10.12871/CLICIT201412

Towards Compositional Tree Kernels

Paolo Annesi, Danilo Croce and Roberto Basili
Department of Enterprise Engineering
University of Roma, Tor Vergata
Via del Politecnico 1, 00133 Roma, Italy
{annesi,croce,basili}@info.uniroma2.it

Abstract

English. Several textual inference tasks rely on kernel-based learning. In particular Tree Kernels (TKs) proved to be suitable to the modeling of syntactic and semantic similarity between linguistic instances. In order to generalize the meaning of linguistic phrases, Distributional Compositional Semantics (DCS) methods have been defined to compositionally combine the meaning of words in semantic spaces. However, TKs still do not account for compositionality. A novel kernel, i.e. the Compositional Tree Kernel, is presented integrating DCS operators in the TK estimation. The evaluation over Question Classification and Metaphor Detection shows the contribution of semantic compositions w.r.t. traditional TKs.

Italiano. *Sono numerosi i problemi di interpretazione del testo che beneficiano dall'applicazione di metodi di apprendimento automatico basato su funzioni kernel. In particolare, i Tree Kernel (TK) sono applicati alla modellazione di metriche di similaritá sintattica e semantica tra espressioni linguistiche. Allo scopo di generalizzare i significati legati a sintagmi complessi, i metodi di Distributional Compositional Semantics combinano algebricamente i vettori associati agli elementi lessicali costituenti. Ad oggi i modelli di TK non esprimono criteri di composizionalitá. In questo lavoro dimostriamo il beneficio di modelli di composizionalitá applicati ai TK, in problemi di Question Classification e Metaphor Detection.*

1 Introduction

Tree Kernels (TKs) (Collins and Duffy, 2001) are consolidated similarity functions used in NLP for their ability in capturing syntactic information directly from parse trees and used to solve complex tasks such as Question Answering (Moschitti et al., 2007) or Semantic Textual Similarity (Croce et al., 2012). The similarity between parse tree structures is defined in terms of all possible syntagmatic substructures. Recently, the Smoothed Partial Tree Kernel (SPTK) has been defined in (Croce et al., 2011): the semantic information of the lexical nodes in a parse tree enables a smoothed similarity between structures, which are partially similar and whose nodes can differ but are nevertheless related. Semantic similarity between words is evaluated in terms of vector similarity in a Distributional Semantic Space (Sahlgren, 2006; Turney and Pantel, 2010; Baroni and Lenci, 2010). Even if achieving higher performances w.r.t. traditional TKs, the main limitations of SPTK are that the discrimination between words is delegated only to the lexical nodes and semantic composition of words is not considered.

We investigate a kernel function that exploits semantic compositionality to measures the similarity between syntactic structures. In our perspective the semantic information should be emphasized by compositionally propagating lexical information over an entire parse tree, making explicit the head/modifier relationships between words. It enables the application of Distributional Compositional Semantic (DCS) metrics, that combine lexical representations by vector operator into the distributional space (Mitchell and Lapata, 2008; Erk and Pado, 2008; Zanzotto et al., 2010; Baroni and Lenci, 2010; Grefenstette and Sadrzadeh, 2011; Blacoe and Lapata, 2012; Annesi et al., 2012), within the TKs computation. The idea is to i) define a procedure to mark nodes of a parse tree that allows to spread lexical bigrams across the tree nodes ii) apply DCS smoothing metrics between such compositional nodes iii) enrich the SPTK formulation with compositional distributional seman-

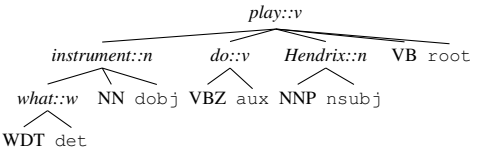

Figure 1: Lexical centered tree of the sentence "*What instrument does Hendrix play?*"

tics. The resulting model has been called Compositional Smoothed Partial Tree Kernel (CSPTK). The entire process of marking parse trees is described in Section 2. Therefore, in Section 3 the CSPTK is presented. Finally, in Section 4, the evaluations over Question Classification and Metaphor Detection tasks are shown.

2 Explicit compositions in Parse Trees

Compositional semantic constraints over a tree kernel computation can be applied when syntagms corresponding to nodes are made explicit. Given the question "*What instrument does Hendrix play?*" and its dependency structure, the corresponding syntactic structure is shown in Figure 1 in terms of a Lexically Centered Tree (LCT), as in (Croce et al., 2011). Nodes are partitioned into: **lexical** nodes in terms of non-terminals $\langle l_n{::}pos_n \rangle$, such as *instrument::n*, where *l* is the lemma of the token and `pos` the part-of-speech; **syntactic** nodes, i.e. children of each lexical node which encodes a dependency function $d \in \mathcal{D}$ (e.g. PREP$_{OF}$) and the `pos`-tag of the parent (e.g. NN).

