Inf Retrieval (2007) 10:69–83
DOI 10.1007/s10791-006-9012-6

Evolutionary learning of document categories

J. I. Serrano · M. D. del Castillo

Received: 3 December 2004 / Accepted: 17 July 2006 / Published online: 22 August 2006

Abstract This paper deals with a supervised learning method devoted to producing categorization models of text documents. The goal of the method is to use a suitable numerical measurement of example similarity to find centroids describing different categories of examples. The centroids are not abstract or statistical models, but rather consist of bits of examples. The centroid-learning method is based on a Genetic Algorithm for Texts (GAT). The categorization system using this genetic algorithm infers a model by applying the genetic algorithm to each set of preclassified documents belonging to a category. The models thus obtained are the category centroids that are used to predict the category of a test document. The experimental results validate the utility of this approach for classifying incoming documents.

Keywords Genetic algorithms · Text categorization · Distance-based methods

1. Introduction

Due to the growth of electronic information stored in text format by companies via intranet or Internet, organizing electronic information in some fashion has become an essential task. Every day millions of documents have to be managed by hand in order to extract useful information and knowledge. A simple automatic system that classifies documents and gives a brief description of the types or categories of documents it has classified would save human searchers an immense amount of work.

There are well-known methods for automating the building of clusters or models of text documents (Doan et al., 2003; Grobelnik and Mladenic, 1998; Sebastiani, 2002). Most such methods are included in the machine learning paradigm, where the categorization problem is

J. I. Serrano · M. D. del Castillo (✉)
Instituto de Automática Industrial, CSIC. Madrid. Spain
e-mail: lola@iai.csic.es

J. I. Serrano
e-mail: nachosm@iai.csic.es

envisioned as a process of learning supervised by the knowledge of the categories and of the training instances that belong to them (Castillo and Serrano, 2004; Weiss et al., 1999). Manually classified documents are the key resource in such a paradigm, and a general inductive process automatically builds a text classifier for every category.

Symbolic methods and the Rocchio method build declarative representations of categories that are easily interpretable by humans. Symbolic methods such as inductive rule learners (Cohen, 1995) obtain classification models that denote the presence or the absence of keywords for every category. Decision tree text learners (Lewis and Ringuette, 1994) build a model of all the categories formed by words and tests on the weights of the words in documents.

The Rocchio method is used for learning linear classifiers consisting of an explicit centroid of a category. Document are represented as vectors of term weights and the centroid of each category is obtained by averaging the term weights of the training documents in the category (Hull, 1994). Classifying a new document comes down to computing some similarity measurement, usually cosine similarity, between the document and the centroids and assigning the document to the category of the closest centroid. The main disadvantage of this method is that it divides the space of documents linearly, and hence if the documents in the category are scattered, the classifier may misclassify most of them (Sebastiani, 2002).

Other learning methods that do not build an explicit classification model of a category are example-driven methods and probabilistic classifiers. Example-driven methods include the *k*-Nearest Neighbor (k-NN) algorithm (Han Eui-Hong, 2001) and Textual Case-Based Reasoning (TCBR) (Godoy and Amandi, 2000; Lenz et al., 1998). Systems using these methods must compare every new test document to be classified to every known preclassified document (TCBR improves on this stage by indexing the cases), a process that is very costly in terms of both time and storage space. Then, the categories associated with these similar documents are used to provide the category to the new document. In spite of the fact that these methods employ no learning stage, their most important disadvantage is their inefficiency at classification time.

Probabilistic classifiers among which the Naïve Bayes classifier is the best-known example (Mitchell, 1997), examine the training documents to extract a vocabulary of terms and compute the necessary probability estimates of the terms. In the classification of new incoming documents, the algorithm uses those estimates to calculate the most probable target category of documents.

