Computer Science Department
University of Crete

Opinion Mining on Parliamentary Commentaries, using Machine Learning.

Moschonas Giannis, Smyrnaios Giorgos

Computer Science Department
UNIVERSITY OF CRETE
Heraklion, Greece 2015

Graduate thesis 2015

Opinion Mining on Parliamentary Commentaries, using Machine Learning.

Moschonas Giannis, Smyrnaios Giorgos

Computer Science Department
University of Crete

Computer Science Department
University of Crete
Heraklion, Greece 2015

Opinion Mining on Parliamentary
Commentaries, using
Machine Learning.

Supervisor: Name, Company or Department
Examiner: Name, Department

Graduate thesis 2015
Department of Computer Science
University of Crete
E-mail (Moschonas): jmoschon@csd.uoc.gr
E-mail (Smyrnaios): gsmyrneo@csd.uoc.gr

Typeset in LaTeX
Heraklion, Greece 2015

Moschonas Giannis, Smyrnaios Giorgos
Compute Science Department
University of Crete

Abstract

Opinion Mining is a field in Computer Science where a better association between Natural Language and Computers is attempted (since it is part of the wider Natural Language Processing) as well as an attempt to mine semantic information from writing. More precisely, in the wider field of “Natural Language Processing”, “Machine Translation”, “Named Entity Recognition and Disambigution”, “Sentiment Analysis” etc. are included. In this Thesis, a better approach to information extraction from plain texts is attempted. The texts contain commentaries of citizens on consultation of Laws issued by the Greek Government. Extraction of proposals and counterproposals that the writers of these texts make, as well as of the arguments they formulate, is attempted. Finally, the general opinion of the writer is extracted, which is summarised by a “Positive” or “Negative” characterisation, depending on the writer’s view, which comes of the while text. Mining of the above data is done entirely by analysing the texts through a three-stage analysis (which will be explained in detail in the following chapters of this Thesis) and by utilising techniques that are included in the wider spectrum of NLP (Information Retrieval, Part-of-Speech Tagging, Sentiment Analysis etc.). The results show that we can create realistic methods that can extract this type of information. This matter troubles the Scientific community because these methods can be put into use in a variety of other fields, apart from the field of Computer Science (e.g. Journalism, Politics etc.).

Keywords: argument extraction, sentiment, machine learning, suggestion extraction, POS Tagging, opinion mining, natural language processing.

Declaration of Authorship

We declare that this thesis titled, “Natural Languages Processing on Parliamentary Commentaries, using Machine Learning.” and the work presented in it are our own. We confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where we have consulted the published work of others, this is always clearly at-tributed.
- Where we have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely our own work.
- We have acknowledged all main sources of help.
- Where the thesis is based on work done by ourselfs jointly with others, we have made clear exactly what was done by others and what I have contributed ourselves.

Giannis Moschonas

Name Sign

Giorgos Smyrnaios

Name Sign

Acknowledgements

TODO

Name Familyname, Gothenburg, Month Year

Contents

List of Figures

List of Tables

1

Introduction

TODO

2

Background

TODO

2.1 Machine Learning

TODO

2.1.0.1 Categorization of Machine Learning Algorithms

TODO

2.1.0.2 Machine Learning Algorithms

TODO

2.2 Part-Of-Speech Tagging (POS-Tagging)

TODO

2.3 Information Retrieval

TODO

2.3.0.3 Information Retrieval Algorithms

TODO

2.4 Related Work

TODO

2.5 Equation

$$f(t) = \begin{cases} 1, & t < 1 \\ t^2 & t \geq 1 \end{cases} \tag{2.1}$$

3

Methods

In this chapter, we will thoroughly analyse the ways with which the three processing stages which were presented in the previous units, were implemented.

In order to describe in the best possible way the process that was followed, a detailed description of the dataset which was used will be given, as well as of the features that characterise it. Then, we will describe the relational database model which we used to store the information from the texts (dataset). Finally, for each one of the process stages, we will describe the methodology with which each matter was approached.

3.1 Preparing the Dataset

In this part of the methodology, the features of the used dataset (**??**) will be described in detail. Some information on the way of choosing data (**??**) will be described as well. Next, the way of data mining from the Greek Open Government platform[1] (**??**) and finally, the Entity-Relation Model (**??**) of the database which was used to store the data will also be described.

3.1.1 Dataset

As it has already been mentioned in some points of this Thesis, the data that were used have been taken from the Greek Open Government platform, which constitutes a platform of electronic consultation of citizens on texts, more specifically on laws and decrees that the Greek Government issues. These data are open and accessible to everyone.

In this section, the basic features of the studied texts will be described. The reason why this section comes first in this part of the methodology, is that the very nature of these texts (they are basically users' comments to the online service), created many problems in their analysis.

As it has already been mentioned, the texts that were studied feature several oddities, some of which made the process of analysing them difficult.

[1] `http://www.opengov.gr`

- Initially, the first that we can notice is that the length of the texts is relatively short. To be precise, it is rare for them to be longer than 3000 characters (approximately 200 words, 80% of the texts). The length of the text did not affect all the stages of the analysis. The biggest difficulty appeared in the effort to extract the degree of the writer's agreement with the initial text (more details will be given later).

- A second remark is the fact that the texts that were studied do not consist an official text. By the term "official", we want first to declare that the texts are made up of users' comments in an online service and secondly, that they contain many errors (spelling etc). This created many difficulties in the studying of these comments. The first difficulty had to do with the tools needed in order to conduct the overall analysis of each text. The basic idea was that the tools had to be tolerant when it came to errors, at least up to a degree.

  Some very usual errors are:
  1. spelling errors
  2. absence of some letters in a word
  3. letter transposition in a word
  4. use only of capital letters
  5. absence of punctuation
  6. wrong sentence separation (there was no gap after the dot)
  7. some more errors that will not be mentioned for ease of reference

- One more issue is that there are many times when syntactic structure errors are spotted. This problem is directly connected to the use of POS-Tagger for the syntactic analysis (parsing) of texts. This issue affects, to some degree, the extrapolation of arguments and of proposals and counter proposals that the user makes.

- Another feature is that the texts are entirely in Greek. This problem is more serious, because there are no tools which we needed at some point of the analysis, that support the Greek Language. Subsequently, as we will see later on, there was the need to resort to some compromising solutions.

- One last issue that is worth mentioning, which constitutes a more qualitative feature, at least in the whole of texts that were studied, is the fact that the majority of users who wrote a comment are "annoyed". This "annoyance" stems from the fact that the texts that are under discussion contain laws and presidential decrees that, essentially, lead to a decrease in public spending towards the citizens. This "annoyance" is noted almost in the entire dataset that we studied. The problem is that the texts in which the writers agree with the initial text are limited. As a result, this issue complicates the process of acknowledging, if the writer agrees with the initial text. Also another serious problem is that the "annoyance" causes the users to make several kind of mistakes (such as grammatical, syntactical etc.).

3.1.2 Choosing Set of Documents

To continue the process, we randomly chose five different bills that contained a significant amount of users' replies. Afterwards, we selected a few, trying to eliminate the replies that we did not want to process. For example:

- replies that only contained one sentence
- replies in greeklish

Next, we limited the dataset so as to contain a number of approximately two hundred replies. This total is the final dataset that was studied and on which the conclusions for all the processing stages were based.

3.1.3 Finalizing the Dataset

The last step for the creation of the final dataset was the data mining from the Greek Open Government platform[2] (the website for public consultation on laws). This process was simple enough, since the service provides the users with the option to locally store all the comments that have been posted for each law or decree. The data were in excel file format, providing for each comment the following meta-data:

- the Law Article which was commented
- an id for each comment
- the name of the user-commenter
- the date

These data were later stored in the database which was created for the storage of data that were collected in all the stages of processing.

3.1.4 Database

In order to store the database, an MySQL[3] Database was created, whose Entity-Relation Model[4] can be presented in the following layout.

