner and a noun. In these cases, the modifier of a non pre-terminal node as well as the syntactic relation are neglected and null slots, i.e. $*$, are used. An example in Figure 2 is the labeling of the bi-gram *the sea*.

The labeling of prepositional phrases constitute a somehow special case of the marking of non pre-terminal nodes $\mathcal{NPT}$. The lexical information carried out by prepositions is not directly expressed lexically but it is integrated both in the grammatical type gT and in the $rel_{h,m}$. This is the case of the triple $\langle towards|pp, np/towards, (sea :: n, *)\rangle$ in Figure 2. The treatment of binary trees whose binary branches involves pre-terminal nodes or non pre-terminal nodes is also straightforward. Where $\mathcal{PT}$ nodes depend strictly from the lexical leaves, $\mathcal{NPT}$ node dominate complex subtrees. The main difference with respect the Equation 5 is that n_1, n_2 or both subtrees may correspond to $\mathcal{NPT}$ node n_i. In a bottom up fashion, the head modifier pair (h_i, m_i) already assigned to n_i is propagated upward. The dominating node is marked-up according to the head h_i of its corresponding dominated branches. For $\mathcal{NPT}$ nodes, the head and the modifier are still assigned by Equation 5, whereas only the heads of the involved subtrees are used. For example, in Figure 2, the root is marked as $\langle s, np/vp, (run :: v, bear :: n)\rangle$ according to the heads, i.e. *run::v* and *bear::n*, of the corresponding branches (i.e. the right *vp* and the left *np*). When $\mathcal{NPT}$ nodes have more than 2 branches (e.g. all pre-terminal nodes or other $\mathcal{NPT}$ nodes), criteria depending on the specific context free rules of the underlying grammar are adopted to select the proper head and modifier.

4 The Compositionally Smoothed Partial Tree Kernel

When the compositionally enriched parse tree is available, it is possible to measure the similarity between this constituency structures through a Tree Kernel. We define here the Compositionally Smoothed Partial Tree Kernel (CSPTK) as a similarity function for such that structures, by extending the SPTK formulation. Let us consider the application of the SPTK on the tree shown in Figure 2. When estimating the similarity with a tree derived from sentences such as "*Bear market runs towards the end*" or "*The commander runs the offense*", the kernel will estimate the similarity among all nodes. Then, the σ function in Equation 4 would not be able to exploit the different senses of the verb *run*, as a traditional distributional model would provide a unique representation. The aim of the CSPTK is to exploit the observable compositional relationships in order to emphasize the contributions of the compounds to the overall meaning, even where the syntactic structure does not change, such as in "*run the offense*", i.e. attacking someone, vs. "*run towards the end*".

The core novelty of the CSPTK is the new estimation of σ as described in Algorithm 1. For the terminal nodes (i.e. LEX type) a lexical kernel σ_{LEX}, i.e. the cosine similarity between words sharing the same POS-Tag, is applied. Otherwise between pre-terminal nodes, a strong matching is required, assigning 0/1 similarity only if pre-terminal nodes share the same POS. The novel part of Algorithm 1 is introduced with the similarity computation over non pre-terminal nodes. In order to activate the similarity function between $\mathcal{NPT}$ nodes, they must have the same gT and $rel_{h,m}$. In this case, the Subspace operator in Equation 2 is applied between the involved (h, m) compounds: lexical information pairs are checked and if their respective heads and modifiers share the corresponding POS, the compositional similarity function is applied.

As discussed in Section 3.1, modifier could be missing in lexical information pair. The DCS model is applied according to three strategies:

General case. If nodes have both heads and modifiers, the similarity function of Equation 4 is applied as usual. Notice that the *pos-tags* of heads and modifiers must be the same.

A modifier is missing. An "optimistic" similarity estimator can be defined in this case. Let be $(h_x, *)$ and (h_y, m_y) the lexical information of two nodes x (that lacks of the modifier) and y. The forced pair (h_x, m_y) and the pair (h_y, m_y) projected and compared into their own subspaces, provide a measure of how the head h_x is similar to h_y, with respect to the meaning that they evoke together with m_y. The more h_x and h_y could be both modified by m_y to specify the same meaning, the higher is the received score.

Both modifiers are missing. This case is reduced to the treatment of lexical nodes (i.e. LEX type), and no composition is observed: the lexical kernel σ_{LEX}, i.e. the cosine similarity between word vectors, is adopted as no subspace is needed.