In order to introduce lexical compositionality to these syntactic representations, a mark-up process is introduced, enabling the compositional extension of the tree kernel. Each link between two non-terminal nodes in a LCT representation reflects a dependency relation d, encoded by the child of the lowest non-terminal node. For example, the dependency between the node *instrument::n* and its parent node *play::v* is of type `dobj`. Thus, semantic compositionality is introduced in terms of a head/modifier pair (h, m) over non-terminal nodes, where lexical head is always the upper node. Every non-terminal node is now marked as

$$\langle d_{h,m}, \langle l_h :: pos_h, l_m :: pos_m \rangle \rangle \qquad (1)$$

Figure 2 shows a fully compositionally labeled tree, called Compositional Lexically Centered Tree (CLCT), for the sentence whose unlabeled version has been shown in Figure 1. Now nodes are partitioned so that: non-terminal nodes represent **compositional lexical pairs** (h, m) marked as in Equation 1: notice that the modifier is missing in the root node; **dependency functions** (`dobj`) and **POS-Tags** (VBZ) are encoded in the terminal nodes as in the original LCT; **lexical nodes**, e.g. *play::v*, are repeated as terminal nodes, in order to reduce data sparseness that may be introduced by considering only compositional compounds. A DCS model can be adopted, allowing to estimate an expressive similarity function between head-modifier pairs (h_1, m_1), (h_2, m_2) within the resulting kernel. In (Mitchell and Lapata, 2008) three general classes of compositional models have been defined: a linear *additive* model $\vec{p} = \mathbf{A}\vec{u} + \mathbf{B}\vec{v}$; a *multiplicative* model $\vec{p} = \mathbf{C}\vec{u}\vec{v}$ and the *dilation* model $\vec{p}_d = (\vec{u} \cdot \vec{u})\vec{v} + (\lambda - 1)(\vec{u} \cdot \vec{v})\vec{u}$. $\mathbf{A}$ and $\mathbf{B}$ are weight matrices; $\mathbf{C}$ 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; eventually, dilation is an asymmetric function where $\vec{u}$ can be used to dilate $\vec{v}$, and viceversa according with a dilation factor λ. Another compositional model adopted here is the so-called **Support Subspace**, proposed in (Annesi et al., 2012), which assumes that a composition is expressed by projecting vectors into subspaces. A projection reflects a selection function over the set of semantic features shared in the (h, m) compound. A subspace local to (h, m) can be found such that only the space dimensions specific to its meaning are selected. Support Subspaces seem very effective for simple syntactic structures by capturing bi-gram semantics, but they are not sensitive to complex linguistic structures.

3 The Compositional Smoothed Partial Tree Kernel

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 T_1 and T_2 are parse trees, while N_{T_1} and N_{T_2} are the sets of the T_1's and T_2's nodes. The Δ function recursively computes the amount of similarity between tree structures in terms of the similarity among substructures. The type of considered fragments determines the expressiveness of the kernel space and different tree kernels are characterized by different choices. In early models, e.g. (Collins and Duffy, 2001), lexical generalization has been neglected in the recursive matching, so that only exact matching between node labels was given a weight higher than 0. Lexical contribution was proposed by (Croce et al., 2011), in the so called Smoothed Partial Tree Kernel (SPTK). In SPTK, the TK extends

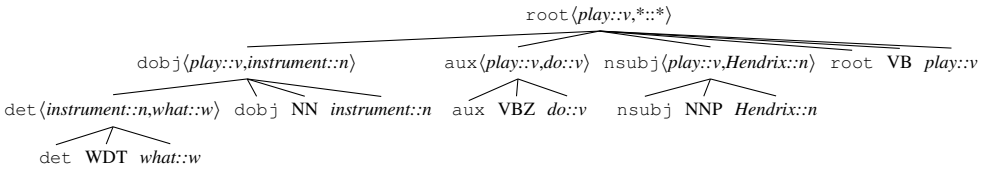

Figure 2: Compositional Lexically Centered Tree (CLCT) of the sentence "*What instrument does Hendrix play?*"

the similarity between tree structures allowing a smoothed function of node similarity σ. It allows to measure the similarity between syntactic tree structures, which are semantically related even when lexical nodes differ. This is achieved by the following formulation of the function Δ:

