Combining tree kernels and tree representations to classify argumentative stances

Davide Liga[1,2][0000−0003−1124−0299] and Monica Palmirani[1][0000−0002−8557−8084]

[1] Alma Mater Studiorum - University of Bologna, Bologna, Italy
{monica.palmirani,davide.liga2}@unibo.it
[2] University of Luxembourg, Luxembourg

Abstract. This work investigates how the combination of different tree representations with different Tree Kernel functions influences the results of the classifications in two specific case studies. One case study is related to the classification of argumentative stances of support, the other one is related to the classification of stances of opposition. Results show that some Tree Kernels achieves not only higher results but also a higher level of generalization. Moreover, it seems that also the kind of tree representation influences the performances of classifiers. In this study, we thus explore this relation between tree representation and different Tree Kernels, considering also compositional trees.

Keywords: Argument Mining · Tree kernel · Argumentative stance.

1 Introduction

This study is related to the field of Argument Mining (AM), a relatively new research field focused on the analysis, detection and classification of argumentative structures, substructures and units from natural argumentation. On the one side, AM is a field which employs Natural Language Processing techniques. On the other side, AM is also connected to the field of Argumentation, which involves a wide range of logical and philosophical aspects. The aspects related to natural language and those related to Argumentation are both crucial when dealing with legal data, because legal texts contain several argumentative structures which are encoded in natural language, e.g. inferential logical-ontological rules or even stereotypical patterns of inference (known as Argumentation Schemes). These rules and argumentative patterns are composed of premises (evidences) and conclusions (claims) which can be classified using Machine Learning algorithms, facilitating the automatic detection of argumentative structures and rules in legal texts.

Due to the complexity of human language, AM scholars often need to create classifiers employing highly-engineered features capable of describing the complexity of argumentative structures in natural language. As suggested in

Lippi and Torroni 2015 [13], the problem of having to engineer features to catch complex structures can be solved by employing classifiers that works directly on structures, such as Tree Kernel classifiers. Following this suggestion, recent studies have shown how Tree Kernels classifiers can discriminate between different kinds of argumentative stances of support [10] and opposition [11].

For example, Liga 2019 [10] described a first attempt to use Tree Kernel classifiers to discriminate argumentative stances of support in the context of a binary classification. On the other side, Liga and Palmirani 2019 [11] showed that a similar approach can be applied also to opposition stances and to a multi-class classification problem. The present work takes inspiration from these two studies and reproduces their settings by using different tree representations and tree kernel functions, to assess whether or not there is a specific combination that better suits the task of classifying/discriminating argumentative stances.

In Section 2, Tree Kernels classifiers will be introduced along with a description of different tree representations and Tree Kernel functions. In the Section 3, we will briefly describe the related works in the domain of AM, considering the studies which employed Tree Kernels in AM and describing the two above-mentioned studies, [10] and [11], from a general perspective. In Section 4, we will reproduce the settings of Liga 2019 [10] applying different tree representations and different Tree Kernel functions to the same scenario and analyzing the results of each classifier. In Section 5, we will do the same process, employing different tree representations and different Tree Kernel functions in the setting of the experiment in Liga and Palmirani 2019 [11]. Lastly, in Section 6 we will open a short discussion and conclude the work.

2 Tree Kernels and tree representations

A Tree Kernel is simply a similarity measure. It works by comparing the similarity between tree-structured pieces of data. Supposing that we want to use Tree Kernel classifiers for textual data, there are two main elements to consider.

The first element is the kind of tree-structure to employ, namely the kind of tree representation we want to use. The second element is defining which fragments of the tree structures should be involved into the calculation of the similarity. The following sections briefly describe these two aspects.

2.1 TREE REPRESENTATIONS

The idea is that our data (e.g. textual data such as sentences) must be represented into a specific tree-structured shape to allow a Tree Kernel function to calculate the similarity between different pieces of tree-structured data. For example, a sentence can be converted into some kind of tree representation such as a dependency tree or constituency tree.

In the following part, we will shortly describe some of the most famous tree representations for the conversion of textual data into tree structures. They can be considered as particular kinds of Dependency Trees which combine grammatical functions, lexical elements and Part-of-Speech tags in different ways [7].