[2] `http://www.opengov.gr`

[3] `https://www.mysql.com`

[4] `https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model`

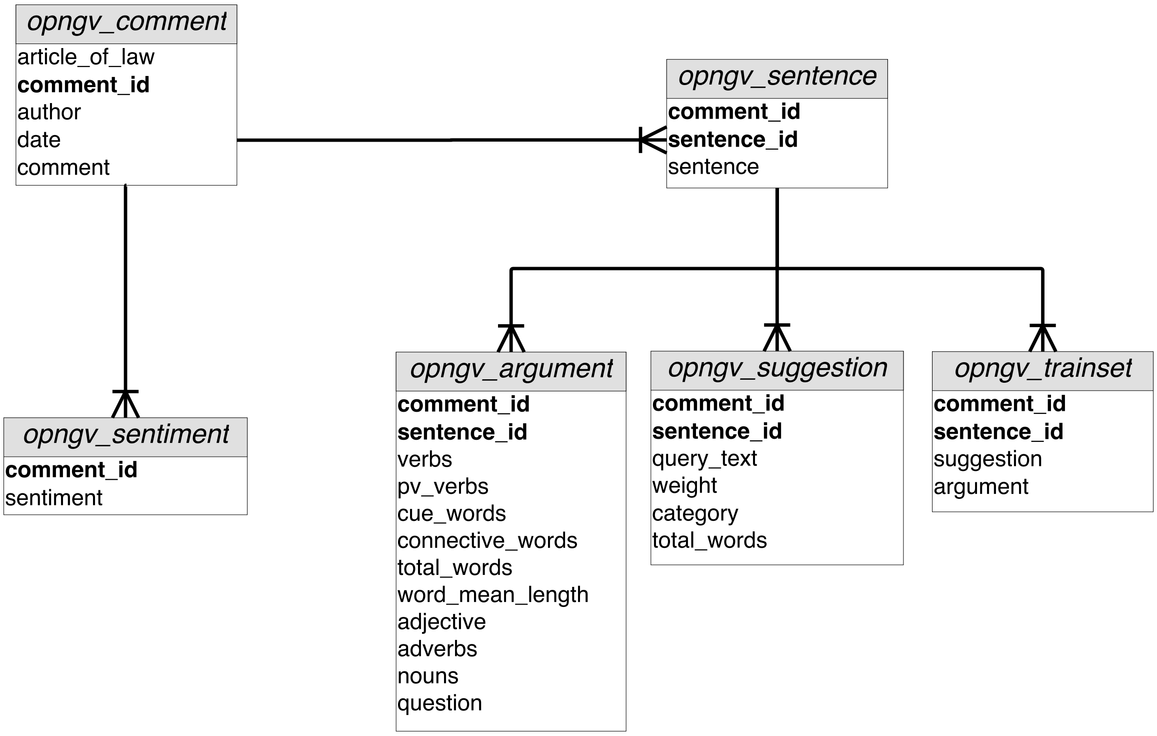


Figure 3.1: Entity - Relation Model

In the above layout, we can see the Entity-Relation Model that was used for storing data. The keys for each table of data have been marked with bold.

3.1.5 Building a Trainset

One last element that deals with the chapter on dataset, is characterising a total of sentences if each one of them contains an Argument or a Suggestion. We should note here that sentence separation will be analysed thoroughly later (**??**). The Train set that we created, as we will see in the chapters that follow, is needed so that the machine learning algorithms can become train, as well as to achieve a better evaluation. The set of sentences that was created contains approximately one thousand sentences.

3.2 Argument Extraction

At this point, the process with which the Argument Extraction was carried out on the whole of the texts will be described. The process is based on three stages:

- Selecting Argument Markers (??).
- POS-Tagging the set of documents (??).
- Applying machine learning to the Argument Markers (??).

Each one of the three stages will be explained right away.

3.2.1 Selecting Argument Markers

This step, in essence, constitutes the selection process of certain “criteria” - “Argument Markers” that will help us define what an argument is. Essentially, this process constituted the hardest part in the whole Argument Extraction stage.

Having studied enough from Related Work (which was mentioned to a great extent above) in the field of the Opinion Mining, it was clear that in order to extract the arguments from the texts that we had available we would have to apply some Machine Learning process. Therefore, it is now clearer that at this step (??), we had to define some parameters which will define with a relative clarity whether a sentence is an argument or not.

But before we get in the process of searching for these variables, we had one more difficulty to face. Even though it sounds relatively easy to recognise the arguments in a text, in reality it was a rather difficult process. There was great difficulty for the whole team that is working on this Project to agree on whether a sentence contains an argument or not. After a lot of discussion, we ended up with the following definition:

Sentence is likely to contain an argument if it contains the following markers:

- *The sentence provides a context clue from which we can interpret that the writer expresses an opinion.*
- *The sentence is explanatory, which means that the writer wants either to further explain or support an opinion.*

If there is any of the above markers in a sentence, then this sentence can be characterised as an argument (Argumentative sentence). Even with the above two conventions, in some cases it is still difficult to determine whether a sentence is an argument. More precisely, another convention has been set, with which we made an effort to exclude the sentences that are interrogative. The reason is that usually, interrogative sentences are likely to express irony. In our dataset this was actually very usual.

Having set the above conventions, we studied a total of Argumentative Sentences, in order to be able to track down “Argument Markers”. In essence, we looked for parts of speech (for example the number of verbs, number of adjectives and more), as well as other variables that often appear in this type of sentences. These variables were to be used in the next steps and especially in the step where the “Machine Learning” process is applied

From the study that was conducted, also combining Related work, we ended up with the following variables:

- **Number of Verbs:** In many studies which have been conducted in the past, it was noted that verbs are closely linked to arguments. To be precise, verbs express action. Another characteristic is that verbs syntactically compose a sentence or, a little more arbitrarily, verbs enrich a sentence. Consequently, it was noted that the sentences which contain arguments usually have a larger number of verbs.
- **Number of Verbs in Passive Voice:** The bibliography that we had available mentions that usually the verbs that are in the passive voice are used when someone claims something concerning a particular matter. In our dataset this was not so clear. However, in the measurements done, we noted that most sentences containing an argument also contain, to a great extent, verbs that are in the passive voice.
- **Key Words:** This variable refers to the number of words that were traced in a group of words that we created. This group contains words that are used when someone tries to explain or support an opinion. For example, in this group there are the words consider, believe, admit, suppose, think, must, consequently, since, until, because, namely.
- **Number of Connective Words:** This variable counts the number of linking words that were found in a sentence. In essence, this number has a direct relation to the next variable which shows the total number of words. The idea of counting the length of a sentence lies to the fact that usually, longer sentences are more likely to contain arguments. Especially when the studied dataset contains texts with arguments and political discussion.
- **Total Number of Words:** respectively with the previous variable.
- **Average Number of Letters in a Word:** This variable came up from the bibliography that we had available. It has been noted that this variable, especially in texts that contain political discussion, can help in a significant way to track arguments.
- **Number of Adjectives:** It is the number of arguments that were found in a sentence. The logic behind this variable is the same with the “Number of Verbs” variable logic.
- **Number of Adverbs:** It came up after studying the bibliography and it states the number of adverbs in a sentence.
- **Number of Nouns:** The role of this variable is equivalent to the “Number of Adjectives” variable.

- **A Boolean Variable that States Whether a Sentence is Interrogative:** The need for this variable was explained thoroughly above.

3.2.2 POS-Tagging

In this part we will analyse the second step in the process of argument extraction from a text. Unlike the previous step (**??**), the execution of this step is essential for every new step we wish to analyse.

As it has already been mentioned above (**??**), the POS-Tagging procedure has as goal to analyse the grammar of the text, as well as to extract an output which will contain a recognition of what part of speech each word is. Clearly, this process is very important for Argument Extraction because it constitutes the way with which all parts of speech will be detected.

3.2.2.1 POS-Tagger

In this Thesis, "ILSP POS-Tagger[5]" was used, which was created by the "Institute of Language and Speech processing[6]". Also the POS-Tagger is used in order to split the text into sentences. This feature appears in the next section (**??**) in which we can see the output that the Tagger produces.

3.2.2.2 POS-Tagger Output

From the outputs supported by POS-Tagger that we had available, we have chosen the "xceslemma" option, with which-apart from POS-Tagging, Lemmatization can also be accomplished. We will need the latter in the next unit that we are going to study. The output given is in XML format. An example can be seen below:

[5] `http://nlp.ilsp.gr/soaplab2-axis/`

[6] `http://www.ilsp.gr/en`

```xml
<?xml version='1.0' encoding='UTF−8'?>
<cesDoc xmlns="http://www.xces.org/schema/2003" version="0.4">
  <text>
    <body>
      <p id="p1">
        <s id="s1">
          <t id="t1" word="..." tag="AtDfNeSgNm" lemma="..."/>
          <t id="t2" word="..." tag="RgFwOr" lemma="..."/>
          <t id="t3" word="..." tag="PnReNe03SgNmXx" lemma="..."/>
          <t id="t4" word="..." tag="VbMnIdPr03SgXxIpPvXx" lemma="..."/>
          <t id="t5" word="..." tag="VbMnIdPr03SgXxIpPvXx" lemma="..."/>
          <t id="t6" word="..." tag="AsPpSp" lemma="..."/>
          <t id="t7" word="..." tag="NoCmFeSgAc" lemma="..."/>
          <t id="t8" word="..." tag="RgFwOr" lemma="..."/>
          <t id="t9" word="..." tag="PTERM_P" lemma="..."/>
        </s>
      </p>
    </body>
  </text>
</cesDoc>
```

The text that was given as input was *"The output which is given is in XML format"*. We note that for every word of the text, the following information is given:

- **Id:** an auto increment identifier given in every word of the text
- **Word:** the initial word of the text
- **Tag:** the Grammar concerning the particular word. For example, verbs start with "Vb" and nouns with "No". The whole tagset can be found here: `http://nlp.ilsp.gr/nlp/tagset_examples/tagset_en/`.
- **Lemma:** it is the dictionary entry of each word. For example, we note that the verb "given" has "give" as its lemma.

3.2.2.3 Parsing XML File

For the operation and data extraction from the XML file, DOM (Document Object Model[7]) was used, which is a platform for the representation and the interaction with objects in XML files. In this platform, the nodes of each file are organised in a tree form called the DOM tree. The objects in the DOM tree can be operated using the methods for objects. DOM's public interface is determined in Application Programming Interface.