There are many other learning methods used for text categorization. Among them, the most noteworthy are regression methods (Lewis and Gale, 1994), support vector machines (Joachims, 1998; Dumais et al., 1998), and neural networks (Ruiz and Srinivasan, 1997). Several experiments have shown these methods to be quite effective (Sebastiani, 2002) but they do not learn classifiers readily understandable by humans

This paper deals with text supervised learning, where text documents and their categories are the only information available. The goal of the developed method is to find the centroids, or prototypical documents, that characterize the different given document categories. The centroids are not abstract models, since they do not contain procedural information like decision rules (Cohen and Singer, 1999), or statistical models, since they are not made up of the words with the best values of statistical measurements such as occurrence frequency (Lewis, 1998). Each centroid of a category can be easily interpreted by humans since it consists of the set of words selected from the training documents in the category. This set of words is found by applying a genetic algorithm for texts, the GAT. Human experts can modify the content of centroids by incorporating or removing words from them based on their experience about the categories. Most current supervised learning systems need exhaustive

training sets involving high download and storage costs before obtaining an effective classifier (Sebastiani, 2002). Because of the difficulty of finding good training samples, an important advantage of the centroid-learning system herein proposed is that a small number of training examples might be used to build effective categorization models.

The following section describes the categorization system based on learning the centroids of several categories. Section 3 describes the preprocessing of documents. The details of the newly developed genetic algorithm and the proposed similarity function are discussed in Section 4. Sections 5 and 6 describe the generalization of the genetic algorithm for texts and the classification application, respectively. Section 7 reviews the experimental settings and results and discusses the comparison between the GAT method and both the k-NN and Naïve Bayes classifiers. Last sections contain the conclusions and pose some issues for future work.

2. Text categorization system

A category centroid is the document of a category that is the most similar document to all the documents in its category and therefore is the most distant document from the documents belonging to other categories. So, although a system using this method requires a stage devoted to learning centroids, at any subsequent time when a test document is entered, the system has only to compare the document to the centroids (which are equal in number to the categories) instead of having to compare the test document to every preclassified document. The proposed centroid-learning stage is based on a genetic algorithm.

Every centroid consists of the set of words selected from the category documents that, when used for document categorization, yields the highest effectiveness. Since there are many specific thematic domains in which it is very difficult to obtain significant samples of training documents, this categorization system might use a small number of training examples and even so obtain very good performance results.

Therefore, the goals of the system are to achieve a high classification effectiveness by making the right classification decisions in any text domain regardless of the domain characteristics, a high training efficiency by building centroids from a small training sample, and a high classification efficiency by comparing a new test document only to the centroid learned for each category.

The centroids learned can be used later for organizing tasks like classification (Sebastiani, 2002) and summarization (Zechner, 1997). This system focuses on the task of classifying incoming documents.

3. Preprocessing

The information contained in text documents is often expressed in a natural language that must be mapped onto a representation that the learner algorithm can understand. The preprocessing step the GAT method uses is comprised within the bag of words approach commonly used in most text applications. The task here is to scan the text of the documents in every category and to turn that text into a list of words, together with the number of times every word occurs in a document. Next, words belonging to a stop list, or words without semantic content, are removed, and several stemming procedures are applied (Porter, 1980). After that, words occurring below a certain threshold and therefore adding no useful information are also removed. A preprocessed document is, therefore, a list of pairs, each pair consisting of a word and its number of occurrences.

4. Learning method

4.1. Genetic algorithms

Genetic algorithms are an optimization technique that simulates the natural evolution process (Goldberg, 1989). Beginning with an initial population of individuals or chromosomes representing tentative solutions to a problem, a new generation is created by combining or modifying the best individuals of the previous generation through genetic operators. The process ends when the best solution is achieved, i.e., when the maximum fitness value of the current individuals of a generation passes a certain threshold, or after a fixed number of generations.

The problem proposed here is to obtain a centroid document representing the documents in a category or class. The central notion is a measure of similarity among documents, so documents in a class show a high intra-class similarity and a low inter-class similarity.

One possible solution to this problem could be generated by taking a random set of words from the documents in a class and measuring the similarity between the random set and every document. Due to the huge search space and the lack of good heuristics based on the semantics of the words, there could be many potential initial sets of words. These considerations make genetic algorithms especially suitable for finding the centroid of a class.

Thus, every document in a category can be seen as a centroid for that category. The GAT starts from a population of tentative centroids and allow these centroids to evolve without using auxiliary knowledge.

