Classifying argumentative stances of opposition using Tree Kernels *

A Preprint

Davide Liga *
CIRSFID
Alma Mater Studiorum - University of Bologna
via galliera, 3 - Bologna, Italy
davide.liga2@unibo.it

Monica Palmirani *
CIRSFID
Alma Mater Studiorum - University of Bologna
via galliera, 3 - Bologna, Italy
monica.palmirani@unibo.it

January 9, 2020

Abstract

The approach proposed in this study aims to classify argumentative oppositions. A major assumption of this work is that discriminating among different argumentative stances of support and opposition can facilitate the detection of Argument Schemes. While using Tree Kernels for classification problems can be useful in many Argument Mining sub-tasks, this work focuses on the classification of opposition stances. We show that Tree Kernels can be successfully used (alone or in combination with traditional textual vectorizations) to discriminate between different stances of opposition without requiring highly engineered features. Moreover, this study compare the results of Tree Kernels classifiers with the results of classifiers which use traditional features such as TFIDF and n-grams. This comparison shows that Tree Kernel classifiers can outperform TFIDF and n-grams classifiers.

***K*eywords** argument mining · tree kernels · natural language understanding · argument schemes

1 Introduction

Publicly open reviews on bills are used in many legal systems. In fact, in some systems, it is mandatory to open public reviews during the legislative process to encourage people's participation and engagement.

Interestingly, web portals for collecting opinions and comments from citizens are becoming more and more frequent, and the idea of supporting people participation and engagement has been embraced by many famous social media.

However, there are still important obstacles when trying to understand the argumentative threads of online debates, since they are often presented as a flat flow of textual interactions.

In this regard, it is extremely difficult for decision makers to extract useful information from debates with hundreds of posts. Similarly, it is difficult to extract a useful map of pros and cons from a given online debate.

In this sense, one of the most ambitious aims for the future of artificial intelligence is to automatically recognize arguments and counter-arguments in debates, along with argumentative fallacies.

This work presents a method which uses Tree Kernels to classify argumentative stances of opposition facilitating the detection of Argument Schemes such as the well-known "Slippery Slope" argument that produces polarization and emphasizes debates.

*This is a preprint version of a paper accepted for publication in the Proceedings of the 2019 2nd International Conference on Algorithms, Computing and Artificial Intelligence, DOI: https://doi.org/10.1145/3377713.3377717

2 Methodology

2.1 The Argument Mining Pipeline

The main target of Argument Mining (AM) is to analyze arguments, including their components and the relations connecting these components [1][2]. With an increasing number of works written and the interest of important private actors like IBM, this field has attracted a growing attention in the last few years [1], achieving important results and applications. These applications have been successfully implemented in a wide range of domains, since AM is physiologically multidisciplinary and facilitates cooperation among fields (e.g. Information Extraction, Knowledge Representation, Legal Reasoning, Sentiment Analysis). Importantly, being language the main means by which humans express their arguments, there is a close relation between AM and Natural Language Processing. Also, AM is closely related to Opinion Mining, with the latter trying to detect *what* people say and the former trying to understand *why*[3].

Lippi and Torroni [2] describe AM as a pipeline composed of three steps: the first step is the identification of argumentative data (which must be distinguished from non-argumentative data); the second step is the detection of the boundaries of argumentative components; the third step consist of predicting the relations among argumentative units and among arguments. Importantly, the last two steps strictly depend on the underlying argumentative model, e.g. the most frequently used two-role model proposed by Walton [4] (which considers argumentative units as "claim" and "premise"), or the more complex five-role model proposed by Toulmin [5] (which considers fact, warrant, backing, rebuttal and qualified claim).

Cabrio and Villata [1] proposed a simpler two-step pipeline, which is the one that we will refer to in this work. In their pipeline, the first step is the identification of arguments, which involves not only the differentiation between argumentative and non-argumentative data but also the identification of the roles of argumentative components (claims, premises, etc.) and their boundaries. The second step involves the prediction of the heterogeneous nature of argument relations (e.g., *supports*, *attacks*) and the connection between premises/evidences and conclusions/claims.

