Extracting Multi-document Summaries with a Double Clustering Approach

Sara Botelho Silveira and António Branco

University of Lisbon, Portugal
Edifício C6, Departamento de Informática
Faculdade de Ciências, Universidade de Lisboa
Campo Grande, 1749-016 Lisboa, Portugal
{sara.silveira,antonio.branco}@di.fc.ul.pt

Abstract. This paper presents a method for extractive multi-document summarization that explores a two-phase clustering approach. First, sentences are clustered by similarity, and one sentence per cluster is selected, to reduce redundancy. Then, in order to group them according to topics, those sentences are clustered considering the collection of keywords. Additionally, the summarization process further includes a sentence simplification step, which aims not only to create simpler and more incisive sentences, but also to make room for the inclusion of relevant content in the summary as much as possible.

Keywords: Multi-document summarization, sentence clustering, sentence simplification.

1 Introduction

Automatic text summarization is the process of creating a summary from one or more input text(s) through a computer program. It seeks to combine several goals: (1) the preservation of the idea of the input texts; (2) the selection of the most relevant content of the texts; (3) the reduction of eventual redundancy; and (4) the organization of the final summary. While meeting these demands, it must be ensured that the final summary complies with the desired compression rate. This is, thus, a complex task to be accomplished by a human let alone a computer.

This paper presents a multi-document summarization system, SIMBA, that receives a collection of texts and retrieves an extract summary, that is, a text composed by sentences obtained from the source texts. The main goals of multi-document summarization are tackled through a double clustering approach, which includes a similarity clustering phase and a keyword clustering phase. Redundancy is addressed by clustering all the sentences based on a measure of similarity. Afterwards, the sentences are assembled by topics, using the keywords retrieved from the collection of texts. This approach impacts on the content of the final summary. On the one hand, the similarity clustering ensures that this content is not repetitive. On the other hand, the keyword clustering assures the

G. Bouma et al. (Eds.): NLDB 2012, LNCS 7337, pp. 70–81, 2012.

selection of the most relevant content, the preservation of the idea of the input texts, and the organization of the final summary. Furthermore, this system includes a step of sentence simplification, at the end of its processing pipeline, which aims at producing simpler and more incisive sentences, thus allowing that more relevant content enters the summary. Finally, an automatic evaluation of SIMBA is presented.

From now on, this paper is organized as follows: Section 2 provides an overview of automatic summarization systems that inspired our work; Section 3 describes our multi-document summarization system; Section 4 reports on the system evaluation; and, finally, the conclusions in Section 5.

2 Related Work

The summarization process comprises typically three stages: analysis, transformation and synthesis [11].

The **analysis** phase processes the input text(s) and builds an internal representation of those text(s). Typical steps include tokenization, part-of-speech annotation, lemmatization, or stop-words removal. Also, in this phase, the information units (sentences) to be handled in the next phases are identified.

The **transformation** phase aims to handle the sentences and to identify the ones that are the most relevant within the input text(s). Each sentence is assigned a score. In many summarization systems, measures of significance of the sentence are calculated over the text(s) to determine each sentence score.

These measures of significance can be in the form of reward metrics or penalty metrics, that take into account features related to the sentence or the words composing it. The reward ones assign positive scores to the sentences, while penalty metrics remove values from the sentence scores.

The keyword method was proposed in [9], and assumes that more frequent words in a document indicate the topic discussed, so that more frequent words are assigned with reward values. [5] experimented the cue method (sentences with cue words are relevant); the location method (sentences in specific positions are important); finally, the title method (words in the title are keywords of the text).

In fact, the most used score measure has been `tf-idf` (Term Frequency $\times$ Inverse Document Frequency). Sentences containing frequent or highly infrequent terms are more likely to be significant in the global context of the collection of texts.

Vocabulary overlap measures are also used to define sentence scores. The Dice coefficient [4], the Jaccard index [6], and the Cosine similarity coefficient compute a similarity metric between pairs of sentences, determining a relation between them. For instance, to, along with other features, define the sentence score, [13] computes the overlap between each sentence and the first sentence of the text, which, in news articles, is considered one of the most important sentences. Yet, [15] computes the overlap between each sentence and the document title, in order to reward the sentences that have a high degree of similarity with the title.

