Document Clustering and Text Summarization

Joel Larocca Neto Alexandre D. Santos
Celso A.A. Kaestner Alex A. Freitas

Pontificia Universidade Catolica do Parana
Postgraduate Program in Applied Computer Science
Rua Imaculada Conceição 1155
Curitiba - PR, 80215-901. Brazil
{joel, denes, kaestner, alex}@ppgia.pucpr.br
http://www.ppgia.pucpr.br/~alex

Abstract

This paper describes a text mining tool that performs two tasks, namely document clustering and text summarization. These tasks have, of course, their corresponding counterpart in “conventional” data mining. However, the textual, unstructured nature of documents makes these two text mining tasks considerably more difficult than their data mining counterparts. In our system document clustering is performed by using the Autoclass data mining algorithm. Our text summarization algorithm is based on computing the value of a TF-ISF (term frequency – inverse *sentence* frequency) measure for each word, which is an adaptation of the conventional TF-IDF (term frequency – inverse *document* frequency) measure of information retrieval. Sentences with high values of TF-ISF are selected to produce a summary of the source text. The system has been evaluated on real-world documents, and the results are satisfactory.

1. Introduction

Text mining is an emerging field at the intersection of several research areas, including data mining, natural language processing, and information retrieval [Feldman & Dagan 95], [Feldman & Hirsh 96], [Landau et al. 98]. The main differences between data mining and text mining is as follows. Data mining usually deals with structured data sets - typically in the first normal form, in the terminology of relational databases. By contrast, text mining deals with unstructured or semi-structured data, namely the text found in articles, documents, etc.

In addition to the availability of little (if any) structure in the text, there are other reasons why text mining is so difficult. The concepts contained in a text are usually rather abstract and can hardly be modeled by using conventional knowledge representation structures. Furthermore, the occurrence of synonyms (different words with the same meaning) and homonyms (words with the same spelling but with distinct meanings) makes it difficult to detect valid relationships between different parts of the text.

In this paper we describe a system that performs two important text mining tasks: document clustering and text summarization. These tasks have, of course, their corresponding counterpart in data mining. However, the textual, unstructured nature of documents makes these two text mining tasks considerably more difficult than their data mining counterparts.

Among the several obstacles to be faced in text mining, two are the focus of this paper. The first is the fact that the amount of preprocessing that has to be applied to a document, for text mining purposes, is usually even greater than the amount of preprocessing required for data mining. As will be seen later, our system uses a combination of several techniques to achieve an effective text preprocessing for text mining.

Second, evaluating the quality of the knowledge discovered by a text mining system is even more subjective than evaluating the quality of the knowledge discovered by a data mining system. Our system takes this account, and it endeavors to give the user comprehensible knowledge. In particular, once document clustering is done, our system uses the main keywords of the documents of a cluster to describe that cluster. These keywords can be regarded as an ultra compact “summarization” of the contents of that cluster. In addition, we also added to our system a text summarization algorithm, which can be effectively used to summarize individual documents. We believe these two kinds of summarization are complementary to each other, and together they significantly help the user to better understand the information stored in the documents being mined.

The system has been evaluated on real-world documents and the results are satisfactory, as will be described later.

This paper is organized as follows. Section 2 presents the overall architecture of the system. Section 3 discusses the text preprocessing techniques used in the system. Section 4 explains the method used in our system for document clustering, and section 5 presents the results of experiments evaluating the performance of the system in this task. Section 6 introduces the method developed for text summarization, and section 7 presents the results of experiments evaluating the performance of the system in this task. Finally, section 8 concludes the paper and discusses future research.

2. Overall System Architecture

The overall architecture of the system is shown in Figure 1. The input of the system is a set of documents. The system applies several preprocessing methods to the input documents, namely case folding, stemming, removal of stop words and n-grams - see section 3. Then the user can run two kinds of text mining algorithms on the preprocessed documents: a document clustering algorithm or a text summarization one.

The document clustering algorithm uses the Autoclass algorithm [Cheeseman et al. 88], [Cheeseman & Stutz 96] - see Section 4. The output of this algorithm is shown to the user together with a list of keywords characterizing each cluster. These keywords help the user to quickly grasp the main topics discussed in the documents belonging to a given cluster. These keywords can be regarded as a kind of “ultra-summary” of the contents of all the documents in the cluster.

The system can also summarize individual documents. In this case it runs a summarization algorithm that extracts the most relevant sentences from a document. The relevance of each sentence is determined by computing the average relevance of all the words in the (preprocessed) sentence. The method is described in detail in section 6.

Note that the document clustering and the text summarization algorithms can be used together in a synergistic, complementary way. For instance, the user can first perform a clustering of some documents, to get an initial understanding of the document base. Then, supposing the user finds a particular cluster interesting, (s)he can perform a summarization of the documents in that cluster.