Ideally, after the above mentioned steps, a last phase can be that of fitting the map of argumentative components into an Argument Scheme (e.g., argument from Analogy, "Slippery Slope" argument, argument from Example).

From the one side, this work tries to classify argumentative stances of opposition. On the other side, it tries to facilitate the detection of those argumentative stances whose classification is more likely to be related to a specific Argument Scheme. Particularly, we targeted the well-known "Slippery slope" argument and we evaluated the ability of Tree Kernel methods to distinguish this scheme from other kinds of opposition stance.

2.2 Tree Kernels Methods

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. The 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 above $\langle ., . \rangle_{\mathcal{V}}$ 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 a binary classification task with a training dataset 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.

Since Tree Kernels belong to the family of kernel methods, they can be considered a *similarity measure* too. In particular, they are designed to calculate similarities between tree-structured documents.

Importantly, there are different kinds of Tree Kernel functions, which operate on different segments of the tree-structured documents. In fact, different TK functions make calculations by watching at different substructures of the given tree-structured data. In this study, data was segmented into Partial Trees (PTs) fragments, where each node is composed of any possible sub-tree, partial or not. The reason for this choice is that PTs are able to provide a high generalization [6].

The resulting function, called Partial Tree Kernel (PTK), can be calculated as follows [6]:

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

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. More information about tree fragments can be found in Nguyen et al. [7] and Moschitti [6].

Another important aspect is the selection of the kind of tree structure that will represent the textual data. A description of different kinds of tree representation is presented in Croce et al. [8]. In this work, we represented text as Grammatical Relation Centered Trees (GRCTs), which take into account not only Part-of-Speach Tags but also lexical terms.

As we will try to show in this work, there are different reasons for using Tree Kernels. On the one side, it seems that they can outperform traditional features, on the other side they can keep a high degree of generalization leveraging structural information.

3 Related Works

This work presents an approach which uses Tree Kernels methods to classify opposition stances. This method is also presented as a way to facilitate the detection of Argument Schemes, one of the most complex sub-task in AM.

Noticeably, only few studies tried to detect Argument Schemes. In this regard, Feng and Hirst [9] managed to achieve an accuracy ranging from 63 to 91% in one-against-others classification and 80-94% in pairwise classification. Some years later, Lawrence and Reed [10] increased the previous performances, achieving F-scores ranging from 0.78 to 0.91. However, since the task of detecting Argument Schemes is very complex, the two above-mentioned works deployed a set of highly engineered features. The aim of this study is to give a further contribution in this part of the AM pipeline, showing a possible method for Argument Scheme discrimination which is also able to preserve high levels of generalization without requiring highly engineered features.

TKs have already been used successfully in question answering [11], metaphor identification [12], semantic role labelling [13] and other NLP-related task. Although results of TKs methods have been strongly encouraging in all the above mentioned tasks, showing the ability of TKs to perform well, their application in AM has been relatively limited. One of the first uses in AM was proposed by Rooney et al. [14], who combined the use of TKs with Part-of-Speech tags. However, it was only after three years that somebody underlined the potential advantages of deploying TKs in the argument detection sub-task (which is the first step of the AM pipeline) [2]. One year later, the same authors presented a web application which uses TKs and is capable of extracting arguments from text automatically [15]. Still today, this is one of the few existing attempts to create a complete argument extraction tool.

This approach is the continuation of two previous works ([16] [17]), which aimed at discriminating among different kinds of argumentative stances of support and opposition. These two works are an attempt to give a contribution to the AM pipeline finding a working methodology capable to discriminate among stances of support/opposition by using Tree Kernels. The underlining assumption is that being able to classify different kinds of support and opposition is a crucial aspect also in the discrimination of different Argument Schemes.

In the present paper, our previously achieved findings will be extended, with a deeper analysis which involves the performances of twenty classifiers.

4 Corpus and Annotation

The analyzed corpus is composed of a group of 638 annotated sentences gathered from public available data. The annotation process is still ongoing under the supervision of experts of domain and our aim is to further extend the amount of annotated sentences. Particularly, these sentences have been extracted from the opinion of voters in the "Opinion Poll" section available in the official website of Nevada Legislature. More specifically, these sentences are taken from the opinion against the Senate Bill 165(SB165), about Euthanasia. Each comment of opposition against SB165 has been segmented into sentences using an automatic sentence segmentation tool.