3.2.2.4 Uploading to Database

Finally, the data from the Parsing of XML file (**??**) are stored in the Database and more precisely in the "opngv_argument" table. For storing, the following Query is

[7] `http://www.w3.org/DOM/`

used:

```
INSERT INTO opngv_argument VALUES (values..)
```

3.2.3 Apply Machine Learning

The last stage for argument retrieval consists of the application of a Machine Learning process. In order for this process to be accomplished, a train set and a test set should have been clearly set.

3.2.3.1 Selecting Train and Test Set

In this case in point, the train set is stored in the Data Base. More precisely, the train set retrieval from the Database is done with the following query:

```
SELECT
        opngv_argument.verbs,
        opngv_argument.pv_verbs,
        opngv_argument.cue_words,
        opngv_argument.connective_words,
        opngv_argument.total_words,
        opngv_argument.word_mean_length,
        opngv_argument.adjective,
        opngv_argument.adverbs,
        opngv_argument.noons,
        opngv_argument.question,
        opngv_trainset.Argument
FROM
        opngv_sentence
        INNER JOIN opngv_argument
                ON opngv_sentence.comment_id = opngv_argument.comment_id
                AND opngv_sentence.sentence_id = opngv_argument.sentence_id
        INNER JOIN opngv_trainset
                ON opngv_sentence.comment_id = opngv_trainset.comment_id
                AND opngv_sentence.sentence_id = opngv_trainset.sentence_id
```

Whereas the test set constitutes the total of the sentences of a text, which we wish to analyse. In essence, to make it clearer, two tables, which contain the same columns, are created every time. The train set table contains the data as described in units (**??**) and (**??**). The second table (test set), contains the data that came up from the process that was described in unit (**??**) for new texts.

3.2.3.2 Machine Learning Process

For the application of a Machine Learning Process, the Weka Framework "Weka 3: Data Mining Software in Java[8]" was used.

[8] http://www.cs.waikato.ac.nz/ml/weka/

"Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes."

3.2.3.3 Machine Learning Algorithms

To verify the accuracy of the Train set, we tried many algorithms:

- Logistic Regression[9]
- Native Bayes[10]
- Support Vector Machines[11]
- Random Forest[12]

The results will be analysed in the following units.

3.3 Suggestion Extraction

At this section we will describe the the procedure that we followed in order to achieve the Suggestion Extraction. The Procedure has five steps:

- Selecting Suggestion Markers (**??**)
- POS-Tagging & Lemmatization the set of Documents (**??**)
- Apply Information Retrieval Methods in order to find the suggestions (**??**)
- Adding additional features for the optimization of Machine Learning Process (**??**)
- Apply Machine Learning (**??**)

Each one from the above steps will be explained in detail, in the next sections.

3.3.1 Selecting Suggestion Markers

As in the previous chapter we have, at this stage, to set some suggestion markers i.e., some characteristics in each sentence, which, if they appear, then the sentence that is being studied probably contains a suggestion.

The basic idea that we applied is based, to a great extent, on Greek Grammar. We found that a suggestion can be "modelled" better than an argument. The basic concept is that in order to make a suggestion, a writer is "obliged" to follow some Grammar rules.

[9] `https://en.wikipedia.org/wiki/Logistic_regression`

[10] `https://en.wikipedia.org/wiki/Naive_Bayes_classifier`

[11] `https://en.wikipedia.org/wiki/Support_vector_machine`

[12] `https://en.wikipedia.org/wiki/Random_forest`

We know that according to Greek Grammar, in order to express incitement in a sentence, the writer should use mainly either the Imperative or the Subjunctive Grammatical mood. These two Grammatical moods are often linked to the meaning we want to detect in our texts.

Starting with the aforementioned concept, we noted that the use exclusively of Greek Grammar is not enough. In some way, we had to specify with more clarity what we were looking for in a text.

So, we extended the above concept. We combined Grammar rules with lemmas in order to achieve higher accuracy.

To make the above process clearer: Initially, using the "ILSP POS-Tagger" output, we noted the Grammar tags (**??**), as well as the lemmas of verbs (**??**) in order to create a list of verbs that are involved in cases where a sentence can be considered as a suggestion. We can say that we noted the verbs which define the meaning of a sentence, making it a suggestion.

An example can make the concept that was presented even clearer: Given the sentence *"a change in paragraph 5 of the law must be made.."* we noted that, in this example, "must be made" plays a determining role in order to put across the meaning of the sentence as a suggestion. This particular part of the sentence also has a particular Grammatical form.

```
<?xml version='1.0' encoding='UTF−8'?>
<cesDoc xmlns="http://www.xces.org/schema/2003" version="0.4">
  <text>
    <body>
      <p id="p1">
        <s id="s1" casing="lowercase">
          <t id="t1" word="..." tag="VbIsIdPr03SgXxIpAvXx" lemma="..."/>
          <t id="t2" word="..." tag="PtSj" lemma="..."/>
          <t id="t3" word="..." tag="VbMnIdXx03SgXxPePvXx" lemma="..."/>
          <t id="t4" word="..." tag="NoCmFeSgNm" lemma="..."/>
          <t id="t5" word="..." tag="AsPpPaFeSgAc" lemma="..."/>
          <t id="t6" word="..." tag="NoCmFeSgAc" lemma="..."/>
          <t id="t7" word="..." tag="DIG" lemma="..."/>
          <t id="t8" word="..." tag="AtDfMaSgGe" lemma="..."/>
          <t id="t9" word="..." tag="NoCmMaSgGe" lemma="..."/>
          <t id="t10" word="..." tag="PTERM_P" lemma="..."/>
        </s>
      </p>
    </body>
  </text>
</cesDoc>
```

More precisely, we noticed that some tags, such as "VbLsLdPr03SgXxLpAvXx" and "VbMnLdXx03SgXxPePvXx" appear in many cases. Finally, we created a list with many cases that we detected. Some cases appear below:

- We have to VbLsLdPr03SgXxpAvXx PtSj
- PtSj VbMnLdXxPepvXx must be studied
- VbMnLdXx03SgXxPePvXx PtSj must be applied
- Add PtSj VbMnLdPr03SXxLpPvxX
- ...

The above list is, in essence, the suggestion markers that we created. (The full list contains about eighty suggestion markers. The list is part of the submitted code.)

3.3.2 POS-Tagging and Lemmatization the set of Documents

The second stage for suggestion mining lies in the POS-Tagger performance. This process is the same as the one described in unit (**??**) and for the sake of brevity it will not be analysed again here.

3.3.3 Apply Information Retrieval Methods in order to find the Suggestions

Having created the suggestion markers, as we saw in unit (**??**), we have, at this point, to analyse the way with which we can find similarities between the sentences being analysed and the suggestion markers.

Initially, for every sentence in the process of analysis, the POS-Tagging & Lemmatization method (**??**) is applied. From the Output of this process we make good use of the information that is absolutely essential for this step. So, for every sentence we keep the following information:

- the verb tag.
- the verb lemma.
- the lemmas of words "to", "will".
- tags of the words "to", "will".
- "Qst" if the sentence is interrogative.

So, the sentence that we saw above, *"a change must be made"* in paragraph 5 of the law" becomes *"VbLsLdPr03SgXxLpAvXx should PtSj be VbMnLdXx03SgXxPePvXx made"*.

The next step is to create a similarity score between the sentence that came up and each one of the suggestion markers. The comparison was made using TF-IDF[13] and Jaccard Similarity[14] and as we will see in the following unit, the results were satisfying enough.

[13] `https://en.wikipedia.org/wiki/Tf%E2%80%93idf`

[14] `https://en.wikipedia.org/wiki/Jaccard_index`

One more point that we need to stress is that we divided the suggestion markers in 4 categories, depending on the Grammatical mood, “Imperative” or “Subjunctive” and depending on whether the verbs are in a Past Tense (should have). The categories, as well as all the suggestion markers can be found in the “suggestionQ file” through the online repository[15].

Finally, having made the above comparisons, only the highest score will be kept, as well as the category from which this score came up. We should also note that the sentences that contain “Qst” automatically get zero score because we wanted to exclude the interrogative sentences for the same reasons that were mentioned earlier.

3.3.4 Adding additional features for the optimization of Machine Learning Processs

In order to achieve better results in each sentence that resulted from the above process, we counted the number of words from which it is made up.

3.3.5 Apply Machine Learning

The last stage for the retrieval of suggestions is the application of a Machine Learning Process. To implement this process, a train set and a test set must be definitely determined.

3.3.5.1 Selecting Train and Test Set

In this case, too, the train set is stored in the Database. To be precise, the retrieval of the train set from the Database is done with the following query:

```
SELECT
    opngv_suggestion.weight,
    opngv_suggestion.category,
    opngv_suggestion.total_words,
    opngv_trainset.Suggestion
FROM
    opngv_sentence
    INNER JOIN opngv_suggestion
        ON opngv_sentence.comment_id = opngv_suggestion.comment_id
        AND opngv_sentence.sentence_id = opngv_suggestion.sentence_id
    INNER JOIN opngv_trainset
        ON opngv_sentence.comment_id = opngv_trainset.comment_id
        AND opngv_sentence.sentence_id = opngv_trainset.sentence_id
ORDER BY
    opngv_trainset.Suggestion DESC
LIMIT 366
```

[15] https://github.com/csdRepo/OpinionMining.git

In this case, too, the test set consists of the sentences of the text, more precisely, the sentences that came up from the process (??).

3.3.5.2 Machine Learning Process

As in the Argument Extraction (??), the Weka Framework was also used here. Again, for the sake of brevity, it will not be analysed here.