Algorithm 1 $\sigma_\tau(n_x, n_y, lw)$ Compositional estimation of the lexical contribution to semantic tree kernel

```
$\sigma_\tau \leftarrow 0,$
**if** $n_x = \langle lex_x{::}pos \rangle$ **and** $n_y = \langle lex_y{::}pos \rangle$ **then**
    $\sigma_\tau \leftarrow lw \cdot \sigma_{LEX}(n_1, n_2)$
**end if**
**if** $n_x = pos$ **and** $n_x = n_y$ **then**
    $\sigma_\tau \leftarrow 1$
**end if**
**if** $n_x = \langle gT, syntRel, \langle li_x \rangle \rangle$ **and** $n_y = \langle gT, syntRel, \langle li_y \rangle \rangle$
**then**
    /*Both modifiers are missing*/
    **if** $li_x = \langle h_x{::}pos \rangle$ **and** $li_y = \langle h_y{::}pos \rangle$ **then**
        $\sigma_\tau \leftarrow \sigma_{COMP}\big((h_x), (h_y)\big) = \sigma_{LEX}(n_x, n_y)$
    **end if**
    /*One modifier is missing*/
    **if** $li_x = \langle h_x{::}pos_h \rangle$ **and** $li_y = \langle h_y{::}pos_h, m_y{::}pos_m \rangle$ **then**
        $\sigma_\tau \leftarrow \sigma_{COMP}\big((h_x, m_y), (h_y, m_y)\big)$
    **end if**
    /*General Case*/
    **if** $li_x = \langle h_x{::}pos_h, m_x{::}pos_m \rangle$ **and**
    $li_y = \langle h_y{::}pos_h, m_y{::}pos_m \rangle$ **then**
        $\sigma_\tau \leftarrow \sigma_{COMP}\big((h_x, m_x), (h_y, m_y)\big)$
    **end if**
**end if**
**return** $\sigma_\tau$
```

Notice that Algorithm 1 could be be still modified further depending on how the non terminal similarity has to be strict on gT and $rel_{h,m}$ and on how much is the weight of terminal and pre-terminal nodes.

5 Experimental Evaluations

In this section the CSPTK model is used in Semantic Textual Similarity (STS) and Question Classification (QC) tasks. The aim of this section is to measure the CSPTK capability to account for the similarity between sentences and as a feature to train machine learning classifiers.

5.1 Experimental Setup

In all experiments, sentences are processed with the Stanford CoreNLP[2], for Part-of-speech tagging, lemmatization, and dependency and compositionally enriched parsing. In order to reduce data sparseness introduce by fined grained Part-of-Speech classes, node are marked by coarse grained classes, e.g. *looking*:`VBG` or *looked*:`VBD` are simplified in *look*:`V`. In order to estimate the basic lexical similarity function employed in the Tree Kernels operators, a co-occurrence Word Space is acquired through the distributional analysis of the UkWaC corpus (Baroni et al., 2009). First, all words occurring more than 100 times (i.e. the *targets*) are represented through vectors. The original space dimensions are generated from the set of the 20,000 most frequent words (i.e. *features*) in the UkWaC corpus. A co-occurrence Word-Space with a window of size 3 is acquired. Co-occurrencies are weighted by estimating the Point-wise Mutual Information between the 20k most frequent words. The SVD reduction is then applied with a dimensionality cut of $d = 250$. Left contexts are treated differently from the right ones, in order to capture asymmetric syntactic behaviors (e.g., useful for verbs): 40,000 dimensional vectors are thus derived for each target. Similarity between lexical nodes is estimated as the cosine similarity in the co-occurrence Word Space, as in (Croce et al., 2011).

5.2 The Semantic Text Similarity task

The first experiment aims to evaluate the contribution of the Kernel-based operators in a STS task. In the **Core STS task** given two sentences, s_1 and s_2, participants are asked to provide a score reflecting the corresponding text similarity (Agirre et al., 2013). PTK, SPTK and CSPTK similarity functions are employed over the dataset of the *SEM 2013 shared task. In Table 1 results of Pearson Correlations between the Kernels operators and the human scores are shown. We considered all datasets composing the challenge training set, i.e. MSRvid, MSRpar, SMTeuroparl, surprise.OnWn and surprise.SMTnews as well as the test set Headlines, FNWN and SMTnews. We did not report any comparison with the best results of the SemEval STS competition as those approaches are mostly supervised. On the contrary the presented approach for the STS estimation is fully unsupervised. The purpose of the experiments is to i) investigate the differences between Kernels operators when lexical semantics (i.e. SPTK and CSPTK) is added to the syntactic information (i.e. PTK), ii) analyze the role of the the compositional compounds made explicit in parse trees and iii) measure the contribution of the DCS model adopted in the recursive CSPTK computation.