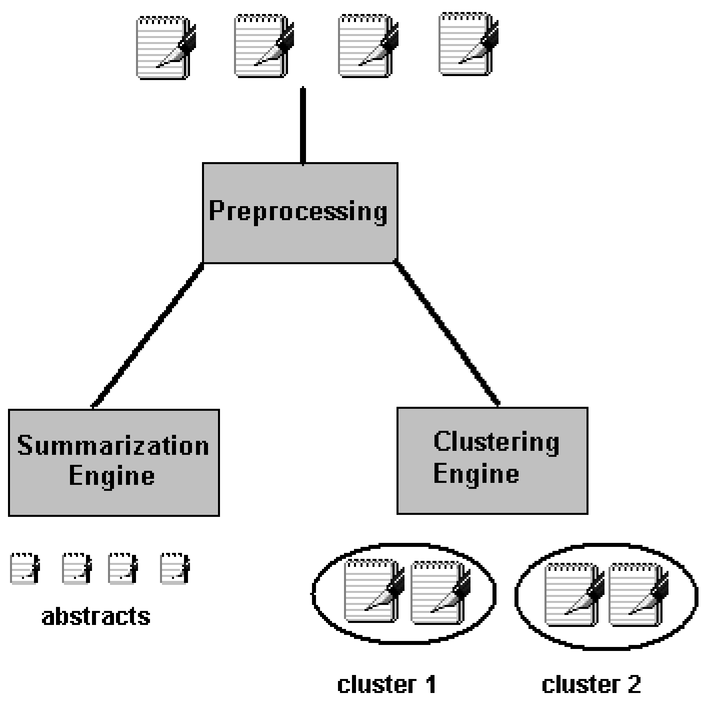


Figure 1: The overall architecture of the system.

2.1. Document Representation

In data mining the choice of data representation is usually straightforward. In general, data is represented as a set of records, where each record is a set of attribute values.

In text mining the situation is more complex. Due to the little (if any) structure of texts, the choice of data representation is not so obvious. The choice of a good data representation involves a trade-off between its ability to preserve the information contained in the original document and its computational efficiency - i.e. the computational cost required to transform the original document into the data representation, the computational cost to manipulate the data representation, etc.

Probably the most used data representation - particularly in information retrieval - is the bag-of-words [Joachims 96]. This representation keeps track of the words occurring in the document and it is conceptually analogous to the attribute-value representation in data mining. Hence, each word of the document is considered to be an “attribute”, whose value is the number of times that the word occurs in the document. Each document is in turn considered to be a “record”.

The main advantages of this representation are its conceptual simplicity and its relative computational efficiency. Its main disadvantage is the fact that it loses important information about the original text. In particular, it contains no information about the order of the words in the text, no information about the frontiers between sentences or paragraphs, etc.

We have chosen to use a vectorial document representation [Salton & Allan 94]. The basic idea is to represent documents as multi-dimensional vectors. In a simple case, each vector coordinate represents the number of times that a word occurs in the document. A document vector can have hundreds of dimensions, most of them taking on the value 0 for most documents.

The basic idea of this representation is conceptually similar to the bag-of-words representation. One important difference, however, is that the vector representation suggests a natural measure of similarity between two documents: the angle between their vectors. More precisely, the smaller the angle between two document vectors, the more similar the two documents.

In general, in a vectorial representation the value of each vector coordinate can be an elaborate measure of a word's relevance, rather than simply the number of times that a word occurs in the document. We use a vectorial representation where the value of each coordinate is given by the *TF-IDF* (term frequency - inverse document frequency) of the corresponding word [Joachims 96], [Salton & Buckley 88] as explained in the following.

The *term frequency* of a word *w* in a document *d*, denoted *TF(w,d)*, is the number of times that the word *w* occurs in document *d*. The higher the *TF(w,d)*, the more the word w is representative (or characteristic) of document d.

The *document frequency* of a word *w*, denoted *DF(w)*, is the number of documents in which *w* occurs. The *inverse document frequency* of a word *w*, denoted *IDF(w)* is given by the formula:

$$IDF(w) = 1 + \log(|D| / DF(w)) .$$

Hence, the *IDF(w)* of a word *w* is low if this word occurs in many documents, indicating that the word has little document-discriminating power. On the other hand, the *IDF(w)* of a word *w* is high if this word occurs in few documents, indicating that the word has a great document-discriminating power.

Of course, we want words that have a high TF and a high IDF. We can express this desire into a single formula, by saying that we want words with a high value of the following *TF-IDF(w,d)* measure:

$$TF\text{-}IDF(w,d) = TF(w,d) * IDF(w) .$$

3. Text Preprocessing: Case Folding, Stemming, Removal of Stop Words and N-Grams.

One of the major problems in text mining is that a document can contain a very large number of words. If each of these words is represented as a vector coordinate, the number of dimensions would be too high for the text mining algorithm. Hence, it is crucial to apply preprocessing methods that greatly reduce the number of dimensions (words) to be given to the text mining algorithm. (Of course, reducing the number of words is a requirement not only in the vector representation, but also in virtually any other text representation, including the bag-of-words one.)

In addition, it is important that the preprocessing method be robust, i.e. able to cope with noisy text containing grammatical and typographical errors.

Our system can apply several preprocessing methods to the original documents, namely case folding, stemming, removal of stop words and n-grams [Witten et al. 94]. Each of these methods is briefly discussed next.

Case folding consists of converting all the characters of a document into the same case format, either the upper-case or the lower-case format. For instance, the words "the", "The", "tHe" "THe", "thE", "ThE", "tHE" "THE" would all be converted to the standard lower-case format "the".

Stemming consists of converting each word to its stem, i.e. a neutral form with respect to tag-of-speech and verbal/plural inflections. In essence, to get the stem of a word it is necessary to eliminate its suffixes representing tag-of-speech and/or verbal/plural inflections. For instance, the words “compression” and “compressed” would both be converted to their stem “compress”. Stemming algorithms usually incorporate a great deal of linguistic knowledge, so that they are language-dependent. We have used Porter’s algorithm [Porter 80], originally developed for the English language.

Stop words are words that occur very frequently in a document. Since they are so common in many documents, they carry very little information about the contents of a document in which they appear. Therefore, it is usually a good idea to remove them from the document representation. For instance, “can”, “will”, “do” and “does” are typical stop words. Our system uses a list of 524 stop words, obtained from the source code of the library BOW, developed by Carnegie Mellon University.