3.3.5.3 Machine Learning Algorithms

To verify the accuracy of the Train Set,we tested enough Algorithms:

- Native Bayes[16]
- Support Vector Machines[17]
- Random Forest[18]
- J48[19]

3.4 Overall Opinion Extraction

In this final chapter, the third leg of this Thesis, where the possibility of recognising the general degree of the writer's agreement in regard with the initial public consultation text of the Greek Government was studied, will be analysed. For this process, no kind of meta-data which would have supplied this information automatically (Agree/Disagree button, Like etc.), was used. The whole process has been completed by studying exclusively the users' texts.

For the effective mining of this kind of information from the texts, we used the "Sentiment Analysis" method, in an attempt to recognise the sentiments with which the users write their texts. Having analysed the sentiments, we noted that very often it seems that they match each writer's degree of agreement. In more detail, this process can be divided as follows:

- Translate Documents to English (??).
- Perform Sentiment Analysis (??).

3.4.1 Translate Documents to English

In order to be able to perform Sentiment Analysis, we had, in the first stage, to automatically translate the texts in the English language. The reason is that all the tools that are available do not support the Greek language. Also, studying and developing software that could perform Sentiment Analysis for Greek could constitute an independent task.

[16] https://en.wikipedia.org/wiki/Naive_Bayes_classifier
[17] https://en.wikipedia.org/wiki/Support_vector_machine
[18] https://en.wikipedia.org/wiki/Random_forest
[19] https://en.wikipedia.org/wiki/J48

It makes sense to think that with automatic translation, there are many times where the meaning of a sentence is altered and the translated text has many mistakes. This problem is even greater in Greek, because it has a complex and rich Grammar and subsequently, the texts in this case may be even more altered.

Nevertheless, the above alterations do not affect the application of Sentiment Analysis. This happens because during Sentiment Analysis implementation, we are more interested in the connotation of the words, i.e. how much of a negative or positive meaning a word gives to a text.

For the automatic translation of texts from Greek to English, "Google Translate" has been used, a service provided for free through the appropriate API[20].

3.4.2 Perform Sentiment Analysis

In the demo that accompanies this Thesis, there is the possibility to use two external tools that provide this kind of option.

- **SentiStrength[21]:** *"SentiStrength estimates the strength of positive and negative sentiment in short texts, even for informal language. It has human-level accuracy for short social web texts in English, except political texts. SentiStrength reports two sentiment strengths:*

  - *-1 (not negative) to -5 (extremely negative)*
  - *1 (not positive) to 5 (extremely positive)*

  Why does it use two scores? Because research from psychology has revealed that we process positive and negative sentiment in parallel - hence mixed emotions. SentiStrength can also report binary (positive/negative), trinary (positive/negative/neutral) and single scale (-4 to +4) results."

- **Sentiment Analysis with Python NLTK Text Classification[22]:** *"Sentiment analysis using a NLTK 2.0.4 powered text classification process. It can tell you whether it thinks the text you enter below expresses positive sentiment, negative sentiment, or if it's neutral. Using hierarchical classification, neutrality is determined first, and sentiment polarity is determined second, but only if the text is not neutral."*

[20] https://cloud.google.com/translate/

[21] http://sentistrength.wlv.ac.uk

[22] http://text-processing.com/docs/sentiment.html

4

Evaluation and Results

In this chapter, the efficiency of the mechanisms that were studied above will be analysed, on whether they can achieve targeted data extraction. The experiments that were carried out, as well as the results that will be presented below are divided in three units, depending on the kind of data we want to extract:

- Argument Extraction (**??**)
- Suggestion Extraction (**??**)
- Overall Opinion Extraction (**??**)

4.1 Argument Extraction

In this subsection the results of the methods that were introduced in the previous chapters on the writer's argument extraction will be presented. In this subsection especially, we are going to try to present some indicators for the Argument Markers that we set. These indicators show the "ability" of each variable to "find" an Argument (**??**). Also, we are going to see the efficiency of each Algorithm that we tried in this procedure (**??**). Finally, we are going to present some diagrams that show the efficiency of the Train Set that we have created on the writer's Argument Extraction from the texts (**??**).

4.1.1 Argument Markers

As we saw above, the corpus that was studied had a total of 1015 sentences, which make up approximately two hundred users' texts (user commentaries). Initially, we need to stress that in the corpus, the number of sentences that were characterised as Argumentative is relatively bigger that the number of sentences that were not characterised as Argumentative. This property is shown in the following diagram.

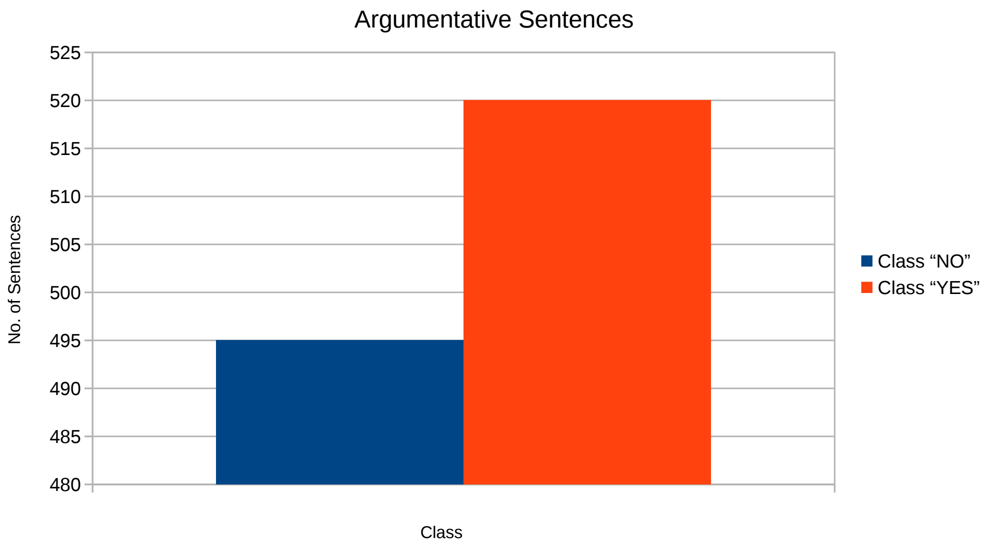


Figure 4.1: Argumentative Sentences in Train Set.

In the rest of this unit (**??**), class "yes", i.e. the sentences that are "Argumentative" will be symbolised with red and respectively, the sentences belonging to class "no" will be symbolised with blue.

In the next diagrams we can see for each one of the Argument Markers that we have set, the ability of each variable to define an Argument. The axis X shows the frequency of each variable in each sentence. What we note in most diagrams, is that as frequency increases, class "yes" surpasses class "no"

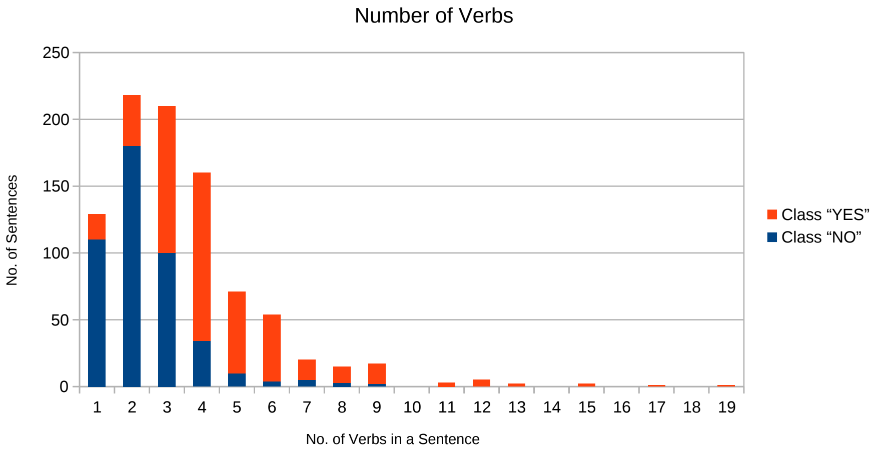


Figure 4.2: Argument Marker - Number of Verbs in a Sentence.

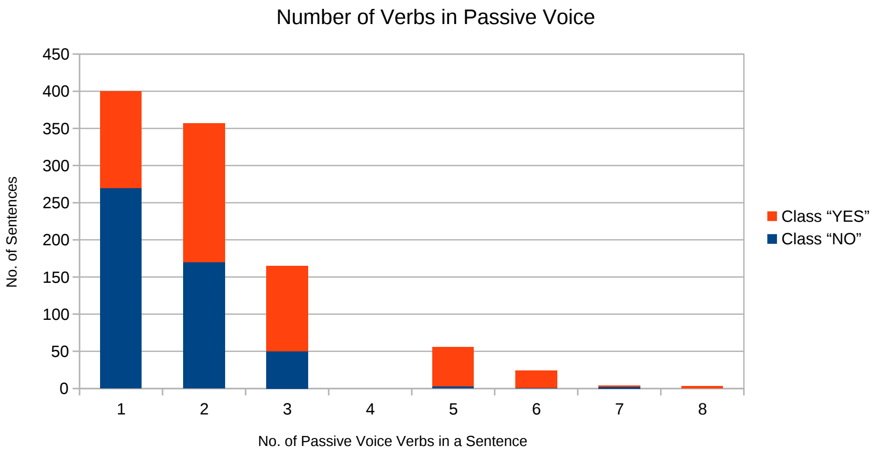


Figure 4.3: Argument Marker - Number of Verbs in Passive Voice in a Sentence.