4.2. Genetic algorithm for texts (GAT)

Before a genetic algorithm can be applied to find text centroids, the problem the algorithm must solve has to be identified. Since every document is the most similar to itself and is one possible centroid with regard to the other documents in its category, the initial population for each category consists of the preprocessed documents in each category.

Next, it is necessary to establish the representation of chromosomes, the definition of genetic operators fitted to chromosome representation and document domain, and the definition of the fitness function used to determine the best chromosomes of a population.

After applying the GAT, the centroid for every category is a document composed of different portions of every training document belonging to the category.

Chromosome representation. Every chromosome represents a document in a category and symbolizes a tentative centroid for the category. The chromosomes of the initial population are the documents obtained after preprocessing. The genes of a chromosome are pairs consisting of a word and its frequency of occurrence in the document. Since the size of documents is variable, the length of chromosomes is also variable.

Genetic operators. The genetic algorithm for texts uses three operators: elitist, crossover and mutation.

- *Elitist operator.* This operator selects some of the best chromosomes of a population and duplicates them in the next generation. Since the evolution of a population over time can produce worse chromosomes than the original set, this operator provides a mechanism for remembering chromosomes that were previously useful.
- *Two-point crossover operator.* Typically, a simple crossover operator generates a new offspring from two selected parent chromosomes by swapping all the genes between a

Parent Chromosome I:	11112012011	(11 words)
Parent Chromosome II:	111012210	(9 words)
Randomly selected portion for I:		Randomly selected portion for II:
- cross point I.1: 3		- cross point II.1: 4
- cross point I.2: 5		- cross point II.2: 8
Parent Chromosome I:	111**12**012011	(Exchanging genes in boldface)
Parent Chromosome II:	1110**1221**0	
Crossover:		
Offspring I:	111**1221**012011	(13 words)
Offspring II:	1110**12**0	(7 words)

Fig. 1 Example of two-point crossover operator

randomly selected position and the chromosome length less one. The version used by the GAT randomly selects two positions in each parent chromosome. There are two reasons for using a random multiple-point crossover operator. The first crossover point must be selected in each parent chromosome due to the different length of chromosomes. The second random crossover point in each parent allows the number of exchanging genes to be smaller than the number of exchanging genes with a simple crossover operator. The resulting offspring may therefore turn out to be modified to a lesser degree. Figure 1 shows an example of the two-point crossover operation. For the sake of simplicity, chromosomes are represented by strings where the numbers 0, 1, 2, and so on represent different words.

- *Mutation operator.* This operator selects one gene of a chromosome and modifies its value. In the proposed algorithm, there are two ways of modifying a gene. One way lies in replacing the selected word of the gene by another that is not contained in the chromosome and that is randomly selected from the full current set of words present in the documents in a category. This option enables all the words to contribute fairly. The other way to mutate a gene lies in increasing or decreasing the value of the number of occurrences of the gene. The change in the word occurrence is limited by a threshold value in order to avoid significantly modifying the chromosome structure. This latter kind of mutation is justified by the design of the fitness function, as discussed below.

Fitness function. The objective of the fitness function is to compute some measurement of the profit or goodness a chromosome would have as a centroid document of a category. According to the assumptions, every chromosome of a concrete population is a tentative centroid. The closer to every preprocessed training document in a category the centroid is, the better it will be. Obviously, the chromosome taking the highest fitness value will be the best centroid. The main point of the fitness function is to find the measurement of similarity or, inversely, the measurement of distance among documents. The more similar a document is to another, the less distance will exist between them.

There are many studies about how to characterize the similarity between any two texts; some are statistics-based, and others are based on word semantics (Ritcher, 1995). In this paper, similarity is calculated by a statistical function that takes into consideration the number of times words occur within the compared texts. Equation (1) reflects the similarity function.

$$\text{Similarity}\,(X, Y) = \Sigma n x_i{}^* n y_j (\forall i, j / x_i \in X \,\&\, y_j \in Y \,\&\, xi = y_j) \quad (1)$$

where nx_i is the number of occurrences of word x_i in document X, ny_j is the number of occurrences of word y_j in document Y, and x_i and y_j are the same word.