The N-gram representation is an alternative to stemming and stop word removal. An n-gram is an N-character slice of a longer string [Cavnar 94]. For instance, the word DATA can be represented by the tri-grams *_DA, DAT, ATA, TA_*, or by the quad-grams *_DAT, DATA, ATA_*, where the underscore character represents a leading or trailing space. In comparison with stemming and stop word removal, the N-gram representation has the advantage of being more robust – less sensitive to grammatical and typographical errors – and requiring no linguistic preparations – which makes it more language independent. However, the n-grams representation is not so effective in reducing the number of dimensions (words) to be given to the text mining algorithm. This benefit is better achieved by stemming and stop word removal.

To summarize, our system performs text preprocessing in two steps. First, it applies case folding to the text. Second, it applies one of the two preprocessing methods: (a) stemming and stop word removal; or (b) n-gram representation. Hence, the user can experiment with both these methods separately and choose the one most suitable for his/her needs.

4. Document Clustering

We have used the Autoclass algorithm to perform document clustering. The task of document clustering, as well as the vast majority of the data mining and text mining tasks, can be cast as a state-space search. Hence, a clustering algorithm carries out a search in the space of possible candidate solutions - i.e. different partitions of the data into clusters. Most clustering algorithms use a single model to represent the possible candidate solutions, so that the search space consists of different parameters of that model.

By contrast, Autoclass considers a search space consisting of different models and parameters, by using bayesian techniques to evaluate the quality of a candidate solution. In order to evaluate the quality of a given partition of the data into clusters, Autoclass uses a criterion that considers the trade-off between the predictive power of a given partition and the complexity of its description. Hence, Autoclass favors simple descriptions. In addition, Autoclass has the advantage that it does not require that the number of clusters be specified a priori.

In the model space searched by Autoclass, each feature is associated with a given probabilistic model. In our system the features were modeled as a conditionally-independent normal distribution [Kroon et al. 96].

The document clustering algorithm implemented in our system can currently work with two data representations: a simplified bag-of-words representation, where each word is represented by a binary attribute (indicating whether or not the word occurs in the corresponding document) and a vector representation using TF-IDF values as vector coordinate values -

described in section 2.1. This latter is the default representation in our system. The former was included to allow a comparison with another document clustering system reported in the literature, as will be seen in section 5.

As mentioned before, in order to help the user to quickly grasp the contents of each cluster, our system associates each cluster with a small set of keywords which frequently occur in the documents belonging to that clusters. For each cluster, the keywords in question are the k words with the largest value of the TF-IDF measure for that cluster, where k is an integer-valued, user-defined parameter. In the experiments reported in this paper we have set k = 8.

5. Computational Results for Document Clustering

Our system was developed in Microsoft Visual C++ 5.0 for the platform Windows 95/NT. The document clustering method implemented in our system requires the specification of some parameters of the Autoclass algorithm, namely the likely error of each attribute (word occurring in some document) and the probabilistic model associated to each attribute. In the experiments reported in this paper, all the attributes were assigned a likely error of 0.1% and, as mentioned above, all attributes were modeled as a conditionally-independent normal distribution. This choice of attribute models corresponds to the value *single_normal_cn* of the parameter *model* of the Autoclass algorithm.

We have performed two experiments to evaluate the document clustering method implemented in our system.

Experiment 1.

We have compared the document clustering performed by our system with the document clustering performed by a SOM (self-organized map) neural network. This latter system, hereafter called the SOM system for short, is described in [Merkl 97]. It produces a kind of topological map of the input patterns. Hence, the output clusters are shown to the user in graphical form.

In order to compare our system with the SOM system we gave the former the same set of documents as the one used to evaluate the latter. In this data set, each document was a page of the manual of the Class Library NIH, called NIHCL.

Our system found nine clusters in the data. The keywords describing each cluster are shown in Figure 2. Some clusters are very coherent. For instance, cluster 0 contains all the classes of the library NIH that implement some I/O device, as can be observed in Figure 3. As another example, cluster 1 contains all the classes of the library NIH that implement basic data types, such as *integer*, *float*, *date* or *time*, as can be observed in Figure 4.

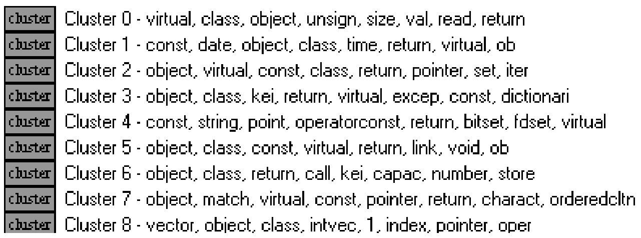
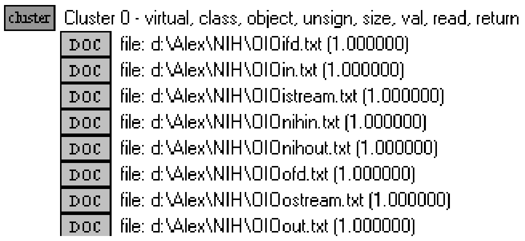

cluster Cluster 0 - virtual, class, object, unsign, size, val, read, return
cluster Cluster 1 - const, date, object, class, time, return, virtual, ob
cluster Cluster 2 - object, virtual, const, class, return, pointer, set, iter
cluster Cluster 3 - object, class, kei, return, virtual, excep, const, dictionari
cluster Cluster 4 - const, string, point, operatorconst, return, bitset, fdset, virtual
cluster Cluster 5 - object, class, const, virtual, return, link, void, ob
cluster Cluster 6 - object, class, return, call, kei, capac, number, store
cluster Cluster 7 - object, match, virtual, const, pointer, return, charact, orderedcltn
cluster Cluster 8 - vector, object, class, intvec, 1, index, pointer, oper