After a preliminary empirical analysis, each sentence of the corpus has been manually annotated following an annotation scheme which is designed to allow different degrees of granularity in the classification process. Table 1 shows the list of classes with some examples, while Table 2 describes how these classes have been grouped in super-classes to

Table 1: The annotation classes with some examples.

Classes	**Examples**
SLIPPERY SLOPE	- *This would turn physicians into legal murderers.*
JUDGEMENT SIMPLE	- *This bill is terrible.*
JUDGEMENT MORAL	- *This bill is an affront to human dignity.*
MORAL ASSUMPTIONS	- *Only God should decide when a person is supposed to die.* - *Being a Christian, I cannot accept this bill.* - *This is totally against the Hippocratic Oath!*
STUDY STATISTICS	- *Our country already experienced 20% increase of suicide rate.*
ANECDOTAL *(PERSONAL EXPERIENCE)* *(NOT PERSONAL EXPERIENCE)*	- *The bible says that this is wrong.* - *My husband struggled a lot of years and* [...] - *In Oregon this bill created the chaos.*
OTHER/NONE	All the sentences that does not belong to the above classes

Table 2: The granularity levels and the grouping options.

<table>
<tr><th>Granularity 1</th><th>Granularity 2</th><th>Granularity 3</th><th>Granularity 4</th></tr>
<tr><td>SLIPPERY SLOPE</td><td>SLIPPERY SLOPE</td><td>SLIPPERY SLOPE</td><td>SLIPPERY SLOPE</td></tr>
<tr><td rowspan="5">OTHER/NONE</td><td rowspan="2">TESTIMONY</td><td rowspan="2">TESTIMONY</td><td>ANECDOTAL</td></tr>
<tr><td>STUDY STATISTICS</td></tr>
<tr><td rowspan="3">OTHER/NONE</td><td rowspan="2">JUDGEMENTS MORAL</td><td>JUDGEMENTS (simple + moral)</td></tr>
<tr><td>MORAL ASSUMPTIONS</td></tr>
<tr><td>OTHER/NONE</td><td>OTHER/NONE</td></tr>
</table>

create different levels of granularity. Thanks to this flexible annotation, the ability of TKs to perform fine-grained differentiation at each level of granularity has been tested.

So far, the classes PERSONAL EXPERIENCE, NOT PERSONAL EXPERIENCE, JUDGEMENTS SIMPLE and JUDGEMENT MORAL have not been used, but they could be useful when the annotation process will be completed and the corpus will be expanded.

As can be seen in Table 2, the first level is the least granular, since it discriminates between just two categories: SLIPPERY SLOPE sentences and the rest of classes. The second level is more granular, since it discriminates among three categories: SLIPPERY SLOPE, TESTIMONY, OTHER/NONE. The third level also involves JUDGEMENTS MORAL. The fourth level is the most granular since it discriminates among six categories: SLIPPERY SLOPE, ANECDOTAL, STUDY STATISTICS, JUDGEMENTS, MORAL ASSUMPTIONS, OTHER/NONE.

Importantly, during the annotation process we aimed to find out how people justify their opposition stances in a monologic debating context. In other words, the selected classes are the product of our empirical analysis on how people express their opposition. Since the focus of this annotation is *why* people are expressing a stance of opposition, all those comments which do not give any explanation for the opposition stance have been considered as part of the class OTHER/NONE (e.g. exhortations like “Please, vote no!”).

The number of sentences grouped by class is listed in Table 3.

5 The Experiment

For each sentence of the corpus a Grammatical Relation Centered Tree (GRCT) representation was created along with a TFIDF vectorization. More precisely, we attempted three different TFIDF vectorizations considering monograms, 2-grams and 3-grams, in order to assess the effects of n-grams on the results.