GRCT The Grammatical Relation Centered Tree (GRCT) representation is a very rich data representation [6]. It involves grammatical, syntactical, lexical elements together with Part-of-Speech and lemmatized words. In this representation, after the root there are syntactical nodes (grammatical relations), then Part-of-Speech nodes and finally lexical nodes. In other words, a tree of this kind is balanced around the grammatical nodes, which determines the structure of dependencies.

LCT Also Lexical Centered Tree (LCT) representations involve grammatical, lexical and syntactical element, along with Part-of-Speech tags. However, the structure of the tree is different. In fact, it is "centered" over Lexical nodes, which are at the second level, immediately after the root. Part-of-Speech nodes and grammatical functions nodes are equally children of the lexical elements.

LOCT The Lexical Only Centered Tree (LOCT) representation contains just the lexical elements. Intuitively, the contribution of LOCT representation can be particularly determinant whenever the tasks to be achieved mostly depend on lexical elements.

cGRCT and cLCT The compositional Grammatical Relation Centered Tree (cGRCT) and the compositional Lexical Centered Tree (cLCT) representations are very similar to the the Grammatical Relation Centered Tree (GRCT) and the Lexical Centered Tree (LTC) representation. The difference here is that the representations allow compositional operators. This aspect will be explained more in depth in the section related to CSPTKs. In fact, cGRCTs and cLCT can be used with Compositionally-Smoothed Partial Tree Kernels (CSPTKs) which are designed specifically for the purpose of considering compositionality [3].

2.2 TREE KERNELS

A kernel function can be considered as a *similarity measure* that perform an implicit mapping $\varphi : \mathcal{X} \to \mathcal{V}$ where $\mathcal{X}$ is a input vector space and $\mathcal{V}$ is a high-dimensional space. A general kernel function can be represented as follows:

$$k(\mathbf{x}, \mathbf{x}') = \langle \varphi(\mathbf{x}), \varphi(\mathbf{x}') \rangle_{\mathcal{V}} \tag{1}$$

Importantly, the $\langle ., . \rangle_{\mathcal{V}}$ in the above formula must necessarily be considered an inner product, while $\mathbf{x}$ and $\mathbf{x}'$ belong to $\mathcal{X}$ and represent the labelled and unlabelled input respectively. If we consider, for example, a binary classification task with a training dataset $\mathrm{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$ composed of n examples, where $y \in \{c_1, c_2\}$ (with c_1 and c_2 being the two possible outputs of a binary classification), the final classifier $\hat{y} \in \{c_1, c_2\}$ can be calculated in the following way:

$$\hat{y} = \sum_{i=1}^{n} w_i y_i k(\mathbf{x}_i, \mathbf{x}') = \sum_{i=1}^{n} w_i y_i \varphi(\mathbf{x}).\varphi(\mathbf{x}') \tag{2}$$

Where the weights w_i are learned by the trained algorithm.

When using Tree Kernels, the function must be adapted to allow the calculations over tree nodes. In this regards, a general Tree Kernel function can be calculated as follows [16]:

$$K(T_1, T_2) = \sum_{n_1 \in N_{T_1}} \sum_{n_2 \in N_{T_2}} \Delta(n_1, n_2) \tag{3}$$

In the above equation, T_1 and T_2 are the two trees involved in the calculation of the similarity, while N_{T_1} and N_{T_2} are their respective sets of nodes and $\Delta(n_1, n_2)$ is the number of common fragments in node n_1 and node n_2.

Importantly, $\Delta(n_1, n_2)$ can be seen as a function considering common fragments between trees. Depending on how this function is configured (i.e., which fragments are considered involved into the calculation of the similarity), different Tree Kernels can be obtained.

Given that our data is tree-structured, the second important element is the definition of the which fragments must be involved when calculating the similarity between trees. Defining which fragments to involve also means defining the Tree Kernel function, because the names of the Tree Kernel functions usually derives from the fragment definition.

In the following part, some famous Tree Kernel functions will be shortly described; each of them defines, in a different way, which fragments should be involved into the calculation of the similarity.

STK In a SubTree Kernel (STK) [18], a fragment is any subtree, i.e. any node of the tree along with all its descendants.

SSTK A SubSetTree Kernel (SSTK) [5] considers as fragments the so-called subset-trees, i.e. it considers any node along with its partial descendancy. Since in SSTKs the only constraint of not breaking grammar production rules, and since fragments' leaves can be also non-terminal symbols, they can be considered a more general representation compared the previously mentioned STKs.