The reward metrics are both suitable for single- or multi-document summarization. However, multi-document summarization represents new challenges. Beyond selecting the most salient information which represents the collection of documents, removing the redundancy present in the texts is another challenging issue to overcome.

Many works addressed these challenges by trying new metrics of word or sentence significance. The Maximal Marginal Relevance (MMR) [3] is a linear combination metric that relates query-relevance with information-novelty. It strives to reduce redundancy while considers query relevance to select the appropriate passages to be part of the summary.

After determining the sentence scores, **penalty metrics** can be computed to refine each score. The most common penalty metric is sentence length. Sentences with less than ten words are considered too small and conveying limited information, so that a predefined score value is removed from the sentence score. [8] used another metric, the stigma words penalty. Sentences starting with conjunctions, question marks, pronouns such as "he"/"she"/"they", and the verb "say" and its derivatives are penalized. Finally, the text publication date can either be a reward or a penalty metric, whether the text is a recent one or not.

Once all the sentences in the text(s) have been scored, the **generation** phase aims to order those sentences and to select the ones that will compose the final summary, seeking to fulfill the compression rate. Typically, the sentences are ordered considering their score. The ones with the highest score are selected until the compression rate is attained.

3 The SIMBA System

SIMBA is an extractive multi-document summarizer for the Portuguese language. It receives a collection of Portuguese texts, from any domain, and produces informative summaries, for a generic audience. The length of the summaries is determined by a compression rate value that is submitted by the user. It performs summarization by using a shallow yet efficient approach that relies on statistical features computed over the text elements.

Summarization is performed by executing five main stages: identification, matching, filtering, reduction, and presentation. These stages are described in the following sections. These five stages can be enclosed in the three generic phases of summarization mentioned above, as identification is a step of the analysis phase; matching and filtering are stages of the transformation phase; and, finally, reduction and presentation are included in the synthesis phase.

3.1 Identification

The identification stage is executed in two phases. The first phase handles the documents submitted by the user, converts them into the same format and removes the existing noise.

Once the documents are in the same format, texts are accessible to be processed automatically. Afterwards, a set of shallow processing tools for Portuguese, LX-Suite [2], is used to annotate the texts. Sentence and paragraph boundaries are identified and words are tagged, with its corresponding POS and lemmata. Also, for each sentence, a parse tree, representing the sentence syntactic structure, is built by LX-Parser [14].

Henceforth, the collection of texts is handled as a set of sentences.

3.2 Matching

The matching stage aims to identify relevant information in the collection of texts. First, sentence scores are computed. Then, sentences are clustered by similarity to remove redundancy within the collection. Finally, sentences are clustered by keywords to identify the ones that have the most relevant information.

It is important to note here that in our summarization process we consider three types of scores: the main score, the extra score and the complete score. The main score reflects the sentence relevance in the overall collection of sentences. The extra score is used in the summarization process to reward or penalize the sentences, by adding or removing predefined score values.[1] The complete score is the sum of these two scores.

Computing Sentence Main Score. Once the sentences and the words have been identified, `tf-idf` score is computed for each word, by considering its lemma. The sentence main score is then the sum of the `tf-idf` score of each word, smoothed by the total number of words in the sentence.

Clustering Sentences by Similarity. In order to identify redundant sentences, conveying the same information, the next step aims to cluster sentences, considering their degree of similarity.

The similarity between two sentences (Equation 3) comprises two dimensions, computed considering the word lemmas: the sentences subsequences (Equation 1) and the word overlap (Equation 2).

The subsequences value is inspired in ROUGE-L and consists in the sum of the number of words in all the subsequences common to each sentence, smoothed by the total number of words of each sentence being considered, and divided by the total number of subsequences found between the two sentences. The overlap value is computed using the Jaccard index [6].

The similarity value is the average of both these values: the overlap and the subsequences value. It is then confronted with a predefined threshold – similarity threshold[2] –, initially set to 0.75, determining that sentences must have at least 75% of common words or subsequences to be considered as conveying the same information.

[1] The predefined value is set to 0.1, both for the reward and the penalty values. This value has been determined empirically, through a set of experiments.