PTK and SPTK functions are both applied to the Constituency Tree representations, labeled with ct, while the CSPTK model consists in: i) lexical mark-up as a form of lexical compositional caching that generates the input Compositionally labeled Constituency Tree representation (denoted by cct) as introduced and discussed in Section 3 and ii) the matching function among the subtrees [3]

Dataset	PTK$_{ct}$	SPTK$_{ct}$	CSPTK$_{cct}$
MSRVid	.12	.18	**.65**
MSRPar	.26	.28	**.32**
SMTEuroparl	.45	.45	**.50**
surprise.OnWN	.49	.55	**.59**
surprise.SMTNews	.46	.46	**.46**
FNWN	.15	.19	**.21**
Headlines	.40	.49	**.52**
OnWN	.04	.24	**.37**
SMTNews	.28	.31	**.33**

Table 1: Unsupervised results of Pearson correlation for Kernel-based features adopted in *SEM - Task 6 datasets

First and second columns in Table 1 show Pearson results of PTK$_{ct}$ and SPTK$_{ct}$ functions applied over a constituency tree, while the last column shows the CSPTK$_{cct}$ results over the compositionally labeled tree. Notice how the introduction of the compositionality enrichment in a constituency tree structure, together with the CSPTK function led to a performance boost over all the training and test datasets. In some cases, the boost between SPTK$_{ct}$ and CSPTK$_{cct}$ is remarkable, switching from .18 to .65 in MSRvid and from .24 to .37 in OnWN.

The above difference is mainly due to the increasing sensitivity of PTK, SPTK and CSPTK to the incrementally rich lexical information. This is especially evident in sentence pairs with very

[2] `http://nlp.stanford.edu/software/corenlp.shtml`

[3] For the SPTK, we selected the parameters $\lambda = 0.1$, $\mu = 0.1$ and $lexical\_weight = 100$ that provided best results in (Croce et al., 2012). Otherwise for CSPTK we selected $\lambda = 0.4$, $\mu = 0.4$ and $lexical\_weight = 10$.

similar syntactic structure. For example in the MSRvid dataset, a sentence pair is given by *The man are playing soccer* and *A man is riding a motorcycle*, that are strictly syntactically correlated. As a side effect, PTK provides a similarity score of .647 between the two sentences. It is a higher score with respect to the SPTK and CSPTK: differences between tree structures are confined only to the leaves. By scoring .461, SPTK introduces an improvement as the distributional similarity (function σ in Eq. 4) that acts as a smoothing factor between leaves better discriminates uncorrelated words, like *motorcycle* and *soccer*. However, ambiguous words such verbs *ride* and *play* are still promoting a similarity that is locally misleading. Notice that both PTK and SPTK receive a strong contribution in the recursive computation of the kernels by the left branching of the tree, as the subject is the same, i.e. *man*. Compositional information about direct objects (*soccer* vs. *motorcycle*) is better propagated by the CSPTK operator. Its final scores for the pair is .36, as semantic differences between the sentences are emphasized. Even if grammatical types strongly contribute to the final score (as in PTK or SPTK), now the DCS computation over these nodes (the compounds traced from the leaves, i.e. (*ride::v, motorcycle::n*) and (*play::v, soccer::n*) is faced with less ambiguous verb phrases, that contribute with lower scores.