Figure 2: Clusters found by our system in the NIHCL corpus.


```
cluster Cluster 0 - virtual, class, object, unsign, size, val, read, return
    DOC file: d:\Alex\NIH\OIOifd.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOin.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOistream.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOnihin.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOnihout.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOofd.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOostream.txt (1.000000)
    DOC file: d:\Alex\NIH\OIOout.txt (1.000000)
```
Figure 3: Contents of cluster 0 found by our system in the NIHCL corpus.

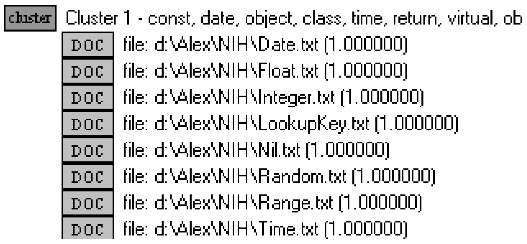

```
cluster Cluster 1 - const, date, object, class, time, return, virtual, ob
    DOC file: d:\Alex\NIH\Date.txt (1.000000)
    DOC file: d:\Alex\NIH\Float.txt (1.000000)
    DOC file: d:\Alex\NIH\Integer.txt (1.000000)
    DOC file: d:\Alex\NIH\LookupKey.txt (1.000000)
    DOC file: d:\Alex\NIH\Nil.txt (1.000000)
    DOC file: d:\Alex\NIH\Random.txt (1.000000)
    DOC file: d:\Alex\NIH\Range.txt (1.000000)
    DOC file: d:\Alex\NIH\Time.txt (1.000000)
```
Figure 4: Contents of cluster 1 found by our system in the NIHCL corpus.

Figure 5 shows the clusters output by the SOM system for this data set. Note that there is no clear frontier between "clusters". [Merkl 97] has also extended the basic SOM learning algorithm to capture the movements of the neural-network weight vector in the weight space during training. The resulting algorithm was called Adaptive Coordinates. It has the advantage of facilitating the identification of the clusters discovered by the system, as illustrated in Figure 6.

One can see that is easier to distinguish clusters in the graphical representation of Figure 6 than in the graphical representation of Figure 5. However, even in the representation of Figure 6 the frontiers between clusters is not very clear. In order to improve the graphical representation of Figure 6, we have extended it with a drawing of the frontiers of 9 clusters, since our system (using AutoClass) found 9 clusters. The result is shown in Figure 7.

Although the clusters shown in Figure 7 are not exactly the same as the clusters found by our system, the two sets of clusters are reasonably similar to each other. The SOM method with Adaptive Coordinates has the advantage of showing the clusters in a graphical format, whereas our system has the advantages of providing a precise specification of the frontiers between clusters (which is somewhat subjective in Figure 6) and providing a small set of keywords to describe, in an ultra-summarized form, the contents of each cluster.

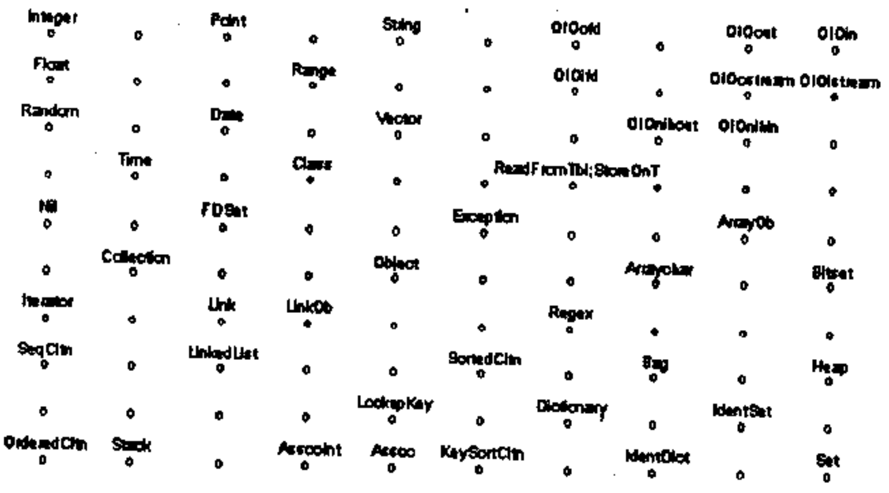


Figure 5: Clusters output by a standard SOM (self-organized map) neural network.

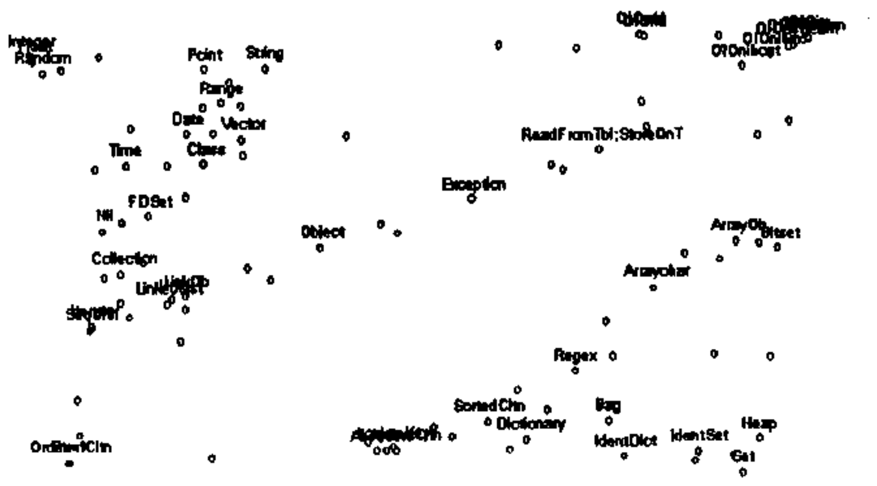


Figure 6: Clusters output by a SOM neural network with adaptive coordinates.