PTK A Partial Tree Kernel [16] is a convolution kernel that considers partial trees as fragments. Similarly to SSTKs, a partial tree is a fragment of a tree which considers a node and its partial descendancy. However, partial trees allow also partial grammar production rules. The fact that production rules can be broken (i.e. partial), makes PTs even more general than SSTs. This is the reason why PTKs should provide a higher ability to generalize.

SPTK A PTK can be also "Smoothed" PTK (SPTK) [6], which adds a further semantic layer into the calculation of node similarity. SPTKs allows to calculate similarities between dependency structures whose surfaces (i.e. the lexical nodes, or words) are partially or totally different. They introduce a lexical similarity which allows the generalization of tree structures through the semantic layer by representing words not just as mere symbols, but as semantic entities.

CSPTK The scenario can be further expanded considering Compositionally-Smoothed Partial Tree Kernels (CSPTK) [4], which apply a composition function between nodes to better represent contextual relations between words.

We have previously showed how compositional trees look like (i.e. cGRCTs and cLCTs), but we did not explain how compositionality works.

The key point of CSPTKs is that they can compute compositional trees integrating Distributional Compositional Semantics (DCS) operators into the kernel evaluation acting on both lexical leaves and non-terminal nodes. On the one side, SPTK already offer a modeling of lexical information, since they extend the similarity between tree structures allowing a smoothed function of node similarity, which makes them able to compare better trees which are semantically related even if their nodes and leaves differ.

However, SPTKs have an major limitation: they cannot consider compositional interactions between the lexical elements of the trees (i.e. between the words of the trees).

The meaning of the verb "to save" can be better captured only if we consider the verb *in composition* with the words it is refered to, namely "files" and "people". In this sense, CSPTK can better capture the role of more complex syntagmatic structures and compositions. Regarding the calculation of Δ, it is similar to SPTK, but the smoothing function is adapted according to [3, 4].

3 Related works

So far, only few studies have employed Tree Kernels in the field of AM. One of the first studies that mentioned the use of Tree Kernels in this field is Rooney 2012 [17], where kernels are used with Part-of-Speech tags and with sequences of words with the aim of detecting whether or not a sentence is related to an argumentative element (i.e. premise, conclusion, or both).

In 2015, Lippi and Torroni wrote an important study in which Tree Kernels are employed in a context-independent scenario, with the aim to detect argumentative claims [13]. In this case, the authors employed PTKs using constituency trees as tree representations. Similar approaches have been applied also in specific domains like the legal domain [14, 12] and the medical domain [15], where SSTK have been applied over constituency trees.

Outside the field of AM, Croce et Al. [7] focused on the combination of different Tree Kernels and tree representations. However, no study has proposed yet a comparison of this kind in AM, particularly in the classification of argumentative stances of support and opposition.

As already stated, this work is based on two previous studies, Liga 2019 [10] and Liga and Palmirani 2019 [11]. Interestingly, the two above-mentioned studies try to discriminate among stances of support and opposition that can be related to specific Argumentation Schemes [20]. While the first paper focused on the stances of support potentially related to the Argumentation Scheme from Expert Opinion (which is a specific kind of source-based Argumentation Scheme [9]), the second study has a particular focus on the argumentative stances related

to the Argumentation Schemes from Negative Consequences and the Slippery Slope argument [19]. In Liga 2019, Tree Kernels have been used in combination with TFIDF vectors to automatically discriminate between different kinds of argumentative support, while in Liga and Palmirani 2019 a similar methodology was employed to detect argumentative stances of opposition.

Importantly, these two studies explore the ability of different kinds of Tree Kernel to perform the tasks of classifying argumentative stances (the first study employs PTK [16], while the second study employs SPTK [6]) However, both studies employ the same kind of tree representation: namely, they only employ Grammatical Relation Centered Tree (GRCT representations). However, as already stated, there are other kinds of tree representation that may be employed in the same settings. For this reason, the present study reproduce the same scenarios of these two works by employing different kinds of tree representation to assess whether or not a particular kind of tree representation can be more suitable for the task of classifying argumentative stances of support and opposition.

To the best of our knowledge, no study of argumentative stance classification has so far presented a comparative analysis of the use of different kinds of tree representations and Tree Kernels. In particular, this study will compare the performance of the 5 tree representations described in Section 2.1 At the same time, we will combine these representations with the 5 Tree Kernels described in Section 2.2.