In other words, each labelled document in the corpus has two typology of “representation”: GRCT and TFIDF. For example, the sentence *“This is a slippery slope.”* can be represented as the GCRT in Figure 1 and can have a TFIDF

Table 3: Number of sentences depending on class and granularity.

<table>
<tr><th>Classes</th><th>Gr.4</th><th>Gr.3</th><th>Gr.2</th><th>Gr.1</th></tr>
<tr><td>SLIPPERY SLOPE</td><td colspan="4">82</td></tr>
<tr><td>STUDY STATISTICS</td><td>26</td><td colspan="2" rowspan="2">133</td><td rowspan="5">556</td></tr>
<tr><td>ANECDOTAL
(PERSONAL EXPERIENCE)
(NOT PERSONAL EXPERIENCE)</td><td>107</td></tr>
<tr><td>JUDGEMENT SIMPLE
JUDGEMENT MORAL</td><td>54</td><td rowspan="2">140</td><td rowspan="3">423</td></tr>
<tr><td>MORAL ASSUMPTIONS</td><td>86</td></tr>
<tr><td>OTHER/NONE</td><td colspan="2">283</td></tr>
</table>

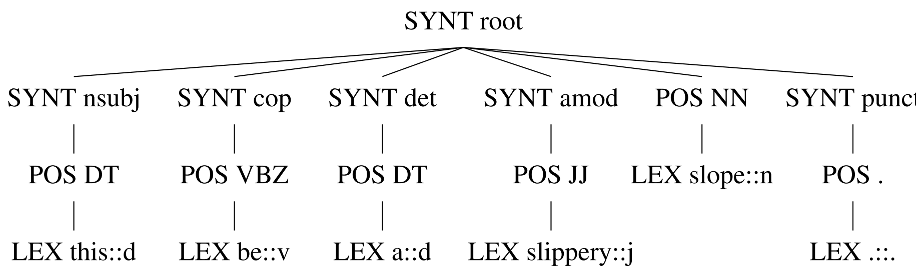


Figure 1: The GCRT representation for the sentence *"This is a slippery slope."*

vectorial representation like one of those in Figure 2, which shows the monogram, 2-grams and 3-grams. This results in three combinations at each level of granularity and, thus, in a total of twelve possible combinations.

All these classifiers were trained on the GRCT and TFIDF representations by using KeLP [18]. This operation was performed by dividing the corpus of 638 sentences into a test set of 191 sentences and a training set of 446 sentences and by using a One-vs-All classification, which is one of the most common approach for multi-class problems. Noticeably, KeLP allows to combine multiple kernel functions. In this work, the classification algorithm was built as a combination of a Linear Kernel and a Partial Tree Kernel (PTK) [6], with the first kernel related to the TFIDF vectors and the second kernel related to the GRCT representations. More details on kernel combinations can be found in Shawe-Taylor and Cristianini [19].

An important contribution of this study is that other two classifiers have been added at each level of granularity to better understand the real contribution of TKs and TFIDF. The first considers just monograms, the second considers just TK (monograms were preferred to other n-grams simply because of their better performances).

6 Results

Table 4 shows the resulting scores for each classifier, grouped by granularity. Since we want to show the non-triviality of the proposed task, we added the performance of a stratified baseline. The stratified method has been chosen because it produces better results, compared to a majority baseline, at all levels of granularity and because it reflects the classes' distribution in the training set.

Figure 2: An example of TFIDF representation of a sentence (monograms and n-grams).

Monograms:
```
is:0.2872 this:0.2944 slippery:0.6445 slope:0.6445
```
2-grams:
```
is:0.1913 this:0.1961 this_is:0.3442 slippery:0.4293 slope:0.4293 is_slippery:0.4962
slippery_slope:0.4374
```
3-grams:
```
is:0.1543 this:0.1581 this_is:0.2776 slippery:0.3462 slope:0.3462 is_slippery:0.4002
slippery_slope:0.3528 this_is_slippery:0.4351 is_slippery_slope:0.4002
```

Table 4: The scores of the classifiers grouped by granularity (P = Precision, R = Recall, F1 = F1 score). Close to the class name, the number of instances is specified. SS = SLIPPERY SLOPE, O = OTHER, T = TESTIMONY, JM = JUDGEMENTS AND MORAL, ST = STUDY STATISTICS, A = ANECDOTAL, MA = MORAL ASSUMPTIONS, J = JUDGEMENTS.