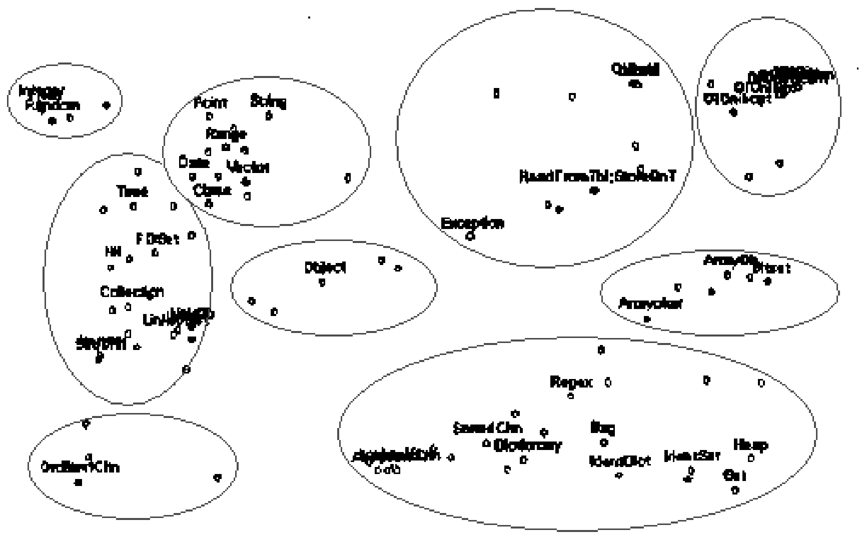


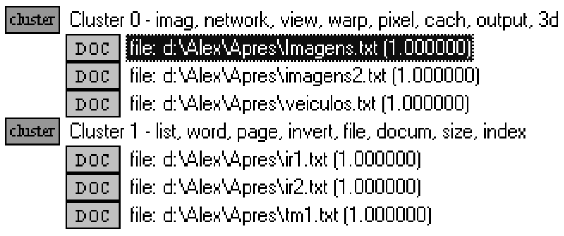
Figure 7: Drawing cluster frontiers in the output of a SOM neural network with adaptive coordinates.

Experiment 2

This experiment used a small set of six articles about text mining, information retrieval and image processing. The documents were represented in vectorial form by using boolean values, rather than TF-IDF weights.

In this data set our system found only two clusters. Figure 8 shows the keywords describing each cluster as well as the contents of each cluster. Once again, one can see that the system found very coherent clusters.

Cluster 0 contains documents related to image processing. This conclusion is strongly supported by several keywords used to describe that cluster, namely *imag*, *pixel*, *3d* and *warp*. Cluster 1 contains documents related to text mining and information retrieval., This conclusion is strongly supported by the keywords used to describe that cluster, namely *word*, *page* and *docum*.



```
cluster Cluster 0 - imag, network, view, warp, pixel, cach, output, 3d
        DOC file: d:\Alex\Apres\Imagens.txt (1.000000)
        DOC file: d:\Alex\Apres\imagens2.txt (1.000000)
        DOC file: d:\Alex\Apres\veiculos.txt (1.000000)
cluster Cluster 1 - list, word, page, invert, file, docum, size, index
        DOC file: d:\Alex\Apres\ir1.txt (1.000000)
        DOC file: d:\Alex\Apres\ir2.txt (1.000000)
        DOC file: d:\Alex\Apres\tm1.txt (1.000000)
```

Figure 8: Clusters found by our system in the text mining, information retrieval and image processing corpus.

6 Text Summarization

As mentioned before, the summarization algorithm developed in our system extracts the most relevant sentences from a document. This kind of summarization algorithm can be categorized as an extractive one, since the output of the algorithm can contain only sentences that are present in the document.

By contrast, some summarization algorithms are capable of producing summaries that contain not only sentences that are present in the document but also new automatically-constructed phrases that are added to the summary to make it more intelligible. In theory, this functionality makes the summarization algorithm more powerful and improves the comprehensibility of the output summary. In practice, the automatic construction of phrases is a quite difficult task and there is no guarantee that the new phrases will be really meaningful for the user.

Hence, in the current version of our system we have opted to develop a safer extractive-summarization algorithm. The automatic construction of new phrases might be investigated in future research. We now describe in detail the text summarization algorithm developed in our system.

Recall that the document has already undergone some text preprocessing, as described in section 3. Hence, when we refer to a word we are actually referring to a preprocessed word, and not to the original word of the document.

The first step is to separate the sentences of a document. The end of a sentence is defined as a "." (full stop) followed by a space or new line character. Note that the existence of a "." is not sufficient by itself to indicate the end of a sentence, since this oversimplified criterion would wrongly detect several sentences in cases where the "." is merely separating parts of a textual string - e.g. in URLs such as http://www.ppgia.pucpr.br.