We will now describe the two settings of the present paper. The first one, related to the first study [10], will be defined Setting One; the second one, related to the second study [11], will be defined Setting Two. The experiments have been performed using the JAVA framework KeLP [8], and a 70/30 train-test split ratio.

4 Setting One

The first study combined two famous AM datasets in the same setting ([1] and [2], namely). We will refer to these datasets as DataOne [2] and DataTwo [1]. The aim of the study was to develop a series of classifiers able to differentiate between argumentative support coming from the opinion of an expert and argumentative support coming from studies or statistics.

Importantly, in this study, the ability of Tree Kernels to generalize over different data was explored by training the classifiers on one dataset and testing them on both datasets [10]. For this reason, we will consider Setting One as divided into two scenarios.

In the first scenario, all classifiers are trained on the training subset of DataOne (and tested not only on the testing subset of the same dataset, but also on the whole DataTwo dataset). In the second scenario, all classifiers are trained on the training subset of DataTwo (and tested not only on the testing subset of the same dataset, but also on the whole DataOne dataset).

As shown in the left side of Table 1, which reports the number of instances per class per dataset, the two datasets are quite balanced. The label expert has 372 and 311 instances in DataOne and DataTwo respectively; while the label

Study/statistics has 281 and 258 instances in DataOne and DataTwo respectively.

Table 1. Number of sentences in the two datasets, grouped by category group (left). Configuration of the kernel parameters for Setting One (right).

DataOne	n.
Expert	372
Study/statistics	281
Total	653

DataTwo	n.
Expert	311
Study/statistics	258
Total	569

Kernels	Decay factors	
STK	$\lambda = .3$	
SSTK	$\lambda = .2$	
PTK	$\lambda = .4$	$\mu = .4$
SPTK	$\lambda = .4$	$\mu = .4$
CSPTK	$\lambda = .4$	$\mu = .4$

4.1 Results for Setting One

The top part of Table 2 is referred to the first scenario, while bottom part is referred to the second scenario. For reason of space, compositional trees are reported jointly with their non-compositional counterpart: compositional tree representations (cGRCT and cLCT) should be thus considered related only with CSPTKs.

The experiments were performed using the configurations of the decay factors reported in the right side of Table 1[3].

Setting One - Scenario A Results of the top part of Table 2 related to GRCT/cGRCT (left sub-table, on the top), show a similar performance on the dataset DataOne (ranging from .85 to .87), but above all they show a growing degree of generalization over DataTwo: from a minimum of .72 (STK) to a maximum of .77 (CSPTK and SPTK). This shows that the degree of generalization increases when using PTKs and SPTKs compared to STKs and SSTKs.

A similar trend can be seen also with regard to LCT/cLCT (central sub-table), where the degree of generalization increase similarly from .72 (STK) to .77 (SPTK). However, performances are sliglty more polarized on DataOne (ranging from .84 to .88).

An even more polarized trend is reported on the sub-table on the right, related to LOCT. In this case, performance on the dataset DataOne range from .80 to .87 and this is the only case in which PTK outperform SPTK. Also the performances on DataTwo are more polarized compared to the other two sub-tables: for the LOCT sub-table scores range from .63 (STK) to .75 (SPTK).

[3] Decay factors are meant to penalise long tree fragments, in order to mitigate the risk that their size might excessively affect similarity scores. In this paper, λ is the vertical decay factor, while μ is the horizontal decay factor.

Setting One - Scenario B Results of bottom part of Table 2 related to GRCT/cGRCT (left sub-table), show that PTKs performed better than the other kernels: they reach a mean F1 score of .74 on DataTwo (which is the dataset on which the classifiers of this scenario have been trained) while all the other kernels range from .71 to .72. Also in this case, results over the other dataset (which in this scenario is DataOne) show a growing capability of generalization ranging from .79 (STK) to .84 (CSPTK and SPTK).

The trend over LCT/cLCT representations (central sub-table) show a similar picture: PTK is the kernel with the best performance over DataTwo (with a Mean F1 score of .72) while the other kernels range between .68 and .71. Also in this case, performances over DataOne show a growing trend ranging from .80 to .84.