Classes	1grams			TK			TK + 1grams			TK + 2grams			TK + 3grams			Stratified
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	baseline
Granularity 1																
SS (82)	0.77	0.52	0.62	0.71	0.48	0.57	0.75	0.60	0.67	0.90	0.36	0.51	0.89	0.32	0.47	
O (556)	0.93	0.98	0.95	0.93	0.97	0.95	0.94	0.97	0.96	0.91	0.99	0.95	0.91	0.99	0.95	
Mean F1		0.79			0.76		—>	**0.81**	<—		0.73			0.71		0.54
Granularity 2																
SS (82)	0.71	0.68	0.69	0.82	0.56	0.67	0.76	0.64	0.70	0.71	0.40	0.51	0.83	0.40	0.54	
T (133)	0.70	0.68	0.69	0.68	0.74	0.70	0.66	0.77	0.71	0.63	0.74	0.68	0.68	0.74	0.70	
O (423)	0.89	0.90	0.89	0.90	0.93	0.91	0.92	0.91	0.91	0.88	0.92	0.90	0.88	0.95	0.91	
Mean F1		0.76			0.76		—>	**0.77**	<—		0.70			0.72		0.31
Granularity 3																
SS (82)	0.70	0.64	0.67	0.76	0.52	0.62	0.73	0.64	0.68	0.65	0.44	0.52	0.69	0.40	0.54	
T (133)	0.68	0.79	0.73	0.61	0.82	0.70	0.63	0.85	0.72	0.60	0.82	0.69	0.60	0.82	0.69	
JM (140)	0.62	0.49	0.55	0.76	0.53	0.62	0.64	0.53	0.58	0.71	0.53	0.61	0.74	0.55	0.63	
O (283)	0.70	0.75	0.73	0.76	0.85	0.80	0.76	0.75	0.76	0.76	0.82	0.79	0.75	0.82	0.79	
Mean F1		0.67		—>	**0.69**	<—	—>	**0.69**	<—		0.65			0.66		0.20
Granularity 4																
SS (82)	0.67	0.72	0.69	0.67	0.56	0.61	0.64	0.72	0.68	0.65	0.68	0.67	0.63	0.61	0.61	
A (107)	0.56	0.77	0.65	0.54	0.85	0.66	0.55	0.88	0.68	0.59	0.88	0.71	0.55	0.85	0.67	
ST (26)	1.00	0.13	0.22	0.33	0.13	0.18	1.00	0.13	0.22	1.00	0.13	0.22	1.00	0.13	0.22	
J (54)	0.78	0.37	0.50	0.33	0.11	0.16	0.78	0.37	0.50	0.88	0.37	0.52	1.00	0.37	0.54	
MA (86)	0.50	0.36	0.42	0.63	0.36	0.45	0.52	0.43	0.47	0.68	0.54	0.60	0.70	0.50	0.58	
O (283)	0.66	0.76	0.71	0.70	0.86	0.77	0.76	0.76	0.76	0.76	0.85	0.80	0.74	0.86	0.79	
Mean F1		0.53			0.47			0.54		—>	**0.59**	<—		0.57		0.21

As expected, results show that F1 scores are lower at higher granularity. Importantly, we remark that classifiers 2 and 3 are probably the best ones in terms of balance among results and number of instances per class. Moreover, since the class OTHER is often responsible for the increase of the mean F1 value, it is important to consider not just the Mean F1 score. For example, Figure 3 shows the results for the mean F1 scores and the results for the F1 scores related to different classes (and sub-classes), particularly SLIPPERY SLOPE, which is the main target of this study. In this way it is possible to have a better understanding of the performance of the classifiers. The same Figure show the decrease of the mean F1 scores at higher granularity.

Finally, results show that TK-only classifiers can be equal or better than monograms (at granularity 2 and 3, respectively). Although monograms outperforms TK-only classifiers at granularity 1 and 4, the combination TK+n-grams is always the most performing.