Once all the sentences of the document are identified, each sentence is represented as a vector of *TF-ISF* (term frequency - inverse sentence frequency) weights. The computation of the *TF-ISF* for each word is similar to the computation of *TF-IDF* described in section 2.1. The differences are as follows. The notion of "document" in section 2.1 is replaced by the notion of sentence. Similarly, the "number of documents" D in section 2.1 is replaced by the number of sentences S in the document. Hence, the *TF-ISF* measure of a word w in a sentence s, denoted *TF-ISF(w,s),* is computed by the formula:

$$TF\text{-}ISF(w,s) = TF(w,s) * ISF(w)$$

where the term frequency *TF(w,s)* is the number of times that the word *w* occurs in sentence *s*, and the inverse sentence frequency *ISF(w)* is given by the formula:

$$ISF(w) = \log(|S| / SF(w)) ,$$

where the sentence frequency *SF(w)* is the number of sentences in which the word w occurs.

For each sentence s, the average *TF-ISF* weight of the sentence, denoted *Avg-TF-ISF(s)* is computed by calculating the arithmetic average of the *TF-ISF(w,s)* weight over all the words w in the sentence, that is:

$$Avg\text{-}TF\text{-}ISF(s) = \sum_{i=1}^{W(s)} TF\text{-}ISF(i,s) / \mathrm{W(s)}$$

where W(s) is the number of words in the sentence s.

Once the value of the *Avg-TF-ISF(s)* measure is computed for each sentence s, the final step is to select the most relevant sentences, i.e. the ones with the largest values of the *Avg-TF-ISF(s)* measure. In the current version of our system this is done as follows.

The system finds the sentence with the largest *Avg-TF-ISF(s)* value, called the *Max-Avg-TF-ISF* value. The user specifies a threshold on the percentage of this value, denoted *percentage-threshold*. This percentage threshold is used to computed the *Threshold-TS-ISF* value, given by:

$$Threshold\text{-}TS\text{-}ISF = percentage\text{-}threshold * Max\text{-}Avg\text{-}TF\text{-}ISF \text{ value}$$

The output of the text summarization algorithm consists of all sentences s whose *Avg-TF-ISF(s)* value is greater than or equal to *Threshold-TS-ISF*. Although *TF-IDF* weights are well-known in information retrieval, to the best of our knowledge the idea of *TF-ISF* weights is introduced in this paper.

7. Computational Results for Text Summarization

In this section we describe the results of an experiment carried out to evaluate our text summarization algorithm. In order to make our experiments more relevant, we have used the evaluation framework of a large scale project for evaluating text summarization algorithms: the TIPSTER Text Summarization Evaluation Conference (SUMMAC) – hereafter referred to as the SUMMAC project for short [Mani et al. 98]. This is an important point, since there is no standard measure of summary quality. To quote [Mani et al. 98, p. 3]: "Text summarization is still an emerging field, and serious questions remain concerning the appropriate methods and types of evaluation."

Summary quality measures can be roughly categorized into intrinsic and extrinsic measures. Intrinsic measures evaluate the quality of a summary directly based on analysis of the summary, whereas extrinsic measures evaluate the quality of a summary based on how it affects the completion of some other task (e.g. determining the relevance of a document to a topic, in some task related to text categorization). Another categorization can be made between objective or subjective measures of summary quality.

To evaluate our text summarization algorithm, we have compared the summaries produced by our system against the summaries produced by algorithms performing the *adhoc* task of the SUMMAC project. This task is defined as follows [Mani et al. 98, p. 4]:

"The real-world activity represented by this task is that of an analyst conducting full-text searches using an IR [information retrieval] system who must determine quickly and accurately the relevance of a retrieved document. Here the topic is provided as an input to the summarization system, and the evaluation seeks to determine whether the ... summary is effective in determining the relevance of the full-text source to a topic. Given a document (which could be a summary or a full-text source – the subject isn't told which), and a topic description, the human subject determines whether the document seen is relevant to the topic. The accuracy of the subject's relevance assessment decision is measured in terms of separately obtained 'ground-truth' judgments of the relevance of the full-text source to the topic, which were separately obtained from the Text Retrieval (TREC) ... conferences..."

It should be noted that this definition implies that, in this task, summary evaluation is both extrinsic and subjective. Ideally, we should evaluate the summaries produced by our algorithm using the same performance metrics as the above task of the SUMMAC project. In this case our algorithm could be directly compared against the 16 summarization algorithms that participated in the *adhoc* task of that project. In practice, unfortunately, this is unfeasible, since the conference is over, and we do not have access to all the infra-structure of the conference (including the human analysts who assessed the relevance of all the summaries produced by each of the algorithms competing in the conference).

Therefore, we have carried out our own subjective comparison of the summaries produced by our system against the summaries produced by the CGI/CMU system. This system was chosen mainly because, overall, it produced the best results of the SUMMAC evaluation in the above-described *adhoc* task.

We have compared the summaries produced by the two systems – our system and the CGI/CMU system - for several source texts. Due to space limitation, we show below the summaries produced by the two systems for a single source text from the Wall Street Journal. (This source text has the code number WSJ910130 in the SUMMAC project.)

The input to our system was a list of all the sentences in the source text, i.e. the detection of the end of a sentence was already performed in the SUMMAC project. Hence, the same list of sentences was given as input to both our system and the CGI/CMU system.

We did experiments with two alternative text preprocessing methods: (a) stemming and stop word removal; and (b) n-gram representation – see section 3. We achieved better results with the n-gram representation, and the results reported below were produced with the quad-gram representation method.