[2] This threshold was determined empirically, using a set of experiments, since there is no reference for such a value for the Portuguese language.

$$subsequences(s_1, s_2) = \frac{\sum_i \left(\frac{\text{subsequence}_i}{\text{totalWords}_{s_1}} + \frac{\text{subsequence}_i}{\text{totalWords}_{s_2}} \right)}{totalSubsequences} \quad (1)$$

$$overlap(s_1, s_2) = \frac{\sum \text{commonWords}(s_1, s_2)}{\text{totalWords}_{s_1} + \text{totalWords}_{s_2} - \sum \text{commonWords}(s_1, s_2)} \quad (2)$$

$$similarity(s_1, s_2) = \frac{\text{subsequences}(s_1, s_2) + \text{overlap}(s_1, s_2)}{2} \quad (3)$$

$\text{overlap}(s_1, s_2)$ – number of overlapping words between the two sentences.
$\text{subsequence}(s_1, s_2)$ – number of overlapping words in the subsequences between the two sentences.
$\text{commonWords}(s_1, s_2)$ – common words between the two sentences.
totalWords_{s_i} – total words in the sentence i.
subsequence_i – number of words of the subsequence i.
totalSubsequences – number of subsequences between the two sentences.

Two examples are discussed below. Taking into account this threshold, the sentences in the following example are considered to be similar.

Sentence#1:
A casa que os Maias vieram habitar em Lisboa, no outono de 1875, era conhecida pela casa do Ramalhete.

The house in Lisbon to which the Maias moved in the autumn of 1875, was known as the Casa do Ramalhete.

Sentence#2:
A casa que os Maias vieram habitar, no outono de 1875, era conhecida pela casa do Ramalhete.

The house to which the Maias moved in the autumn of 1875, was known as the Casa do Ramalhete.

Overlap	Subsequences	Similarity Value
0.89	0.95	0.92

These sentences share most of the words, but there is a leap (*"em Lisboa"*) between Sentence#1 and Sentence#2. Both the overlap and the subsequences values are high, because these two sentences share most of their words. So, as the similarity value is high (0.92), the sentences are considered similar.

The two sentences in the following example are not similar though having many words in common.

Sentence#1:
A casa que os Maias vieram habitar em Lisboa, no outono de 1875, era conhecida pela casa do Ramalhete.

The house in Lisbon to which the Maias moved in the autumn of 1875, was known as the Casa do Ramalhete.

Sentence#2:
A casa que os Maias vieram habitar em Lisboa, no outono de 1875, era conhecida na vizinhança da Rua de S. Francisco de Paula, pela casa do Ramalhete ou simplesmente o Ramalhete.

The house in Lisbon to which the Maias moved in the autumn of 1875, was known in Rua S. Francisco de Paula, as the Casa do Ramalhete or, more simply, as Ramalhete.

Overlap	Subsequences	Similarity Value
0.59	0.79	0.69

Despite Sentence#1 is contained in Sentence#2, both sentences are considered not to be similar, since their similarity value is below the threshold.

Afterwards, the sentences are actually clustered considering their similarity value. A cluster is composed by a collection of sentences, a similarity value, and a centroid – the highest scored sentence in the collection of values.

The algorithm starts with a empty set of clusters. All sentences in the collection of texts are considered. The first sentence of the collection creates the first cluster. Then, each sentence in the collection of sentences is compared with the sentences already clustered. For each cluster, the similarity value is computed between the current sentence being compared and all the sentences in the collection of sentences of each cluster. The similarity value considered is the one that is the highest between the current sentence and all the sentences in the collection of sentences. Then, if the similarity value is higher than the already mentioned similarity threshold, the sentence will be added to the current cluster.

When a sentence is added to a cluster, its centroid must be updated. If the score of the sentence being added is higher than the centroid one, the newly added sentence becomes the centroid of the cluster. Also, each centroid is given an extra score value (0.1), which is subtracted from the sentences which are replaced as centroids.

Finally, if all the clusters have been considered, and the sentence was not added to any cluster, a new cluster with this sentence is created.

Once the procedure is finished, sentences with redundant information are grouped in the same cluster and the one with the highest score (the centroid) represents all sentences in the cluster.

Clustering Sentences by Keywords. After identifying similar sentences, the centroids of each similarity cluster are selected to be clustered by keywords. The sentences contained in the collection of values of each cluster are ignored, and will not be clustered in this phase, since they have been considered redundant.

Our system produces a generic summary, so it is not focused on a specific matter. Thus, the keywords that represent the global topic within the collection of texts are identified.