$\Delta_\sigma(n_1, n_2) = \mu\lambda\sigma(n_1, n_2)$, where n_1 and n_2 are leaves, else

$$\Delta_\sigma(n_1, n_2) = \mu\sigma(n_1, n_2)\Big(\lambda^2 + \sum_{\vec{I}_1, \vec{I}_2, l(\vec{I}_1)=l(\vec{I}_2)} \lambda^{d(\vec{I}_1)+d(\vec{I}_2)} \prod_{j=1}^{l(\vec{I}_1)} \Delta_\sigma(c_{n_1}(\vec{I}_{1j}), c_{n_2}(\vec{I}_{2j}))\Big) \quad (2)$$

In Eq. 2 , $\vec{I}_{1j}$ 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. The semantic similarity between nodes is measure by $\sigma(n_1, n_2)$.

One main limitation of SPTK is that σ does not consider compositional interaction between words. Given the phrases "*to play sport*" and "*to play instrument*", the SPTK relies only on a unique meaning for *play*, ignoring the compositional role of each modifier. 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 "*What instrument does Hendrix play?*" or "*What sport does Bolt play?*", the kernel will estimate the similarity among all nodes. Then, the σ function in Equation 2 would not be able to exploit the different senses of the verb *play*, as a traditional DCS model would provide a unique vector representation.

The Compositional Smoothed Partial Tree Kernel (CSPTK) tries to overcome this limitation by measuring the similarity between constituency structures in which lexical compositionality have been made explicit. DCS operators are employed within the CSPTK computation. The core novelty of the CSPTK is the new estimation of σ as described in Algorithm 1. For the lexical nodes the kernel σ_{LEX} is applied, i.e. the cosine similarity between words sharing the same `pos`-tag. Moreover, the other non-lexical nodes contribute according to a strict matching policy: they provide full similarity only when the same `pos`, or

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

```
σ_τ ← 0,
if n_x = ⟨lex_x::pos⟩ and n_y = ⟨lex_y::pos⟩ then
    σ_τ ← σ_LEX(n_1, n_2)
end if
if (n_x = pos or n_x = dep) and n_x = n_y then
    σ_τ ← lw
end if
if n_x = ⟨d_{h,m}, ⟨li_x⟩⟩ and n_y = ⟨d_{h,m}, ⟨li_y⟩⟩ then
    /*Both modifiers are missing*/
    if li_x = ⟨h_x::pos⟩ and li_y = ⟨h_y::pos⟩ then
        σ_τ ← σ_COMP((h_x), (h_y)) = σ_LEX(n_x, n_y)
    end if
    /*One modifier is missing*/
    if li_x = ⟨h_x::pos_h⟩ and li_y = ⟨h_y::pos_h, m_y::pos_m⟩
    then
        σ_τ ← σ_COMP((h_x, h_x), (h_y, m_y))
    else
        /*General Case*/
        σ_τ ← σ_COMP((h_x, m_x), (h_y, m_y))
    end if
end if
return σ_τ
```

dependency, is matched and 0 otherwise. The factor lw is here adopted to reduce the contribution of non-lexical nodes. The novel part of Algorithm 1 is introduced with the similarity computation over compositional nodes. In order to activate the similarity function between non-terminal nodes, they must have the same $d_{h,m}$. In this case a DCS metric can be applied between the involved (h, m) compounds: the lexical information related to pairs are checked and if their respective heads and modifiers share the corresponding POS, a compositional similarity function is applied.If a *modifier is missing*, e.g. the compounds are $(h_x, *)$ and (h_y, m_y), the virtual pair (h_x, h_x) and the pair (h_y, m_y) are used; if *both modifiers are missing*, e.g. the compounds are $(h_x, *)$ and $(h_y, *)$, the σ_{LEX}, i.e. the cosine similarity between word vectors, is adopted.

4 Experimental Evaluation

We evaluated CSPTK w.r.t. two inference tasks, i.e. Question Classification (QC) and Metaphor Detection (MI). Texts are processed with Stanford CoreNLP and compositional trees are generated as discussed in Section 2. The lexical similarity func-

tion is derived from a co-occurrence Word Space, acquired through the distributional analysis of the UkWaC corpus, as in (Croce et al., 2011).

CSPTK in Question Classification. In the QC task, the reference corpus is the UIUC dataset (Li and Roth, 2002), including 5,452 questions for training and 500 questions for test, organized in six coarse-grained classes. SVM training has been carried out over the UIUC by applying (i) the PTK and SPTK kernels over the LCT representation of the questions and (ii) the compositional tree kernels (CSPTKs), according to different compositional similarity metrics σ_{COMP}, to the CLCT representation. For learning our models, we used an extension of the SVM-LightTK software. Different compositional kernels are distinct according to the adopted compositionality metrics: *simple additive model* (Mitchell and Lapata, 2010), denoted by a "$^+$" superscript with $\alpha = \beta$; the *pointwise product operator*, denoted by a "$^{\cdot}$" superscript; the *dilation operator* model, denoted by a ^{d} superscript with $\lambda = 1$; the *support subspace* model of (Annesi et al., 2012), denoted by ^{SS}.