The summaries produced by our system and by the CGI/CMU system are shown in Figures 9 and 10 respectively. The summary shown in Figure 9 was produced by setting the *Threshold-TS-ISF* to 83% - this value was empirically determined. The reader can read both summaries and judge by himself/herself which one (if any) is the best. In each Figure the summary is presented as a sequence of sentences. Each sentence is presented as a separate paragraph, regardless of the actual paragraph in which the sentence was included in the source document. Right after the end of a sentence there is a number between square brackets. This is the number of the sentence in the source text.

Our own subjective evaluation is that both summaries are high-quality ones, capturing the main ideas in the source text (which unfortunately cannot be shown here due to space limitations).

Both systems achieved a good compression rate (ratio of summary length to source length). Our system produced a summary with 10 sentences, whereas the CGI/CMU system produced a summary with 16 sentences. The source text has 75 sentences.

Clearly, none of the two systems produced perfect summaries, which is normal, considering the high degree of difficulty associated with the text summarization task. For instance, our system included in the summary the sentence "David Rogers and David Wessel contributed to this article. [73]," which would probably be considered irrelevant by most readers. As another example, the CGI/CMU system included in the summary the sentence "They are the birthright of every American. [30],” which does not make sense in the summary, since we cannot know (reading only the summary) which birthrights the text is talking about. (Technically speaking, this is known as the *dangling-anaphora* problem.)

In an attempt to capitalize on the spectacular success U.S. missile defenses have had in the war, he announced that he is altering the Strategic Defense Initiative to focus on the kind of "limited ballistic missile strikes" Iraq has been launching on Saudi Arabia and Israel. [5].

He said he will send Congress a list of $20 billion in specific federal grants to states from which the administration and Congress would jointly select the grant programs to be killed. [19].

"Good health care is every American's right, and every American's responsibility" Mr. Bush asserted, though he offered no plan to provide health care to those without insurance. [31].

"We can find meaning and reward by serving some purpose higher than ourselves a shining purpose, the illumination of a thousand points of light." [36].

The proposal, which the administration intends to announce next week, will call for changing the deposit insurance system, augmenting bank capital, allowing banks to combine with other businesses within new financial services holding companies and simplifying bank regulation. [38].

"We cannot oppose repression in one place and overlook it in another." [42].

Mr. Mitchell also specifically attacked the president's proposal to cut the capital-gains tax rate, saying it would chiefly benefit those with incomes over $200,000 a year, and he pointedly noted that "not many kids whose families earn more than $200,000 a year volunteer to join the Army." [50].

The war is "on course," he declared, adding: "Iraq's capacity to sustain war is being destroyed. [62].

He said dialogue between the two countries is important to encouraging increasing democracy in the U.S.S.R. And Mr. Bush also said that in his recent talks with Soviet leadership he met with new Soviet Foreign Minister Alexander Bessmertnykh on Monday he has been given "representations" that, if acted upon, could lead to the withdrawal of some Soviet forces from the Baltics and "a reopening of dialogue" between the Soviet government and the Baltic states. [72].

David Rogers and David Wessel contributed to this article. [73].

Figure 9: Example of summary produced by our system.

WASHINGTON Calling the war against Iraq part of the "hard work of freedom" Americans are obliged to do, President Bush promised the nation victory in the Persian Gulf. [1]

But he also used it to emphasize an often-repeated appeal that the nation's other business go on. [9]

To that end he devoted more than half the speech to a discussion of domestic concerns and said he will propose a handful of new programs, including an overhaul of the banking system and a National Energy Strategy to promote energy efficiency, development, and conservation. [10]

However, the president did revive proposals to foster long-term growth, such as lower capital-gains tax rates and tax incentives for personal savings and research and development.[14]

He said he will send Congress a list of $20 billion in specific federal grants to states from which the administration and Congress would jointly select the grant programs to be killed. [19]

Saying that if the nation can "selflessly" confront Iraq for the "sake of good....[23]

then surely we can make this land all that it should be. [24]

If anyone tells you America's best days are behind her, they're looking the wrong way," Mr. Bush said. [25]

"They are the birthright of every American." [30]

Instead, he promised new programs for preventive health care, which aides said include money to reduce infant mortality in big cities and a new program to detect breast and cervical cancer.[32]

"We can find meaning and reward by serving some purpose higher than ourselves a shining purpose, the illumination of a

thousand points of light." [36]

The president also addressed the credit crunch by calling for lower interest rates and urging banks to make more loans. [39]

Delivering the Democratic response, Senate Majority Leader George Mitchell of Maine sought to put aside past differences over U.S. policy in the Gulf, but his remarks were implicitly critical of the administration for not doing more to address

wrongs elsewhere in the world as well as at home. [40]

Mr. Bush devoted much of his speech to a general, almost philosophical, explanation to Americans of why he thinks U.S. armed forces should be leading the fight to evict Iraqi troops from Kuwait. [51]

Mr. Bush argued that the U.S. has a special responsibility to accomplish such lofty goals because of America's powerful position in the world. [55]

"But the fact that all voices have the right to speak out is one of the reasons we've been united in purpose and principle for 200 years." [69]

Figure 10: Example of summary produced by the CGI/CMU system.

8. Conclusions and Future Research

We have described a text mining tool that integrates information retrieval and data mining techniques. Our system uses information retrieval techniques to perform text preprocessing and to represent the preprocessed text. Once the input documents have been preprocessed, the system can perform two text mining tasks: document clustering and text summarization.