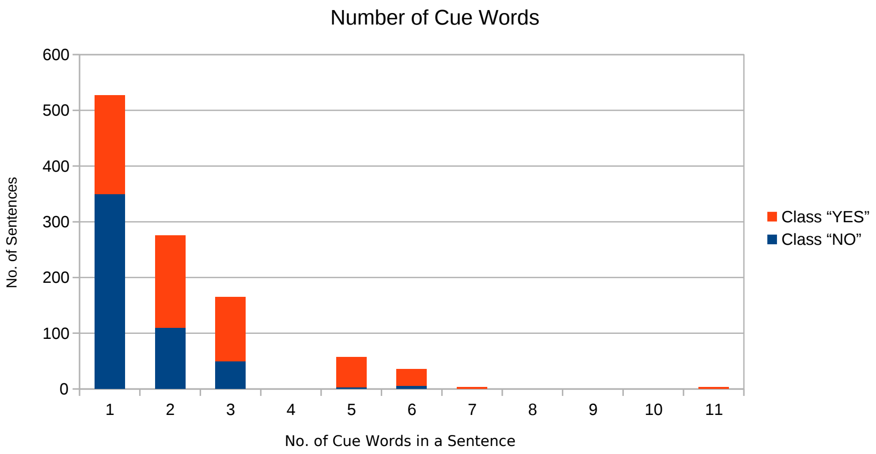


Figure 4.4: Argument Marker - Number of Cue Words in a Sentence.

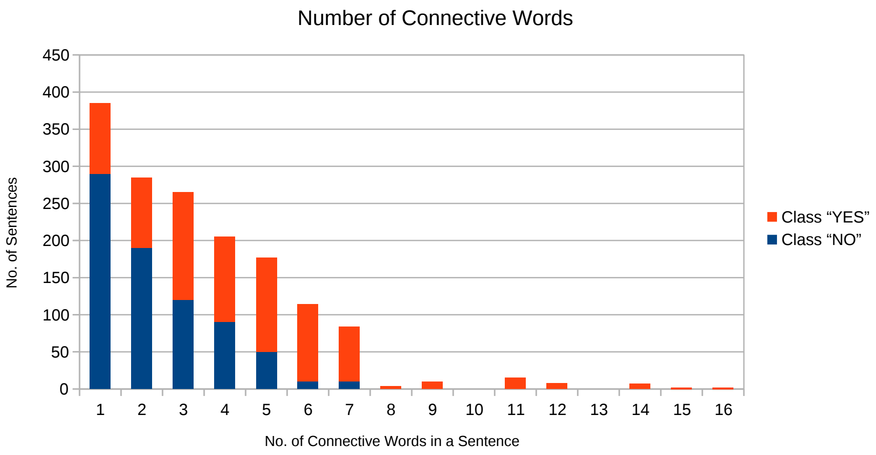


Figure 4.5: Argument Marker - Number of Connective Words in a Sentence.

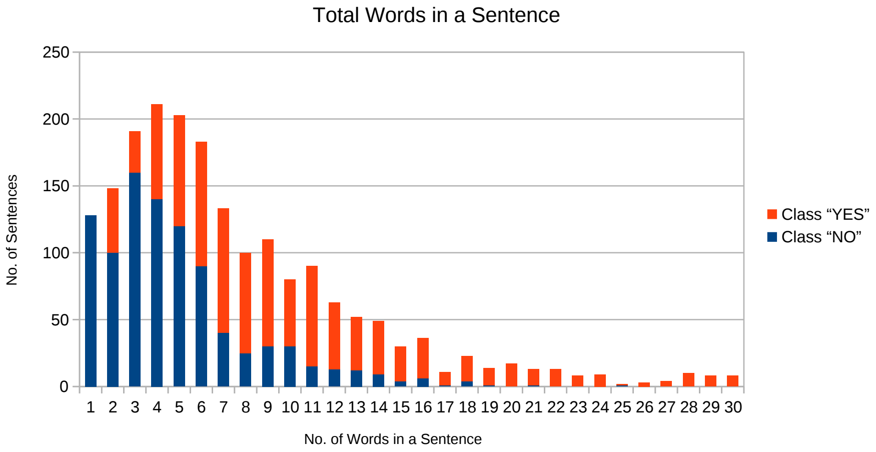


Figure 4.6: Argument Marker - Total words in a Sentence.

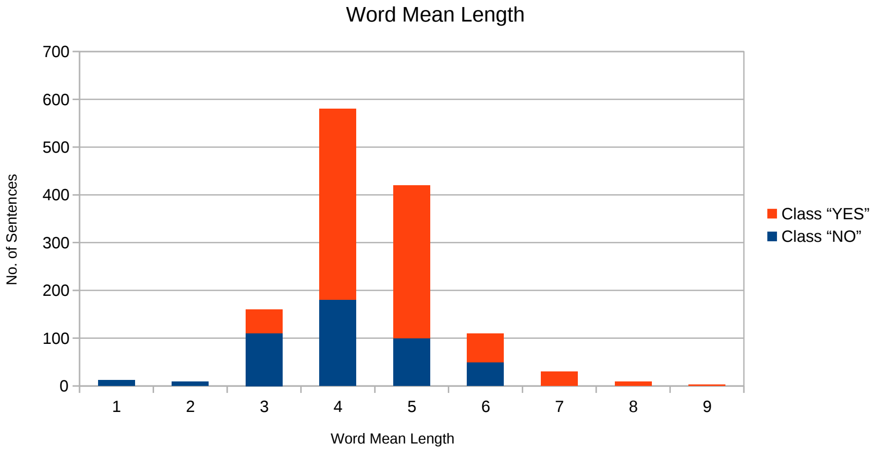


Figure 4.7: Argument Marker - Word Mean Length.

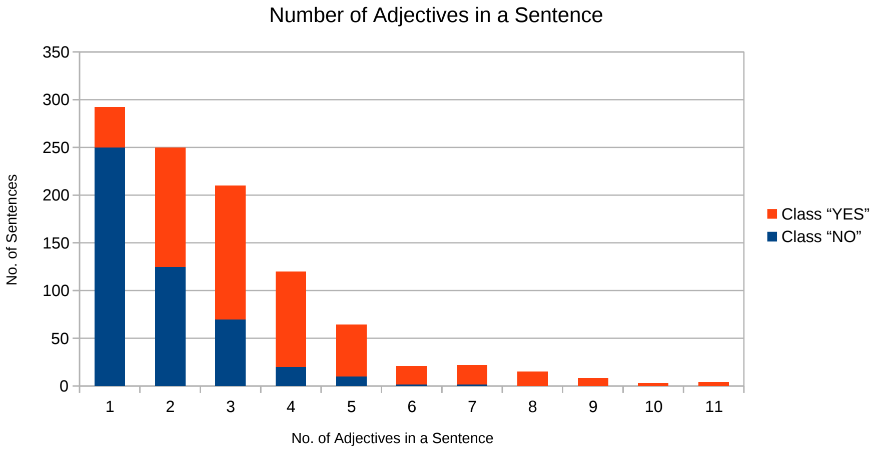


Figure 4.8: Argument Marker - Number of Adjectives in a Sentence.

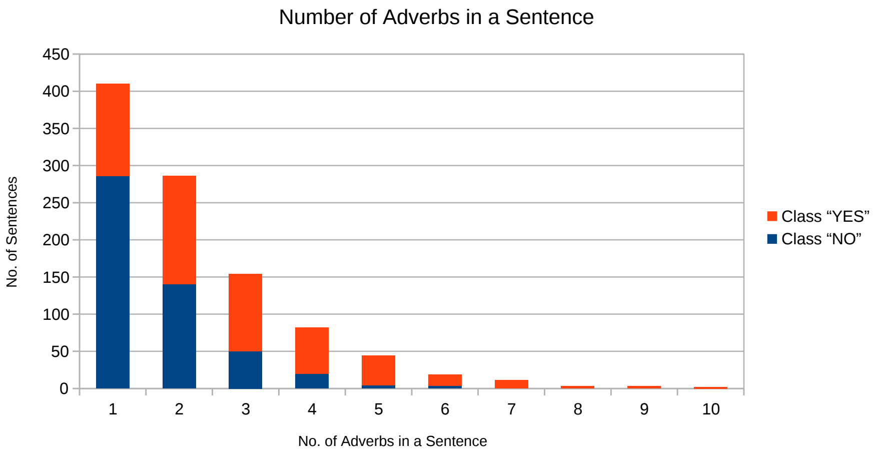


Figure 4.9: Argument Marker - Number of Adverbs in a Sentence.