Also in this scenario, the sub-table on the right, related to LOCT, is the most polarized one: results on DataTwo range from .64 to .69, with PTK outperforming SPTK; while results on DataOne range from .71 to .83, with SPTK showing again the best ability to generalize over the other dataset.

Table 2. Results for Scenario A (top) and Scenario B (bottom) of Setting One. Results report the mean F1 score of the binary classification ("Expert" vs "Study/Statistics").

SCENARIO A:

	GRCT/cGRCT		**LCT/cLCT**		**LOCT**	
Kernel	DataOne	DataTwo	DataOne	DataTwo	DataOne	DataTwo
STK	.86	.72	.84	.72	.80	.63
SSTK	.86	.74	.84	.73	.80	.66
PTK	.85	.76	.86	.75	**.87**	.73
SPTK	.86	**.77**	**.88**	**.77**	.83	**.75**
CSPTK	**.87**	**.77**	.87	.76		

SCENARIO B:

	GRCT/cGRCT		**LCT/cLCT**		**LOCT**	
Kernel	DataOne	DataTwo	DataOne	DataTwo	DataOne	DataTwo
STK	.79	.71	.80	.69	.71	.64
SSTK	.81	.71	.81	.68	.71	.64
PTK	.83	**.74**	.81	**.72**	.79	**.69**
SPTK	**.84**	.72	.83	.70	**.83**	.67
CSPTK	**.84**	.72	**.84**	.71		

5 Setting Two

In the second study [11], a similar approach has been employed on a dataset created ad hoc for the analysis of argumentative stances of opposition. The dataset has been created by extracting the comments of opposition that citizens wrote on the public website of Nevada Legislature against a bill aiming at regulating

euthanasia. The limitation of this study is that the annotation has not been accomplished yet and the number of instances per class is still unbalanced. However, an interesting aspect of this dataset is that a granular labelling system has been proposed in order to assess the ability of Tree Kernel classifiers to detect different kinds of argumentative opposition in a multi-class setting. The original setting presents four levels of granularity in which the dataset can be divided. In the present paper, we are going to select just the granularities with the best balance in the number of instances, namely granularity 2 and granularity 3.

For this reason, similarly to what has been done with Setting One, also Setting Two has been divided into two scenarios. In the first scenario, all classifiers are trained and tested considering three labels ("Slippery Slope", "Other" and "Testimony"). In the second one, all classifiers are trained and tested considering four labels ("Slippery Slope", "Other", "Judgements" and "Testimony"). The number of sentences grouped by class is listed on the left in Table 3, while the configurations of the decay factors used in the experiment are reported on the right in Table 3.

Table 3. Number of sentences depending on class and granularity (Left). Configuration of the kernel parameters for Setting Two (right).

Classes	Granularity 2	Granularity 3
Slippery Slope	82	82
Testimony	133	133
Judgements	*part of Other*	140
Other	423	283
Total	638	

Kernels	Decay factors	
STK	$\lambda = [.1 - .4]$	
SSTK	$\lambda = [.1 - .4]$	
PTK	$\lambda = [.3 - .4]$	$\mu = [.3 - .4]$
SPTK	$\lambda = [.3 - .4]$	$\mu = [.3 - .4]$
CSPTK	$\lambda = [.3 - .4]$	$\mu = [.3 - .4]$

5.1 Results for Setting Two

The top part of Table 4 is referred to the first scenario, while bottom part is referred to the second scenario. It is important to underline that this results should be observed by watching not only at the F1 scores (which can be misleading, since they are trained upwards by the results of the class "Other").

To partially overcome this problem, we should instead focus on the F1 scores of the single classes, as described in the following subsections.

Also in this case, compositional trees are reported jointly with their non-compositional counterpart and should be considered related only with CSPTKs.

Setting Two - Scenario A Results of the top part of Table 4 related to GRCT/cGRCT (left sub-table), show that the the Mean F1 score is achieved by the PTK classifier (.76). However, it should be remarked that STK and CSPTK seem to perform better on the class "Testimony" (reaching a F1 score of .71).

In the central sub-table, related to LCT/cLCT, the SPTK and the CSPTK are the best ones both in terms of Mean F1 score (.73-.74) and in terms of balance

between the scores for the "Slippery Slope" and "Testimony" classes (which is .59-.60 and .70-.71, respectively). Conversely, the STK, SSTK and PTK show nearly one decimal point less (floating between the values .50-.51).