The results for each level of granularity will be now discussed.

6.1 Results for Granularity 1

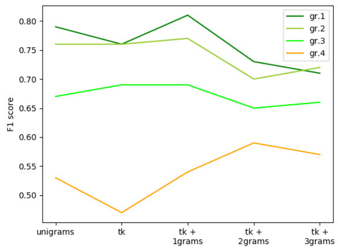
The classifiers at granularity level 1 show that the best performance can be achieved by combining the GRCT Tree Kernel with monograms. In fact, although the monograms classifier outperforms the TK-only classifier, with a mean F1 score of 0.79 and 0.76 respectively, the combination of TK and monograms outperform all the other combinations, achieving a mean F1 score of 0.81.

However, the problem of these classifiers is that the number of SLIPPERY SLOPE instances (82) is too little compared to the instances of OTHER (556). In fact, the good result is mostly due to the F1 score related to the class OTHER (0.96), while the F1 score related to the SLIPPERY SLOPE class, our main target, is at 0.67.

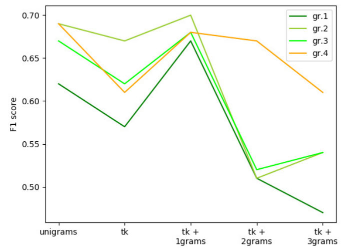
6.2 Results for Granularity 2

The classifiers with granularity level 2 achieved more encouraging results. Interestingly, the results of the TK-only classifier and the monograms classifier are equal in this case, with a mean F1 score of 0.76. Again, the best results is achieved by combining TK and monograms, with a mean F1 score 0.77. In this case, the F1 score for SLIPPERY

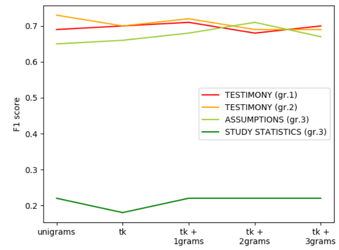
Figure 3: A comparison between classes' scores and Mean F1 scores (a,b,c,d,e) and the decrease of F1 over granularity (f).



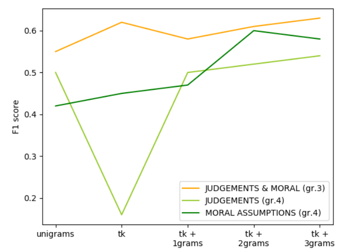
(a) Mean F1 score



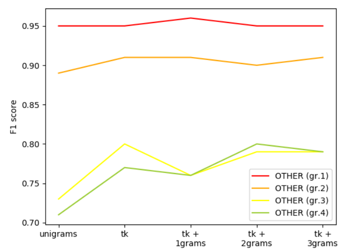
(b) F1 scores for SLIPPERY SLOPE



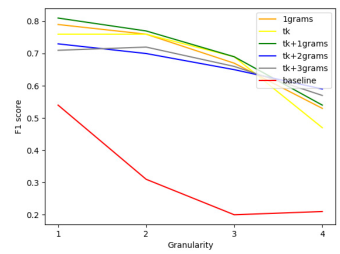
(c) TESTIMONY and its sub-classes



(d) JUDGEMENTS AND M. and its sub-classes



(e) Scores for the class OTHER



(f) Mean F1 scores decrease

SLOPE reaches 0.70, while the score for TESTIMONY is 0.71. Even though the instances of OTHER are still too many compared to the number of instances of the other two classes, the numbers of instances is more balanced compared to granularity 1.

6.3 Results for Granularity 3

The granularity level 3 is maybe the one with the best balance in terms of number of instances. The classifiers of this group achieved a mean F1 score ranging from 0.65 to 0.69. Interestingly, the TK-only classifier outperformed the monograms classifier, achieving the best performance together with the TK+monograms combination. This means that TK can outperform traditional TFIDF representations.

However, for the purposes of this work, the TK+monograms combination is still preferred, since it produce a better performance on the SLIPPERY SLOPE and TESTIMONY classes.