This function calculates the similarity between document X and document Y. The degree of similarity between two documents is obtained by multiplying the number of occurrences of the words that are common to both documents. Thus, if a centroid contains many relevant words that are present in many documents, the centroid will have a high average similarity value with every document and therefore a low average distance value. Since this similarity function takes into account the number of identical words present in any two documents instead of the frequency of occurrence of these words, the similarity function is independent of the length of the documents.

The fitness function value of a chromosome in a certain population is the average similarity between that chromosome and all the training documents in its category plus the average distance between that chromosome and all the training documents in the remaining categories (see Eq. (2)).

$$\text{Fitness}(Chr) = \frac{\Sigma_i^{\text{N}}\text{Similarity}(i, Chr)}{N} + \frac{1}{\frac{\Sigma_j^{\text{M}}\text{Similarity}(j, Chr)}{M}} \tag{2}$$

where Chr is the chromosome evaluated, i is a training document in the target category, j is a training document in the remaining categories, N is the number of documents in the target category, and M is the number of documents in the remaining categories.

Figure 2 shows an example of computation of the values taken by the Similarity and Fitness functions for a population consisting of three documents of category A. The training sample is comprised of four documents in category A and two documents in other categories.

Preprocessed training documents:

	String	*Pairs (word, occurrence)*	
Training Doc. 1	100110011211	(0, 4) (1, 7) (2, 1)	category A
Training Doc. 2	11111201201	(0, 2) (1, 7) (2, 2)	category A
Training Doc. 3	110010112201	(0, 4) (1, 6) (2, 2)	category A
Training Doc. 4	00110001000	(0, 8) (1, 3)	other category
Training Doc. 5	101010101000	(0, 7) (1, 5)	other category

Population of chromosomes in category A:

	String	*Pairs (word, occurrence)*
Chromosome I	1000101001	(0, 6) (1, 4)
Chromosome II	11112012011	(0, 2) (1, 7) (2, 2)
Chromosome III	111012210	(0, 2) (1, 5) (2, 2)

Similarity	**1**	**2**	**3**	**4**	**5**
I	52	40	48	60	62
II	59	57	54	37	49
III	45	43	42	31	39

Fitness (I) = 46.018 **Fitness** (II) = 58.201 **Fitness** (III) = 44.019

Fig. 2 Similarity and fitness values of a population of chromosomes

Table 1 Values of the parameters used by the GAT

Genetic operators	Workspace (% of best individuals)	Application probability
Elitist	30	0.4
Two-point crossover	65	0.65
Mutation	80	0.8

Other algorithm parameters. The application of the GAT requires that certain other parameters must be determined as well, such as the maximum number of generations, the stop fitness value, the probability of application of operators, and the operator workspace. Table 1 presents the values for some parameters used with the different genetic operators. The workspace is the subset of the population comprised of the best individuals on which all the operators can work. The elitist operator works on those individuals that comprise the best 30% of a population. The actual number of chromosomes involved in the elitist operation is determined by the elitist probability. The two-point crossover operator works on the best 65% of the population that has not selected by the elitist operation. The mutation operator can modify the best 80% of the individuals that have not been selected for copying or crossing. The actual proportion of chromosomes in each workspace involved in crossover and mutation operations is determined by the application probabilities of each operator.

The values of these parameters in the GAT have been set empirically depending on several factors, like the document domain, the number of training documents, and the distribution of words in the documents. The GAT searches from an initial population consisting of the preprocessed documents of the categories under study and stops after running an upper limit on the number of generations. Some proofs concerning the maximum number of generations are given in the section discussing the experiments.

5. Generalization of the GAT

The application of the GAT to text web pages or hypertext implies taking into account certain additional issues concerning the presence of often ungrammatical text in web pages. First, in the preprocessing step, the number of word occurrences is increased for certain word formats. If a word is in the title of the page or in boldface, its number of occurrences is increased to highlight its interest (for example, the frequency is increased by adding ten units for a word found in the title, nine units for a word in boldface, and so on). If a word is tagged within an *html* header tag $< \mathrm{H}n >$, its number of occurrences is increased by adding 6-n. These settings can be determined empirically.