Regarding the sub-table related to the LOCT representation, there is a clear superiority of the SPTK over the other kernels not only in terms of Mean F1 score (.75), but also in terms of balance between "Slippery Slope" and "Testimony" scores (.67 and .69, respectively). In fact, although PTK reaches .68 on the class "Slippery Slope", it stops at .58 on the class "Testimony".

Setting Two - Scenario B Regarding the second scenario of the Setting Two (the one considering four labels), results can be seen in the bottom part of Table 4. The sub-table on the left, related to GRCT/cGRCT, shows that performances of PTK and CSPTK classifiers are slightly better in terms of mean F1 score (.69). However, when one considers the results of the classes separately, one can see that the SSTK classifier shows better results in the classification of the "Slippery Slope" class (.67), while the CSPTK over the cGRCT representation shows better results in the classification of "Judgements" (.66) and "Testimony" (.74).

Regarding the central sub-table (LCT/cLCT), it seems that the CSPTK classifier is the one that has the best performance in terms of mean F1 (.68). It is also the classifier that reach the best performance in the classification of the class "Testimony" (.77). Regarding the class "Judgements", the best performances are achieved by the SSTK (.64) and the CSPTK (.63), while the class "Slippery Slope" is the one with the worst performances, with SPTK and CSPTK as best classifiers (stopping at .52).

Regarding the sub-table on the right (related to the LOCT representation), all mean F1 show a similar performance ranging from .60 to .63. Moreover, it can be seen that the performances for the class "Judgements" are the worst ones, where the best score is achieved by the STK (.54). On the other side, SSTK and SPTK achieve the best scores with the class "Slippery Slope" (.65), while the "Testimony" class has the SPTK as most performing classifier (reaching .66).

6 Discussion and Conclusions

From the results, some clear patterns can be observed. In general, it seems that PTKs and above all (C)SPTKs collect the best performances. This appears particularly evident in Setting One, when watching the numbers of Table 2 from the top to the bottom.

Another trend that can be seen from Setting One is the growing degree of generalization (always watching from the top to the bottom of the tables).

Finally, another interesting trend can be observed on Setting One, namely the growing degree of polarization, watching from the left (GRCT/cGRCT) to the right (LOCT), related to the scores between the most and less performing kernels in each column. It seems that the last table (LOCT) is the most polarized one, reaching up to .13 points of difference between the most and less performing

Table 4. Results for Scenario A (top) and B (bottom) of Setting Two. SS = "Slippery Slope", O = "Other", J = "Judgements", T = "Testimony". In bold the maximum F1 scores for SS, J and T.

SCENARIO A:

	GRCT/cGRCT			
Kernel	SS	O	T	$\overline{F1}$
STK	.61	.90	**.71**	.74
SSTK	.62	.91	.67	.73
PTK	**.67**	.91	.69	.76
SPTK	.58	.87	.68	.71
CSPTK	.63	.90	**.71**	.75

LCT/cLCT			
SS	O	T	$\overline{F1}$
.51	.90	.68	.70
.50	.92	**.71**	.71
.51	.90	.68	.70
.60	.89	.70	.73
.59	.91	**.71**	.74

LOCT			
SS	O	T	$\overline{F1}$
.60	.87	.61	.70
.62	.89	.63	.71
.68	.86	.58	.71
.67	.89	**.69**	.75

SCENARIO B:

	GRCT/cGRCT				
Kernel	SS	O	J	T	$\overline{F1}$
STK	.62	.76	.55	.71	.66
SSTK	**.67**	.80	.60	.63	.67
PTK	.64	.79	.61	.71	.69
SPTK	.55	.79	.63	.68	.67
CSPTK	.57	.79	**.66**	**.74**	.69

LCT/cLCT				
SS	O	J	T	$\overline{F1}$
.51	.76	.62	.69	.65
.50	.79	**.64**	.71	.66
.49	.80	.61	.75	.65
.52	.78	.58	.71	.65
.52	.79	.63	**.77**	.68

LOCT				
SS	O	J	T	$\overline{F1}$
.62	.76	**.54**	.61	.63
.65	.77	.51	.60	.63
.59	.72	.47	.62	.60
.65	.75	.47	**.66**	.63

score (i.e. in the column DataTwo, Scenario A; and in the column DataOne, in Scenario B).