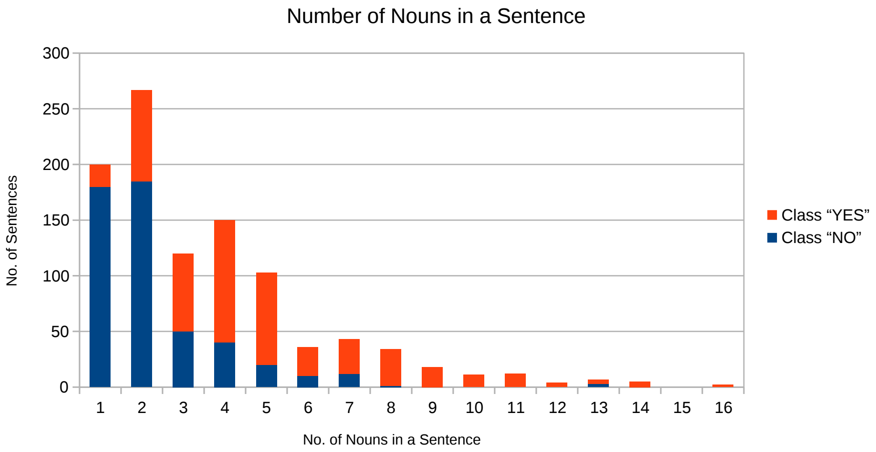


Figure 4.10: Argument Marker - Number of Nouns in a Sentence.

4.1.2 Algorithms used in Machine Learning Procedure

In this subsection, we will see the results that the Algorithms which were used to check the accuracy of Machine Learning Process, produced. The results have been produced by carrying out classification on the Train Set and using “10 Fold Cross Validation”[1].

Table 4.1: Detailed Accuracy for Class “No” (Argument Extraction).

Algorithm	Precision	Recall	F-Measure
SVM	*0.815*	*0.830*	**0.823**
Random Forest	**0.818**	*0.818*	*0.818*
Native Bayes	*0.718*	**0.899**	*0.798*
Logistic Regression	*0.801*	*0.819*	*0.819*

[1] `http://www.opengov.gr`

Table 4.2: Detailed Accuracy for Class "Yes" (Argument Extraction).

Algorithm	Precision	Recall	F-Measure
SVM	*0.836*	*0.821*	**0.828**
Random Forest	*0.827*	**0.827**	*0.827*
Native Bayes	**0.873**	*0.663*	*0.754*
Logistic Regression	*0.837*	*0.802*	*0.819*

Table 4.3: Weighted Average on both Classes (Argument Extraction).

Algorithm	Precision	Recall	F-Measure
SVM	**0.826**	**0.826**	**0.826**
Random Forest	*0.823*	*0.823*	*0.823*
Native Bayes	*0.797*	*0.778*	*0.776*
Logistic Regression	*0.820*	*0.819*	*0.819*

From the above tables, it is obvious that the best results were achieved by using the "Support Vector Machines" Algorithm. For this reason, we will look at some useful information about this result straight away.

Table 4.4: Additional Statistical Information (Argument Extraction).

	Frequenncy	Percentage
Correctly Classified Instances	*838*	*82.56%*
Incorrectly Classified Instances	*177*	*17.4383*
Kappa statistic	*0.6512*	-
Mean absolute error	*0.1744*	-
Root mean squared error	*0.4176*	-
Relative absolute error	-	*34.90%*
Root relative squared error	-	*83.54%*
Coverage of cases (0.95 level)	-	*82.56%*
Mean rel. region size (0.95 level)	-	*50%*
Total Number of Instances	*1015*	-

4.1.3 Information about the Train Set

One more measurement that was performed concerns the Train Set. More precisely, in the following diagram, the "Error Rate" in the ability to correctly characterize the sentences that are being analysed, appears. We can note that by using from 30% of the Train Set (approximately 300 sentences) up to 100%, the "Error Rate" is limited between 17.5% - 18.3%. For the next diagram, "Support Vector Machines" Algorithm was used.

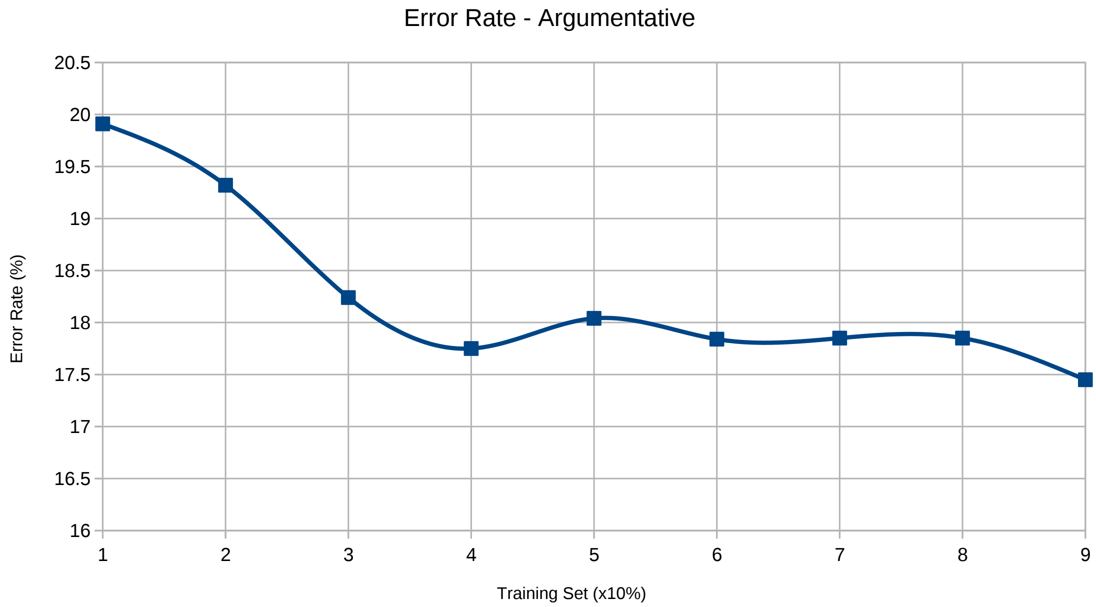


Figure 4.11: Error Rate of Argumentative Sentence Classification.

4.2 Suggestion Extraction

For the confirmation of the results for extraction of the suggestions that the writers of the texts made, we needed to create two scenarios. Initially, we carried out "10 Fold Cross Validation" (**??**) on the Train Set. Next, we created a second scenario, according to which we divided the Train Set so that it contains an equal number of sentences that constitute suggestions and of sentences that do not. Next, we used this set as a Train Set and performed again the Machine Learning Process (**??**).

4.2.1 "10 Fold Cross Validation" on Train Set

As described above, initially, we tried to apply "10 Fold cross validation" on the Train Set. The results are shown in the following matrix:

Table 4.5: Detailed Accuracy for Class "No" (Suggestion Extraction).

Algorithm	**Precision**	**Recall**	**F-Measure**
J48	*0.881*	*0.923*	*0.901*
Random Forest	*0.890*	*0.915*	*0.902*
Native Bayes	**0.912**	*0.915*	*0.604*
SVM	*0.839*	**0.989**	**0.908**

Table 4.6: Detailed Accuracy for Class “Yes” (Suggestion Extraction).

Algorithm	Precision	Recall	F-Measure
J48	*0.552*	*0.432*	*0.485*
Random Forest	*0.556*	*0.489*	*0.519*
Native Bayes	*0.608*	**0.601**	**0.604**
SVM	**0.735**	*0.137*	*0.230*

Table 4.7: Weighted Average on both Classes (Suggestion Extraction).

Algorithm	Precision	Recall	F-Measure
J48	*0.822*	*0.834*	*0.826*
Random Forest	*0.830*	*0.837*	*0.833*
Native Bayes	**0.858**	**0.858**	**0.858**
SVM	*0.820*	*0.835*	*0.786*

From the above tables, we can see that the best results were achieved by using the “Native Bayes” Algorithm. For this reason, we will now see some useful information about this result.

Table 4.8: Additional Statistical Information (Suggestion Extraction).

	Frequenncy	Percentage
Correctly Classified Instances	*871*	*85.81%*
Incorrectly Classified Instances	*144*	*14.19%*
Kappa statistic	*0.518*	-
Mean absolute error	*0.1901*	-
Root mean squared error	*0.3382*	-
Relative absolute error	-	*64.21%*
Root relative squared error	-	*87.98%*
Coverage of cases (0.95 level)	-	*95.67%*
Mean rel. region size (0.95 level)	-	*70.64%*
Total Number of Instances	*1015*	-

4.2.2 Equivalent Train Set

In the second scenario, a different Train Set was used. We divided the Train Set in a way that it contains equal amounts of sentences that can be characterised as suggestions, compared to those that cannot. The reason is that the Train Set contains relatively less entries that refer to suggestions compare to the sentences that do not constitute suggestions. So, we used a corpus that contained two hundred sentences that are characterised as suggestions and two hundred more that are not. Next, we used this set as a Train Set and as a Test Set we used the whole corpus. The results are shown in the following tables:

Table 4.9: Detailed Accuracy for Class “No” (Suggestion Extraction, using Equivalent Train Set).

Algorithm	Precision	Recall	F-Measure
J48	*0.940*	*0.810*	*0.870*
Random Forest	**0.996**	*0.810*	**0.893**
Native Bayes	*0.941*	**0.828**	*0.881*
SVM	*0.939*	*0.819*	*0.875*

Table 4.10: Detailed Accuracy for Class “Yes” (Suggestion Extraction, using Equivalent Train Set).