Keywords are determined in three steps. The first step selects the potential keywords in the complete collection of words present in all texts. Words such as clitics, and contractions are ignored, filtering out words with little discriminative power. The list with candidate keywords is constructed containing common and proper names, since these words are the ones that identify ideas or themes. To be added to this list, words are compared considering their lemmas, to ensure that the words in the collection are unique. The second step orders this list, containing the words which are candidates to be keywords, by the word score. Finally, a predefined number of keywords is retrieved in order to build the final set of keywords. We define k, the number of keywords, as $k = \sqrt{\frac{N}{2}}$, where N is the total number of words in the collection of documents.

Afterwards, sentences are clustered based on these keywords. The procedure starts by retrieving from the similarity clusters all its centroids. In this phase,

a cluster is identified by a keyword, and contains a centroid (a sentence), and a collection of values (the sentences related to the keyword). The algorithm that clusters sentences by keywords is an adapted version of the K-means algorithm [10], and follows the steps described below:

1. Choose the number of clusters, k, defined by the number of keywords previously selected;
2. Create the initial clusters, represented by each keyword obtained;
3. Consider each sentence:
   (a) Compute the occurrences of each keyword in the sentence;
   (b) Assign the current sentence to the cluster whose keyword occurs more often;
4. Recompute the cluster centroid. If the current sentence has more occurrences of the keyword than the previous centroid sentence had, the newly added sentence becomes the cluster centroid;
5. If the sentence does not contain any keywords, it is added to a specific set of sentences which do not have any keyword ("no-keyword" set);
6. Recompute the set of keywords if:
   (a) All the sentences have been considered;
   (b) The "no-keyword" set contains new sentences.
7. Repeat previous steps (2 – 6) while the "no-keyword" set of sentences remains different in consecutive iterations.

As in the similarity algorithm, each centroid is assigned with an extra score value. When the centroid is changed, the extra scores of the current centroid and of the previous centroid are updated. In addition, an extra score is also assigned to the sentences in the clusters that represent the original set of keywords.

Finally, sentences in the "no-keyword" set are ignored, while the ones that have been clustered are considered in the next phases.

3.3 Filtering

In this phase, the sentences that have been clustered by keywords are considered. The ones that have less than ten words are penalized and an extra score value is subtracted from their extra score. However, the sentences that have more than ten words are assigned with an extra score value.

The complete score, defined in Equation 4, is used to rank all the sentences:

$$completeScore = score_s + extraScore_s \tag{4}$$

So, sentences are ranked by their complete score, defining the order of the sentences to be chosen to be part of the final summary.

3.4 Reduction

The reduction process aims at reducing the original content to produce a summary containing simpler and more informative sentences. This phase comprises two steps: simplification and compression.

Simplification. Sentence simplification is performed by removing expressions or phrases whose removal is less detrimental to the comprehension of the text. Three types of structures are identified: sentence clauses, parenthetical phrases, explanatory or qualifying phrases; and apposition phrases, which are a specific type of parentheticals, composed by a noun phrase that describes, details or modifies its antecedent (also a noun phrase).

Parentheticals are typically enclosed either by parenthesis, or by commas or dashes. Appositions, in turn, can only be enclosed by commas or dashes and consist of a noun phrase. These phrases are candidates to removal.

In order to identify the passages, the sentence parse tree (built in the identification phase) is used. Once have been identified, all those passages are candidates to be removed.

Apposition and parenthetical phrases are removed from the sentence without further constraints, since they are considered less important than the rest of the sentences. Concerning sentence clauses, for each sentence, the simplification score (Equation 5) is computed. The simplification score is the sum of each word score entering the sentence, divided by the length of the sentence.

$$simplificationScore_s = \frac{\sum_{w \in s} score_w}{length_s} \tag{5}$$

For each sentence, the algorithm selects each clause separately. Afterwards, the simplification score is computed for both the main sentence and the new sentence. If the new sentence has a higher simplification score than the one of the main sentence, the main sentence is marked to be replaced by the sentence without this clause.

Compression. The compression rate previously defined (either given by the user or defined by default as 70% – the most commonly used default value – , that is the summary will contain 30% of the words in the collection of texts) is then applied to the collection of sentences, which are added to the final summary based on its total number of words. If the total words of the already added sentences reaches or surpasses the maximum number of words determined by the compression rate, no more sentences will be added, and the summary is created.

3.5 Presentation

Once the summary sentences have been identified, the summary is delivered to the user in the form of a text file.