Four different types of information can be explicitly found in web pages: *url*, meta-keywords, hyperlinks and plain text. Words can assume different semantic power depending on their placement. Based on this division of documents, the preprocessing step will generate four lists of word/occurrences pairs, one each for the four types of information.

The GAT can consider a web page as a whole text or as four parts of text. When the web page is to be processed as a whole, the GAT will apply the crossover operator only to a part chosen at random for every two parent chromosomes selected to be crossed. If the web page is considered as consisting of four parts, then the chromosomes handled by the GAT are divided into four parts, and the genetic operators are applied to the four parts in a

parallel manner. Therefore, the fittest chromosomes (and the text categories) can be modeled by as many centroids as different types of information exist in the web pages belonging to the categories.

A general GAT only needs to receive a number between 1 and 4, indicating the presence of one or more types of information, in order to apply the crossover operator to the chromosomes correctly. Thus, the application of the GAT can be generalized to include grammatical texts and hypertexts, because any kind of document can be mapped onto the described web representation; and therefore use can be made of the information that web page authors give when they place a word in some special position and/or format.

6. Application to classification

Text categorization can be applied in any context requiring document organization or selective and adaptive document dispatching. Assigning categories to documents is essential to the efficient management and retrieval of information and knowledge. The application of the GAT-based system focuses on the task of classifying incoming documents in several non-disjoint categories.

The classification process begins when the system receives a test document. First, the similarity between that document and every learned centroid according to Eq. (1) is calculated. Next, the document is classified as belonging to the category or categories whose centroid or centroids are closest to the document. In each category a threshold value is set for similarity so that any document whose similarity fails to reach the threshold is not classified into the category. One interesting advantage of the similarity measurement is that the values it takes for a document with respect to each centroid can be seen as degrees of membership in each category.

When the categorization system classifies web pages, it takes into account the possible existence of centroids composed of four subcentroids, one centroid for each type of information. Therefore, when a new web page is being classified, the similarity between each type of information on the page and the corresponding subcentroid is calculated. In this case, at most four similarity measurements can be obtained for each category. The final similarity between a web page and a category is given by the average of these values.

7. Experiments

The system described above has been evaluated using two text collections. The first experiment found the number of generations and the number of centroids per category that yield the best classification results. Once these parameters were determined, a second experiment was set up to evaluate two issues involved in the genetic centroid-based approach: (1) differences in classification performance, checked by considering web documents as a whole or as four separate types of information (url, meta, text, links); (2) the results obtained with a two-point crossover operator instead of a simple crossover operator. For each experiment, the GAT started with an initial population of tentative centroids formed by the preprocessed documents obtained from each collection on which the experiment was performed.

7.1. Parameter setting

The first experiment was performed on the Reuters-21578 collection. This experiment was set up to compare the performance of the GAT using different values for the number of generations and the number of centroids. The sample taken into account is comprised of the documents of eight categories with more than 100 documents (Acq, Coffee, Earn, Gold, Nat-Gas, Money, Sugar, Trade). The 1,987 resulting documents, each one labeled with its correct category, were randomly divided into two groups, a training set of 240 examples (30 documents per category) and a test set of 1,747 examples. The GAT was applied to the training set, and only the best centroid for each category was obtained. Next, the similarity between each test document and the eight learned centroids according to Eq. (1) was calculated. Finally, the GAT-based categorization system classified every test document as belonging to the category whose centroid was closest to the document. The comparison between the correct categories of all test documents, as defined in the Reuters collection, and the predicted categories for them by the categorization system allows to compute the performance of the system.

Table 2 shows the classification performance for each category in the rows and the results for different values of the maximum number of generations in the columns. Classification performance is based on calculating three different measurements: precision (*Pr*) or percentage of predicted documents for a category correctly classified, recall (*Rc*) or percentage of documents for a category correctly classified, and F-measure (*F*) which is a combination of the precision and recall measurements, *F-measure* = (*2*precision*recall)/(precision* + *recall*). Table 2 also indicates that the greater the number of generations, the better the results. Although the number of generations that yields the best results depends on the categories, it seems that the best macroaveraged value of the maximum number of generations is 100.