To the best of our knowledge, this study is the first attempt to offer a comparative analysis of the combination of five tree representations and five tree kernels in the classification of argumentative stances of opposition and support.

A limitation of this work is that of being related to the specific data employed. The contribution of different tree representation should be assessed also in different scenarios and with different kinds of argumentative data.

Moreover, it is important to investigate the relation between the type of tree representation, the tree kernel function employed and the targeted argument to be classified. In other words, are there tree representation that can express better specific kinds of argument? Are there tree kernel functions that better calculate the similarities between these argumentative representations? This study suggests that the answer to these questions is positive, showing a first attempt of investigation in this direction.

References

1. Aharoni, E., Polnarov, A., Lavee, T., Hershcovich, D., Levy, R., Rinott, R., Gutfreund, D., Slonim, N.: A benchmark dataset for automatic detection of claims and evidence in the context of controversial topics. In: Proceedings of the First Workshop on Argumentation Mining. pp. 64–68 (2014)
2. Al Khatib, K., Wachsmuth, H., Kiesel, J., Hagen, M., Stein, B.: A news editorial corpus for mining argumentation strategies. In: Proceedings of COLING 2016, the

26th International Conference on Computational Linguistics: Technical Papers. pp. 3433–3443 (2016)
3. Annesi, P., Croce, D., Basili, R.: Towards compositional tree kernels. In: Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora. pp. 15–23 (2013)
4. Annesi, P., Croce, D., Basili, R.: Semantic compositionality in tree kernels. In: Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. pp. 1029–1038. ACM (2014)
5. Collins, M., Duffy, N.: Convolution kernels for natural language. In: Advances in neural information processing systems. pp. 625–632 (2002)
6. Croce, D., Moschitti, A., Basili, R.: Semantic convolution kernels over dependency trees: smoothed partial tree kernel. In: Proceedings of the 20th ACM international conference on Information and knowledge management. pp. 2013–2016. ACM (2011)
7. Croce, D., Moschitti, A., Basili, R.: Structured lexical similarity via convolution kernels on dependency trees. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. pp. 1034–1046. Association for Computational Linguistics (2011)
8. Filice, S., Castellucci, G., Croce, D., Basili, R.: Kelp: a kernel-based learning platform for natural language processing. Proceedings of ACL-IJCNLP 2015 System Demonstrations pp. 19–24 (2015)
9. Lawrence, J., Visser, J., Reed, C.: An online annotation assistant for argument schemes. In: Proceedings of the 13th Linguistic Annotation Workshop. pp. 100–107. Association for Computational Linguistics (2019)
10. Liga, D.: Argumentative evidences classification and argument scheme detection using tree kernels. In: Proceedings of the 6th Workshop on Argument Mining. pp. 92–97 (2019)
11. Liga, D., Palmirani, M.: Detecting "slippery slope" and other argumentative stances of opposition using tree kernels in monologic discourse. In: Rules and Reasoning. Third International Joint Conference, RuleML+RR 2019 (2019)
12. Lippi, M., Pałka, P., Contissa, G., Lagioia, F., Micklitz, H.W., Sartor, G., Torroni, P.: Claudette: an automated detector of potentially unfair clauses in online terms of service. Artificial Intelligence and Law pp. 1–23 (2018)
13. Lippi, M., Torroni, P.: Context-independent claim detection for argument mining. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
14. Lippi, M., Torroni, P.: Margot: A web server for argumentation mining. Expert Systems with Applications **65**, 292–303 (2016)
15. Mayer, T., Cabrio, E., Lippi, M., Torroni, P., Villata, S.: Argument mining on clinical trials. Computational Models of Argument: Proceedings of COMMA 2018 **305**, 137 (2018)
16. Moschitti, A.: Efficient convolution kernels for dependency and constituent syntactic trees. In: European Conference on Machine Learning. pp. 318–329. Springer (2006)
17. Rooney, N., Wang, H., Browne, F.: Applying kernel methods to argumentation mining. In: Twenty-Fifth International FLAIRS Conference (2012)
18. Vishwanathan, S., Smola, A.J., et al.: Fast kernels for string and tree matching. Kernel methods in computational biology **15**, 113–130 (2004)
19. Walton, D.: The basic slippery slope argument. Informal Logic **35**(3), 273–311 (2015)
20. Walton, D., Reed, C., Macagno, F.: Argumentation schemes. Cambridge University Press (2008)