4 Evaluation

In order to perform evaluation we used the *CSTNews* corpora [1] (described below), containing sets of texts in Portuguese and its corresponding ideal summaries. Concerning the evaluation itself, we compared SIMBA with GISTSUMM [12].

GISTSUMM is a summarizer built to deal with texts in Portuguese. It is based on the notion of gist, which is the most important passage of the text, conveyed by just one sentence, the one that best expresses the text's main topic. The system algorithm relies on this sentence to produce extracts. GISTSUMM is the only summarizer for Portuguese available on-line. Despite it has been built to produce summaries from a single-document, it also performs multi-document summarization by means of an option in its interface that allows to produce a summary from a collection of texts. GISTSUMM is used as a baseline for our work.

4.1 Corpus

CSTNews is an annotated corpus, whose texts were collected from five Brazilian newspapers. It contains 50 sets of news texts from several domains. Each set contains in average 3 documents which address the same subject, accompanied by its ideal summary. Table 1 summarizes the corpus data.

Table 1. Corpus statistics

Source texts statistics:

Total documents	Average documents	Total sentences	Average sentences	Total words	Average words
140	2.8	2,234	15.9	47,350	338.2

Ideal summaries statistics:

Total words	Average words	Average compression rate
6,859	137.18	85%

4.2 Results

First, for each set of the *CSTNews* corpora two summaries were created: one by SIMBA, and another by GISTSUMM.

In order to understand the differences that may lie between both summaries – the GISTSUMM and the SIMBA summary –, Table 2 details the phrases considered in the simplification process.

Hence, 35% of the sentences have an apposition phrase, and 17% have a parenthetical phrase. Thus, in the simplification process, we can work with around half of the sentences of the corpora. Appositions have in average five words, while parentheticals have three words in average. In fact, there is a large number of phrases to be examined. Considering both the apposition and the parenthetical phrases, they provide a total of 4,437 words.

The summaries have a compression rate of 85%, which means that the summary contains 85% of the words contained in the set of texts. This is the chosen value because it is the average compression rate of the ideal summaries.

Table 2. Phrases that are candidates to removal in the simplification process

Apposition phrases:

Tot. Sentences	Avg./document	Avg./sentence	Tot. Words	Avg. Words
774	15	0.35	3457	5

Parenthetical phrases:

Tot. Sentences	Avg./document	Avg./sentence	Tot. Words	Avg. Words
369	3	0.17	980	3

After both summaries have been built, they were compared with the corpus ideal summaries, using ROUGE [7]. In fact, a more precise metric of ROUGE was used, ROUGE-L (longest common subsequence), since it identifies the common subsequences between two sequences. The simplification process introduces gaps in the extracted sentences. This metric does not require consecutive matches but in-sequence matches, which reflect sentence level word order. This is a fairer metric, considering the type of arrangements made in the text. Precision, recall and f-measure values for each summarizer are detailed in Table 3.

Table 3. Multi-document evaluation metrics

	GISTSUMM	SIMBA
Precision	0.38469	0.47375
Recall	0.43616	0.45542
F-measure	0.40398	0.45980

By observing graph in Figure 1, we can see that SIMBA has a better performance than GISTSUMM.

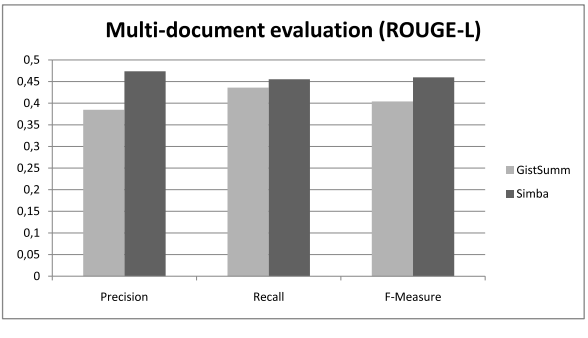

Fig. 1. ROUGE-L metric for GISTSUMM and SIMBA summaries

The complete summarization process has an overall better performance than GISTSUMM. SIMBA f-measure value overcomes the one of GISTSUMM in five percentage points, meaning that SIMBA summaries have more significant information than the ones of GISTSUMM.

The precision value obtained by SIMBA is very interesting. A high precision value means that considering all the information in the input texts, the retrieved information is relevant. Thus, obtaining the most relevant information in the sentences by discarding their less relevant data ensures that the summary contains indeed the most important information conveyed by each of its sentences.