The second part of this experiment was designed to study the possibility of improving the GAT classification performance by selecting more than one centroid for each category from the last generation. The best final centroids for each category have many words in common with one another and differ by only a few words. Therefore, selecting the centroids after the very best centroid is equivalent to taking words that are in the selected centroids but not in the very best centroid and adding them to the very best centroid. Table 3 shows the precision, recall and F-measure classification results for each category (rows) and for different numbers of centroids used in the classification process (columns).

Table 2 Average results from five runs of the GAT on Reuters test set for classification with different maximum numbers of generations

	20 Generations			50 Generations			100 Generations			175 Generations		
	Pr	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*
ACQ	69.91	64.6	67.15	93.88	61.4	74.24	84.68	**79.6**	82.06	**99.5**	70	**82.15**
COF	49.18	93.75	64.51	**56.86**	90.62	**69.87**	50.81	**96.87**	66.66	37.03	93.75	53.09
EAR	96.33	72.37	82.65	97.82	78.62	87.17	**99.85**	**87.87**	**93.48**	98.40	87.5	92.59
GOL	90.9	80	85.1	**99.99**	**88**	**93.61**	95.65	**88**	91.66	79.16	76	77.55
NAT	88.23	44.11	58.82	78.26	**52.94**	63.15	74.99	44.11	55.55	**99.99**	**52.94**	**69.23**
MON	**99.99**	**77.77**	**87.49**	80.64	55.55	65.78	**99.99**	71.11	83.11	**99.99**	**77.77**	**87.49**
SUG	**99.99**	70.73	82.85	**99.99**	73.17	84.5	**99.99**	**75.6**	**86.11**	**99.99**	73.17	84.5
TRA	**99.99**	49.6	66.31	**99.99**	57.6	73.09	**99.99**	**64**	**78.04**	98.79	59.2	74
Avg.	86.81	69.11	76.95	96.93	59.5	73.73	92.33	**71.8**	**80.78**	**99.14**	64.6	78.22

Table 3 Average results from five runs of the GAT on Reuters test set for classification with different numbers of centroids

	1 Centroid			2 Centroids			4 Centroids			8 Centroids			16 Centroids		
	Pr	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*
ACQ	77.9	62.8	**69.5**4	69.91	**64.6**	67.15	62.64	55	58.57	70.7	50.2	58.71	**82.5**2	34	48.15
COF	21.2	87.5	34.14	**49.1**8	**93.7**5	**64.5**1	37.03	**93.7**5	53.09	30	**93.7**5	45.45	44.99	84.37	58.69
EAR	97.4	61.5	75.40	96.33	**72.3**7	**82.6**5	96.89	62.5	75.98	97.59	65.87	78.65	**98.0**8	70.37	81.95
GOL	**99.9**	72	83.72	90.9	**80**	85.1	95.23	**80**	**86.9**5	84.99	68	75.55	90.47	76	82.6
NAT	86.3	**55.8**8	**67.8**5	88.23	44.11	58.82	74.99	52.94	62.06	71.42	44.11	54.54	**90.9**	29.41	44.44
MON	97.2	**80**	**87.8**0	**99.9**9	77.77	87.49	**99.9**9	73.33	84.61	90.9	66.66	76.92	88.57	68.88	77.49
SUG	96.9	**78.0**4	**86.4**8	**99.9**9	70.73	82.85	**99.9**9	70.73	82.85	**99.9**9	70.73	82.85	**99.9**9	68.29	81.15
TRA	**99.9**	**68**	**80.9**5	**99.9**9	49.6	66.31	**99.9**9	54.4	70.46	**99.9**9	56	71.79	**99.9**9	44	61.11
Avg.	84.6	**70.7**1	**77.0**5	86.81	69.11	76.95	83.34	67.83	74.78	80.69	64.41	71.63	**86.9**3	59.4	70.58

The table shows that the more centroids there are, the worse the results. Using more than two centroids is not a good option because, although the precision value is maintained, the remaining performance values fall.