Algorithm	Precision	Recall	F-Measure
J48	*0.470*	*0.765*	*0.582*
Random Forest	**0.533**	**0.984**	**0.641**
Native Bayes	*0.495*	*0.765*	*0.601*
SVM	*0.479*	*0.760*	*0.588*

Table 4.11: Weighted Average on both Classes (Suggestion Extraction, using Equivalent Train Set).

Algorithm	Precision	Recall	F-Measure
J48	*0.855*	*0.802*	*0.818*
Random Forest	**0.912**	**0.841**	**0.857**
Native Bayes	*0.861*	*0.817*	*0.831*
SVM	*0.856*	*0.808*	*0.823*

From the above tables we can see that the best results were achieved by using the “Random Forest” Algorithm.

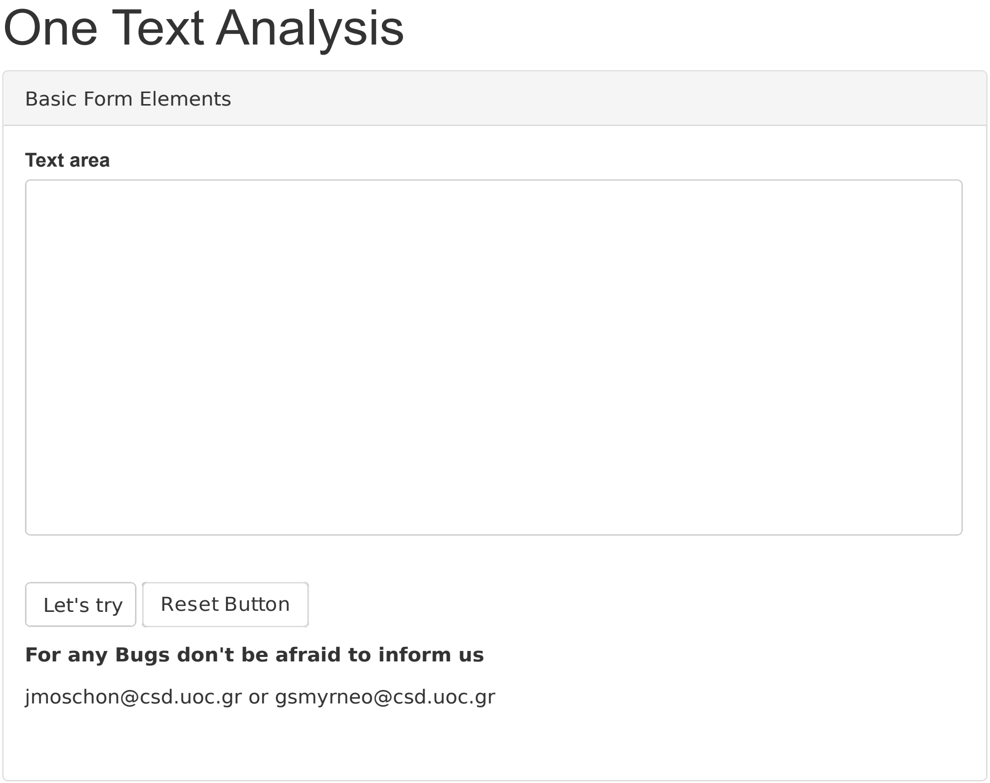
Table 4.12: Additional Statistical Information (Suggestion Extraction, using Equivalent Train Set).

	Frequenncy	Percentage
Correctly Classified Instances	*854*	*84.14%*
Incorrectly Classified Instances	*161*	*15.86%*
Kappa statistic	*0.5966*	-
Mean absolute error	*0.2273*	-
Root mean squared error	*0.3467*	-
Relative absolute error	-	*47.98%*
Root relative squared error	-	*73.02%*
Coverage of cases (0.95 level)	-	*97.14%*
Mean rel. region size (0.95 level)	-	*83.00%*
Total Number of Instances	*1015*	-

5
Demo Application

Within the frame of this Thesis, a “Demo Application” which is divided in two distinct and independent parts, was created. The system can be tested by following the next link: `http://snf-552985.vm.okeanos.grnet.gr/opinion/`.

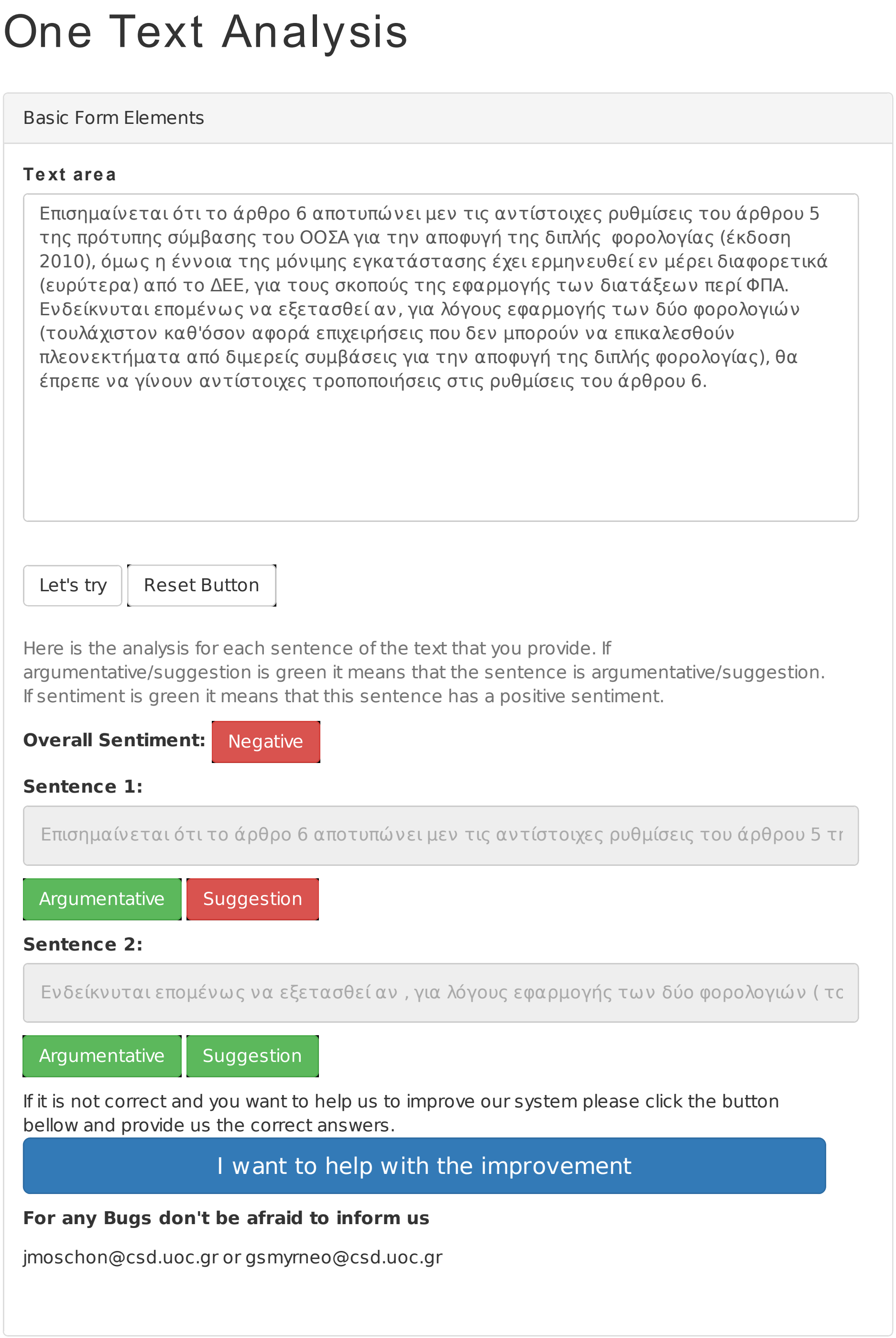
The first part is the “Back-End” which is implemented in Java SE8 following the methodology we analysed in Chapter 3, creating in that way, an HTTP server aiming at the communication of these two parts. The second part is the “Front-End”, which also constitutes the part with which the user communicates with our system, is created with the use of HTML5 and Javascript. These two parts communicate through message exchange in “JSON” form. In this demo, the user is given the opportunity to give as “entry” a text from the OpenGov Website.



For any Bugs don't be afraid to inform us

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

In the above picture we can see the initial page in which the user gives the texts that will be analysed.



One Text Analysis

Basic Form Elements

Text area

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 της πρότυπης σύμβασης του ΟΟΣΑ για την αποφυγή της διπλής φορολογίας (έκδοση 2010), όμως η έννοια της μόνιμης εγκατάστασης έχει ερμηνευθεί εν μέρει διαφορετικά (ευρύτερα) από το ΔΕΕ, για τους σκοπούς της εφαρμογής των διατάξεων περί ΦΠΑ. Ενδείκνυται επομένως να εξετασθεί αν, για λόγους εφαρμογής των δύο φορολογιών (τουλάχιστον καθ'όσον αφορά επιχειρήσεις που δεν μπορούν να επικαλεσθούν πλεονεκτήματα από διμερείς συμβάσεις για την αποφυγή της διπλής φορολογίας), θα έπρεπε να γίνουν αντίστοιχες τροποποιήσεις στις ρυθμίσεις του άρθρου 6.