The recall values of the two systems are closer than the ones concerning precision. Due to the simplification process, less in-sequence matches are likely to be found in SIMBA summaries, when compared to the ideal summaries. Thus, its recall values should be similar or even decrease. Still, considering both recall values, we can conclude that SIMBA summaries cover more significant topics than the summaries produced by GISTSUMM.

Thus, considering the evaluation results, we can conclude that this approach produces better summaries when compared to the one used by GISTSUMM.

5 Concluding Remarks

The quality of an automatic summary can be improved by (1) performing specific multi-document tasks – as removing the redundant information, or considering all the texts in each set as a single information source; and (2) executing an algorithm that seeks to optimize the content selection, combined with a simplification process that removes less relevant content, allowing the addition of new relevant information.

Despite the core algorithm is language independent, this system uses language specific tools that aim to improve not only the content selection, but also the general quality of a summary produced from a collection of texts written in Portuguese.

The multi-document summarizer presented relies on statistical features to perform summarization of a collection of texts in Portuguese. On the one hand, the double-clustering approach identifies the most relevant sentences in the texts. On the other, the simplification process removes from those sentences the information that adds no crucial content to the final summary. As the encouraging results, shown in the final evaluation, state, the combination of these two approaches produces highly informative summaries. In fact, these summaries not only preserve the idea conveyed by the collection of texts to be summarized, but also they cover the most significant topics mentioned in that collection.

References

1. Aleixo, P., Pardo, T.A.S.: Cstnews: Um córpus de textos jornalísticos anotados segundo a teoria discursiva multidocumento cst (cross-document structure theory). Tech. rep., Universidade de São Paulo (2008)

2. Branco, A., Silva, J.: A suite of shallow processing tools for portuguese: Lx-suite. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2006 (2006)
3. Carbonell, J.G., Goldstein, J.: The use of MMR, diversity-based reranking for reordering documents and producing summaries. In: Research and Development in Information Retrieval, pp. 335–336 (1998)
4. Dice, L.R.: Measures of the amount of ecologic association between species. Ecology 26(3), 297–302 (1945)
5. Edmundson, H.P.: New methods in automatic extracting. J. ACM 16(2), 264–285 (1969)
6. Jaccard, P.: Nouvelles recherches sur la distribution florale. Bulletin de la Sociète Vaudense des Sciences Naturelles 44, 223–270 (1908)
7. Lin, C.Y.: Rouge: A package for automatic evaluation of summaries. In: Marie-Francine Moens, S.S. (ed.) Text Summarization Branches Out: Proceedings of the ACL 2004 Workshop, pp. 74–81. ACL, Barcelona (2004)
8. Lin, C.Y., Hovy, E.: From single to multi-document summarization: A prototype system and its evaluation. In: Proceedings of the ACL, pp. 457–464. MIT Press (2002)
9. Luhn, H.P.: The automatic creation of literature abstracts. IBM Journal of Research and Development 2 (1958)
10. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297. University of California Press (1967)
11. Mani, I.: Automatic Summarization. Benjamins Pub. Co., Amsterdam (2001)
12. Pardo, T.A.S., Rino, L.H.M., das Graças Volpe Nunes, M.: GistSumm: A Summarization Tool Based on a New Extractive Method. In: Mamede, N.J., Baptista, J., Trancoso, I., Nunes, M.d.G.V. (eds.) PROPOR 2003. LNCS, vol. 2721, pp. 210–218. Springer, Heidelberg (2003)
13. Radev, D.R., Jing, H., Budzikowska, M.: Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies. In: Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization, NAACL-ANLP-AutoSum 2000, pp. 21–30. ACL (2000)
14. Silva, J., Branco, A., Castro, S., Reis, R.: Out-of-the-Box Robust Parsing of Portuguese. In: Pardo, T.A.S., Branco, A., Klautau, A., Vieira, R., de Lima, V.L.S. (eds.) PROPOR 2010. LNCS, vol. 6001, pp. 75–85. Springer, Heidelberg (2010)
15. White, M., Korelsky, T., Cardie, C., Ng, V., Pierce, D., Wagstaff, K.: Multidocument summarization via information extraction. In: HLT 2001: Proceedings of the First International Conference on Human Language Technology Research, pp. 1–7 (2001)