6.4 Results for Granularity 4

The last group of classifiers is the most granular one. The main problem of this classifiers is that they were trained on a small number of instances per class, especially the classes STUDY STATISTICS and JUDGEMENTS, which have just 26 and 54 instances respectively. On the other side, an important achievement of this group of classifiers is that they produce an F1 score for the class SLIPPERY SLOPE which is comparable or superior to granularity 3, achieving good results also with the class ANECDOTAL.

7 Conclusions and Future Work

This study shows that Tree Kenels can outperform traditional features such as TFIDF. Importantly, we wanted to remark that one of the main advantages of Tree Kernels is the possibility of leveraging structural information while preserving a high generalization.

The proposed method shows the ability of Tree Kernels to classify different kinds of opposition stance with relatively good results and without using highly engineered features, while at the same time presenting a working methodology for Argument Scheme discrimination.

The experiment was performed on a corpus of 638 short comments expressing opposition against the Nevada's Senate Bill 165, which aims to regulate Euthanasia .

Although results are encouraging, especially with the second and third groups of classifiers, there are still some obstacles when trying to deepen the degree of granularity. In the future, creating a chain of classifiers could help solve this problem, with a gradual and more complex advancement into granularity.

Moreover, we are working on the enlargement of the annotated data, since the imbalanced number of instances per class is a significant obstacle towards the achievement of better scores.

Despite the above-mentioned limitations, the present work shows that TKs can differentiate between argumentative stances and recognize stances that are related to the "Slippery Slope" argument. Still, the combination TK+n-grams outperforms the other classifiers.

8 Acknowledgments

This work was partially supported by the European Union's Horizon 2020 research and innovation programme under the MSCA grant agreement No 690974 "MIREL: MIning and REasoning with Legal texts".

References

[1] Elena Cabrio and Serena Villata. Five years of argument mining: a data-driven analysis. In *IJCAI*, pages 5427–5433, 2018.

[2] Marco Lippi and Paolo Torroni. Argument mining: A machine learning perspective. In *International Workshop on Theory and Applications of Formal Argumentation*, pages 163–176. Springer, 2015.

[3] Ivan Habernal, Judith Eckle-Kohler, and Iryna Gurevych. Argumentation mining on the web from information seeking perspective. In *ArgNLP*, 2014.

[4] Douglas Walton, Christopher Reed, and Fabrizio Macagno. *Argumentation schemes*. Cambridge University Press, 2008.

[5] Stephen E Toulmin. *The uses of argument*. Cambridge university press, 2003.

[6] Alessandro Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *European Conference on Machine Learning*, pages 318–329. Springer, 2006.

[7] Truc-Vien T Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1378–1387. Association for Computational Linguistics, 2009.

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

[9] Vanessa Wei Feng and Graeme Hirst. Classifying arguments by scheme. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 987–996, 2011.

[10] John Lawrence and Chris Reed. Argument mining using argumentation scheme structures. In *COMMA*, pages 379–390, 2016.

[11] Simone Filice and Alessandro Moschitti. Learning pairwise patterns in community question answering. *Intelligenza Artificiale*, 12(2):49–65, 2018.

[12] Dirk Hovy, Shashank Shrivastava, Sujay Kumar Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. Identifying metaphorical word use with tree kernels. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 52–57, 2013.

[13] Alessandro Moschitti, Daniele Pighin, and Roberto Basili. Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224, 2008.

[14] Niall Rooney, Hui Wang, and Fiona Browne. Applying kernel methods to argumentation mining. In *Twenty-Fifth International FLAIRS Conference*, 2012.

[15] Marco Lippi and Paolo Torroni. Margot: A web server for argumentation mining. *Expert Systems with Applications*, 65:292–303, 2016.

[16] Davide Liga. Argumentative evidences classification and argument scheme detection using tree kernels. In *Proceedings of the 6th Workshop on Argument Mining*, pages 92–97, 2019.

[17] Davide Liga and Monica Palmirani. 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.

[18] Simone Filice, Giuseppe Castellucci, Danilo Croce, and Roberto Basili. Kelp: a kernel-based learning platform for natural language processing. *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 19–24, 2015.

[19] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.