Here is the analysis for each sentence of the text that you provide. If argumentative/suggestion is green it means that the sentence is argumentative/suggestion. If sentiment is green it means that this sentence has a positive sentiment.

Overall Sentiment: 🟥 Negative

Sentence 1:

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 τ

🟩 Argumentative 🟥 Suggestion

Sentence 2:

Ενδείκνυται επομένως να εξετασθεί αν , για λόγους εφαρμογής των δύο φορολογιών (τ

🟩 Argumentative 🟩 Suggestion

If it is not correct and you want to help us to improve our system please click the button bellow and provide us the correct answers.

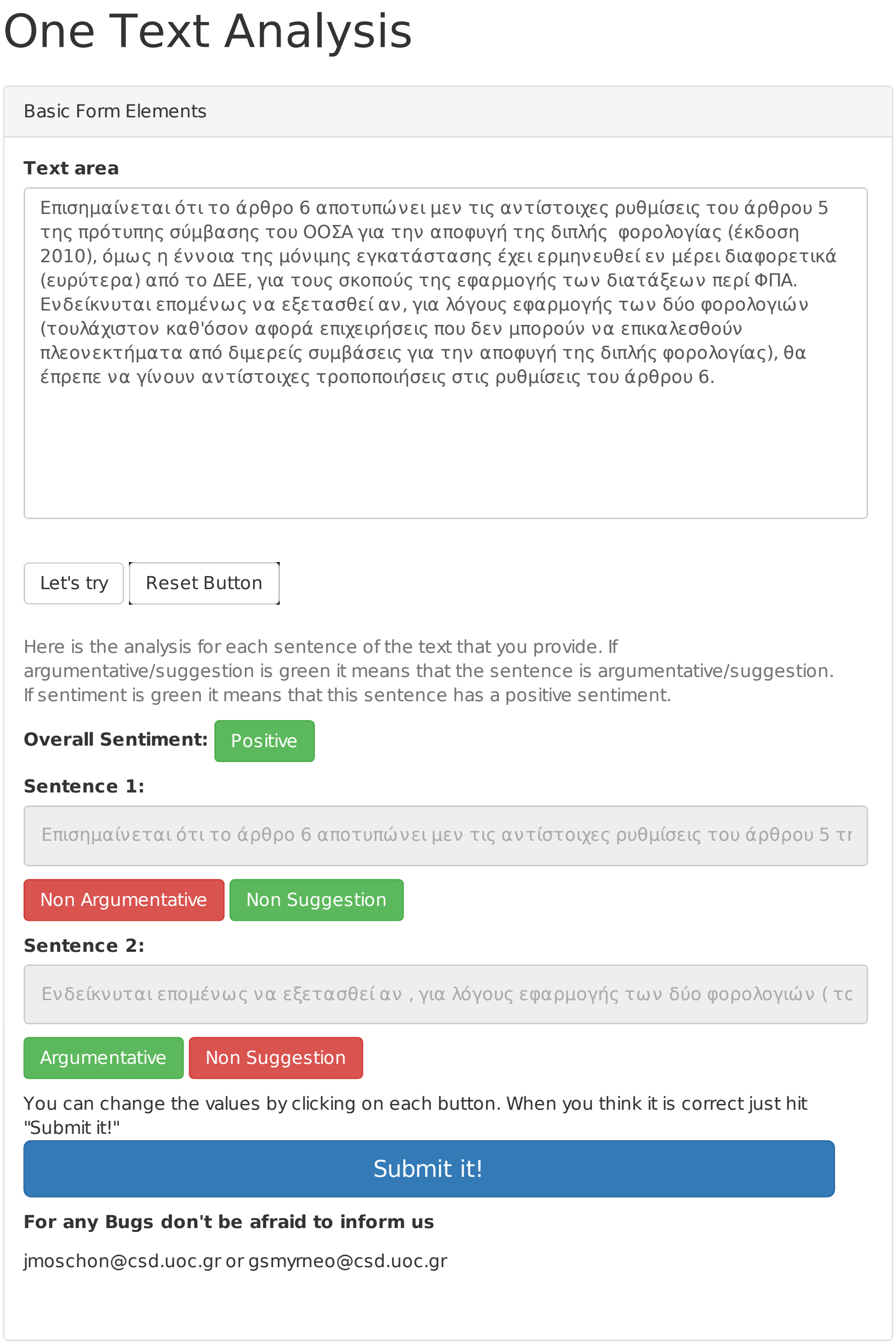
For any Bugs don't be afraid to inform us

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

By pressing the "Let's try" button, the results of the analysis of that specific text appear. The "Overall Sentiment" field, is red if the user is negative towards the text and green if he is positive. Then, one by one the sentences of the text appear and below each sentence an indication whether the sentence is an Argument or a Suggestion and red colour indicates that it is a non-Argument or non-Suggestion.

Another useful function that this demo application offers is that the user is being given the chance to "correct" that same System in case the results are not correct.

This is done by pressing the blue button “I want to help with the improvement”. We should note that it is an important enough function, because in its total the System is based on Machine Learning.



One Text Analysis

Basic Form Elements

Text area

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 της πρότυπης σύμβασης του ΟΟΣΑ για την αποφυγή της διπλής φορολογίας (έκδοση 2010), όμως η έννοια της μόνιμης εγκατάστασης έχει ερμηνευθεί εν μέρει διαφορετικά (ευρύτερα) από το ΔΕΕ, για τους σκοπούς της εφαρμογής των διατάξεων περί ΦΠΑ. Ενδείκνυται επομένως να εξετασθεί αν, για λόγους εφαρμογής των δύο φορολογιών (τουλάχιστον καθ'όσον αφορά επιχειρήσεις που δεν μπορούν να επικαλεσθούν πλεονεκτήματα από διμερείς συμβάσεις για την αποφυγή της διπλής φορολογίας), θα έπρεπε να γίνουν αντίστοιχες τροποποιήσεις στις ρυθμίσεις του άρθρου 6.

Here is the analysis for each sentence of the text that you provide. If argumentative/suggestion is green it means that the sentence is argumentative/suggestion. If sentiment is green it means that this sentence has a positive sentiment.

Overall Sentiment: Positive

Sentence 1:

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 τη

Sentence 2:

Ενδείκνυται επομένως να εξετασθεί αν , για λόγους εφαρμογής των δύο φορολογιών (τc

You can change the values by clicking on each button. When you think it is correct just hit "Submit it!"

For any Bugs don't be afraid to inform us

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

After pressing the “I want to help with the Improvement” button, as we can see in the above picture the “characterising” buttons are now in a deeper colour and the user, by pressing on them can change their colour. When he changes them and considers them to be correct, the next step is to press the “Submit it” button in order to send the corrected results back to the System.

6
Conclusion

In this Thesis, three self-contained mechanisms for data mining from users' texts that were published in the Greek Government's online service for Public consultation were presented. The information that is extracted by the three Mechanisms constitutes information of semantic nature. The three mechanisms for argument extraction, suggestions and overall opinion are to a great extent adapted for this particular corpus that we studied, at the same time especially for the mechanism for argument extraction, a relatively high accuracy is achieved in comparison to other studies that were taken into consideration for the completion of this Thesis. At the same, the process by which we attempted to extract suggestions to a great enough extent constitutes our own idea, given that in the past, few Systems that create that particular type of analysis had been implemented. Subsequently, up to this point, the results we received from the evaluation are very satisfying.

For the completion of this Thesis, many arguments were dealt with, which were mainly algorithmic difficulties. A lot of studying was needed to be done in the available Related Work so that we can move on to the implementation. One more big difficulty was the incomplete knowledge we had about Machine Learning. All the difficulties were dealt with by studying the available bibliography.

The first thought about Future work based on this Thesis is that further separation of Arguments could be done, as it was analysed in chapter 2. That means seeking to separate the part of the sentence in which the writer of the text claims something from the part that constitutes evaluation-validation. In the Future, we could also attempt Optimization of the suggestion extraction process, by trying other techniques. Finally, when it comes to the mechanism that extracts a writer's general knowledge on a matter, it would be important enough to test its efficiency on a better dataset.

Bibliography

[1] Wandhofer, T., Van Eeckhaute, C., Taylor, S., and Fernandez, M. (2012). Wegov analysis tools to connect policy makers with citizens online. In Proceedings of the tGov Conference May 2012, Brunel University.

[2] Diakopoulos, N. A. and Shamma, D. A. (2010). Characterizing debate performance via aggregated twitter sentiment. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.

[3] Tumasjan, A., Sprenger, T. O., Sandner, P. G., and Welpe, I. M. (2010). Predicting elections with twitter: What 140 characters reveal about political sentiment.

[4] Argument Extraction from News, Blogs, and Social Media Theodosis Goudas Department of Digital Systems, University of Piraeus, Christos Louizos, Department of Informatics & Telecommunications University of Athens, Georgios Petasis and Vangelis Karkaletsis Software and Knowledge Engineering Laboratory.

[5] Simone Teufel. 1999. Argumentative Zoning: Information Extraction from Scientific Text. Ph.D. thesis, University of Edinburgh.

[6] Alan Sergeant. 2013. Automatic argumentation extraction. In Proceedings of the 10th European Semantic Web Conference, ESWC '13, pages 656–660,Montpellier, France.

[7] Annotating Argument Components and Relations in Persuasive Essays C Stab,I Gurevych COLING 2014.