Document clustering is performed by using the Autoclass data mining algorithm. Our system was compared with a Self-Organized Map (SOM) neural network algorithm. The results achieved by our system can be interpreted in a more objective way than the results achieved by the SOM system. This latter presents its results in a graphical form (a topological map) which does not tell the user how many clusters were found nor does it allow the user to clearly distinguish frontiers between clusters. Hence, different users might think that a given document belongs to different clusters. In addition, our system assigns to each cluster a small set of keywords, which can be regarded as an ultra-summarization of the contents of the cluster. This helps the user to quickly grasp the topics of the documents belonging to each cluster.

We have also developed a text summarization algorithm that capitalizes on the vectorial document representation used in our system. In order to evaluate the relevance of each sentence, this algorithm essentially uses only the value of the *TF-ISF* measure (an adaptation of the *TF-IDF* measure) for each word, which was already calculated when the original document was transformed into a vectorial representation in the preprocessing stage. Therefore, this algorithm is quite fast. In addition, since the *TF-ISF* measure relies only on information about the frequency of words in documents, avoiding the need for any semantic information about the words, this algorithm can be used to summarize texts in languages other than English. (However, it should be noted that some of the text preprocessing algorithms used in our work are language-dependent, and therefore they would have to be replaced in this case. In practice, we can avoid this by using the n-gram representation offered by our system, which is, to a large extent, language-independent.)

The current version of our text mining tool can be improved along several research directions. Probably the most important future research is to extend the functionality of the tool so that it can perform other text mining tasks, mainly text categorization and the discovery of association rules between keywords of documents.

In addition, we are currently developing other techniques for text summarization, based on the use of genetic algorithms.

References

[Cavnar 94] Cavnar, W.B. Using An N-Gram-Based Document Representation With A Vector Processing Retrieval Model. *Proc. of TREC-3 (Third Text REtrieval Conference).* Gaithersburg, Maryland, USA. 1994

[Cheeseman et al. 88] Cheeseman, P. ; Kelly, J. ; Self, M. ; Stutz, J. ; Taylor, W. ; Freeman, D. Autoclass: a Bayesian Classification system. In Proceedings of *Fifth International Conference on Machine Learning*. 1988.

[Cheeseman & Stutz 96] P. Cheeseman and John Stutz. Bayesian classification (AutoClass): theory and results. In: U.M. Fayyad et al. (Eds.) *Advances in Knowledge Discovery and Data Mining,* 153-180. AAAI, 1996.

[Feldman & Dagan 95] Feldman, R.; Dagan, I. Knowledge Discovery in Textual Databases (KDT). In *Proc. of the 1st Int. Conf. on Knowledge Discovery & Data Mining (KDD-95),* 112-117. AAAI Press, 1995.

[Feldman & Hirsh 96] Feldman ,R. ; Hirsh , H. Mining Associations in Text in the Presence of Background Knowledge. In *Proc. of the 2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD-96),* 343-346. AAAI Press, 1996.

[Joachims 96] Joachims, T. A Probabilistic Analysis of Rocchio Algorithm with TFIDF for Text Categorization. *Technical Report CMU-CS-96-118*. Departament of Computer Science, Carnegie Melow University. 1996.

[Kroon et al. 96] Kroon, H.C.M. ; Mitchell, T.M. ; Kerckhoffs, E. J. H. Improving Learning Accuracy in Information Filtering. In Proceedings of *International Conference on Machine Learning - Workshop on Machine Learning Meets HCI (ICML-96).* Bari. Italy. 1996.

[Landau et al. 98] Landau, D ; Feldman, R. ; Aumann,Y. ; Fresko, M. ; Lindell, Y. ; Lipshtat, O. ; Zamir, O. TextVis: An Integrated Visual Environment for Text Mining. In *Proc. of the 2nd European Symp. on Principles of Data Mining and Knowledge Discovery (PKDD '98), Lecture Notes in Artif. Intell. 1510,* pp. 56-64. Springer-Verlag, 1998.

[Mani et al. 98] Mani, I.; House, D.; Klein, G.; Hirschman, L.; Obrsl, L.; Firmin, T.; Chrzanowski, M.; Sundheim, B. *The TIPSTER SUMMAC Text Summarization Evaluation.* MITRE Technical Report MTR 98W0000138. The MITRE Corporation, Oct. 1998.

[Merkl 97] Merkl, D. Exploration of Document Collections with Self-Organizing Maps: A Novel Approach to Similarity Visualization. In *Proc. of the 1st European Symp. on Principles of Data Mining and Knowledge Discovery (PKDD'97)., Lecture Notes in Artif. Intell. 1263,,* pp. 101-111. Springer-Verlag, 1997.

[Porter 80] Porter, M.F. An algorithm for suffix stripping. *Program 14*, 130-137. 1980. Reprinted in: Sparck Jones, K. and Willet, P. (Eds.) *Readings in Information Retrieval*, 313-316. Morgan Kaufmann, 1997.

[Salton & Buckley 88] Salton, G. and Buckley, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management 24,* 513-523. 1988. Reprinted in: Sparck Jones, K. and Willet, P. (Eds.) *Readings in Information Retrieval*, 323-328. Morgan Kaufmann, 1997.

[Salton & Allan 94] Salton, G. ; Allan, J. Text Retrieval Using the Vector Processing Model. In Proceeding of *Third Symposium on Document Analysis and Information Retrieval*. University of Nevada Las Vegas. 1994.

[Witten et al. 94] Witten, Ian H.; Moffat, Alistair; Bell, Timothy C. Managing Gigabytes. *Van Nostrand Reinhold*. New York. 1994.