Kernel	Accuracy	Std. Dev.
BoW	86.3%	±0.3%
PTK_{LCT}	90.3%	±1.8%
$SPTK_{LCT}$	92.2%	±0.6%
$CSPTK^{+}_{CLCT}$	**95.6%**	±0.6%
$CSPTK^{\cdot}_{CLCT}$	94.6%	±0.5%
$CSPTK^{d}_{CLCT}$	94.2%	±0.4%
$CSPTK^{ss}_{CLCT}$	93.3%	±0.7%

Table 1: Results in the Question Classification task

In Table 1 the accuracy achieved by the different systems is reported as the percentage of sentences correctly assigned to the proper question class. As a baseline, a simple bag-of-word model (i.e. BoW) is also computed: it represents questions as binary word vectors and it results in a kernel measuring the lexical overlap. The introduction of lexical semantic information in tree kernel operators, such as in SPTK vs. PTK, is beneficial thus confirming the outcomes of (Croce et al., 2011). $CSPTKs$ seem to make an effective use of the lexical semantic smoothing as they all outperform the non-compositional counterparts. In particular $CSPTK^{+}_{CLCT}$ outperforms all the other compositional operators. Eventually, the error reduction ranges between 12% and 42%.

CSPTK for Metaphor Detection. For the second experiment we choose the annotated Metaphor corpus by (Hovy et al., 2013). The task consists to classify the target words use as literal or metaphorical. The dataset consists of 3,872 sentences divided into training, development, and test sets, using a 80-10-10 split. In Table 2, the accuracy achieved by the different systems is reported. The complexity of the task is confirmed by the low inter annotator agreement achieved over the dataset, i.e. 0.57. As detecting metaphor depends on the deep interaction among words, it seems reasonable that the models using only syntactic information (i.e. PTK) or distributional words in isolation (i.e. BoW) or both (i.e. SPTK) achieve poor performances. The method proposed in (Srivastava et al., 2013) confirms the impact of a proper semantic generalization of the training material. It reaches the SoA by applying a walk-based graph kernel that generalizes the notion of tree kernel as a general framework for word-similarity, and incorporates distributed representations in a flexible way. In our test the syntactic information together with the compositional smoothing, activated by the compositional nodes of the CSPTK, make also an effective use of the lexical semantic smoothing and outperform all the non-compositional counterparts, achieving an accuracy of 75.3%. Even though CSPTK does not outperform (Srivastava et al., 2013), it represents a completely automatic method, largely applicable to different tasks.

Kernel	Accuracy
BoW	71.3%
PTK_{LCT}	71.6%
$SPTK_{LCT}$	71.0%
$CSPTK^{+}_{CLCT}$	72.4%
$CSPTK^{ss}_{CLCT}$	**75.3%**
(Srivastava and Hovy, 2013)	**76.0%**

Table 2: Results in the Metaphor Detection task

5 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 tree, by building a Compositionally labeled Tree. The resulting Compositional Smoothed Partial Tree Kernel measures the semantic similarity between complex linguistic structures by applying metrics sensible to distributional compositional semantics. Empirical results in the Question Classification and Metaphor Detection tasks demonstrate the positive contribution of compositional information for the generalization capability within the proposed kernel.

References

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.

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

William Blacoe and Mirella Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 546–556, Stroudsburg, PA, USA. Association for Computational Linguistics.

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.

Edward Grefenstette and Mehrnoosh Sadrzadeh. 2011. Experimental support for a categorical compositional distributional model of meaning. In *EMNLP*, pages 1394–1404. ACL.

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.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *ACL*. The Association for Computer Linguistics.

Magnus Sahlgren. 2006. *The Word-Space Model*. Ph.D. thesis, Stockholm University.

Shashank Srivastava and Dirk Hovy, 2013. *A Walk-based Semantically Enriched Tree Kernel Over Distributed Word Representations*, pages 1411–1416. Association for Computational Linguistics.

Shashank Srivastava, Dirk Hovy, and Eduard H. Hovy. 2013. A walk-based semantically enriched tree kernel over distributed word representations. In *EMNLP*, pages 1411–1416.

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 '10, pages 1263–1271, Stroudsburg, PA, USA. Association for Computational Linguistics.