5.3 CSPTK for Question Classification

Thanks to structural kernel similarity, a question classification (QC) task can be easily modeled by representing questions, i.e., the classification targets, with their parse trees. The aim of the experiments is to analyze the role of lexical similarity embedded in the compositionally enriched constituency trees by the CSPTK operator. Thus, questions have been represented by the classic constituency tree, i.e. *ct*, and by the compositionally enriched variant, i.e. cct. The first representation is used over PTK and SPTK functions, while the latter over the CSPTK function. Our referring corpus is the UIUC dataset (Li and Roth, 2002). It is composed by a training set of 5,452 questions and a test set of 500 questions[4]. The latter are organized in six coarse-grained classes, i.e. ABBREVIATION, ENTITY, DESCRIPTION, HUMAN, LOCATION and NUMBER. For learning our models we employed LIBSVM[5] after computing the entire Gram Matrix. The F1 of SVMs using (i) PTK and SPTK applied to ct and (ii) CSPTK applied to cct for QC, is reported in Table 2. Notice that we want to carry out a comparative evaluation of different syntactic kernels and not to optimize the QC accuracy as this requires a combination of lexical and syntagmatic kernels as discussed in (Croce et al., 2011). The results are in general outperforming the alternative kernel formulations, even if the improvement is modest. First, we outline that a more stable behavior wrt parameters is observed, with a corresponding lower risk of over-fitting the training data. Second, it is to be noticed that a large number of questions have a really simple syntactic structure: as a consequence the interrogative form of the sentence is very simple and very few compositional phenomena are observed that are captured by the distributional information about word vectors.

SVM par	PTK$_{ct}$	SPTK$_{ct}$	CSPTK$_{cct}$
c=1	.78	.89	**.91**
c=2	.83	.90	**.90**
c=5	.88	.92	**.92**

Table 2: Results in the Question Classification task

6 Conclusions

In this paper, a novel kernel function has been proposed in order to exploit Distributional Compositional operators within Tree Kernels. The proposed approach propagates lexical semantic information over an entire constituency parse tree, by building a Compositionally labeled Constituency Tree. It enables the definition of the Compositional Smoothed Partial Tree Kernel as a Convolution Kernel that measures the semantic similarity between complex linguistic structures by applying metrics sensible to the compositionality. First empirical results of the CSPTK in STS and QC tasks demonstrate the robustness and the generalization capability of the proposed kernel. Future investigation is needed in the adoption of the same compositionality perspective on dependency graphs, were the compositional representations of the head/modifier compound are even more explicit. Further experiments for assessing the methodology are also foreseen against other NLP-tasks, e.g. verb classification.

[4]http://cogcomp.cs.illinois.edu/Data/QA/QC/

[5]http://www.csie.ntu.edu.tw/cjlin/libsvm/

References

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA, June. Association for Computational Linguistics.

P. Annesi, V. Storch, and R. Basili. 2012. Space projections as distributional models for semantic composition. In *In Proceedings of CICLing 2012*, volume 7181 of *Lecture Notes in Computer Science*, pages 323–335. Springer.

M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

P. Blackburn and J. Bos. 2005. *Representation and Inference for Natural Language. A First Course in Computational Semantics*. CSLI Publications.

M. Collins and N. Duffy. 2001. Convolution kernels for natural language. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 625–632.

D. Croce, A. Moschitti, and R. Basili. 2011. Structured lexical similarity via convolution kernels on dependency trees. In *Proceedings of EMNLP*, Edinburgh, Scotland, UK.

D. Croce, P. Annesi, V. Storch, and R. Basili. 2012. Unitor: Combining semantic text similarity functions through sv regression. In **SEM 2012*, pages 597–602, Montréal, Canada, 7-8 June.

K. Erk and S. Pado. 2008. A structured vector space model for word meaning in context. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 897–906. ACL.

P. Foltz, W. Kintsch, and T. Landauer. 1998. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25:285–307.

E. Guevara. 2010. A regression model of adjective-noun compositionality in distributional semantics. In *Proceedings of the GEMS '10*, pages 33–37, Stroudsburg, PA, USA. Association for Computational Linguistics.

D. Haussler. 1999. Convolution kernels on discrete structures. Technical report, Dept. of Computer Science, University of California at Santa Cruz.

H. Kamp and U. Reyle. 1993. *From discourse to logic: introduction to modeltheoretic semantics of natural language, formal logic and discourse representation theory. Part 1*. Kluwer Academic.

T. Landauer and S. Dumais. 1997. A solution to platos problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, pages 211–240.

X. Li and D. Roth. 2002. Learning question classifiers. In *Proceedings of ACL '02*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics.

J. Mitchell and M. Lapata. 2008. Vector-based models of semantic composition. In *In Proceedings of ACL/HLT 2008*, pages 236–244.

J. Mitchell and M Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

H. Schütze. 1998. Automatic word sense discrimination. *Comput. Linguist.*, 24(1):97–123, March.

P. Smolensky. 1990. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artif. Intell.*, 46:159–216, November.

Richard Socher, John Bauer, Christopher D. Manning, and Ng Andrew Y. 2013. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, Sofia, Bulgaria, August. Association for Computational Linguistics.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141.

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)*.