7.2. Evaluation of the GAT

The second experiment was performed using a collection of web pages called BankSearch. This data set is jointly provided by BankSearch Information Consultancy Ltd. and the Computer Science Department at the University of Reading. The collection consists of 10,000 web documents classified into ten categories of equal size, each containing 1,000 web pages (Sinka and Corne, 2002). A subset of this collection, consisting of 4,000 examples fairly arranged into five categories, was selected. All the categories were divided into three disjoint sets: one training set to learn the category centroids, consisting of only 50 examples per category, and two test sets to validate them, as shown in Table 4. In this experimental setting, the GAT was run for 100 generations and only one centroid per category was selected from the last generation, in the light of the results of the first experiment.

The first part of this experiment was intended to check the GAT performance with two different configuration conditions: web pages as a whole text or as four separate types of information (url, meta, text, links). Table 5 shows the values of the effectiveness measurements with the two different GAT configurations, both of them with a two-point crossover operator.

Since the best performance values are achieved by dividing documents into four types of information, the second part of the experiment, shown in Table 6, was designed to confirm that the two-point crossover operator behaves better than the simple crossover operator for documents divided into four types of information.

Table 4 Distribution of the examples of the BankSeach dataset into training and Test Sets

Categories number of examples	Training set	Test set I	Test set II
Commercial banks	50	250	500
Java	50	250	500
Astronomy	50	250	500
Soccer	50	250	500
Sport	50	250	500
Total	250	1,250	2,500

Table 5 Average results from five runs of the GAT on Test Set I

Categories Test Set I	Full document two-point crossover			Four information types two-point crossover		
	Pr	*Rc*	*F*	*Pr*	*Rc*	*F*
Commercial B.	**95.45**	8.4	15.441	90.53	**95.6**	**92.996**
Java	**80.83**	54	64.748	77.18	**98.8**	**86.666**
Astronomy	98.03	40	56.818	**99.41**	**68.4**	**81.042**
Soccer	53.57	**98.8**	69.47	**91.86**	90.4	**91.129**
Sport	**100**	5.6	10.606	**100**	**72**	**83.720**
Average	85.57	41.36	55.765	**91.79**	**85.04**	**88.286**

Table 6 Average results from five runs of the GAT on Test Set I

Categories Test Set I	Four information types simple crossover			Four information types two-point crossover		
	Pr	*Rc*	*F*	*Pr*	*Rc*	*F*
Commercial B.	89.28	60	71.770	90.53	**95.6**	**92.996**
Java	79.23	**99.2**	**88.099**	77.18	**98.8**	**86.666**
Astronomy	93.13	**76**	**83.700**	**99.41**	**68.4**	**81.042**
Soccer	89.64	90	89.820	**91.86**	90.4	**91.129**
Sport	**100**	67.6	80.668	**100**	**72**	**83.720**
Average	90.26	78.56	84.004	**91.79**	**85.04**	**88.286**

In both tables, the rows show the classification performance associated with each category. The last row presents the macroaveraged values of each performance measurement. The maximum value of each measurement for every category is printed in boldface type. All the numerical values given in the tables are the average result of five runs of the genetic algorithm.

As the results show in Table 5, when the GAT is configured to consider four different kinds of information in web documents, it gives a better average performance than when it processes each document as a full text. The two-point crossover operator yields a better performance than the simple crossover. The best configuration for the algorithm therefore seems to be the configuration that considers the types of information in each document separately and employs the two-point crossover operator.

This configuration was used to calculate classification performance in Test Set II, the test set with the largest number of web documents. Table 7 shows that the system performed very

Table 7 Average results from five runs of the GAT on Test Set II

Categories Test Set II	Four information types two-point crossover		
	Pr	*Rc*	*F*
Commercial	71.69	98.8	83.095
Java	78.55	76.2	77.360
Astronomy	96.15	75	84.269
Soccer	89.76	91.2	90.476
Sport	100	71.6	83.449
Average	87.23	82.56	84.830

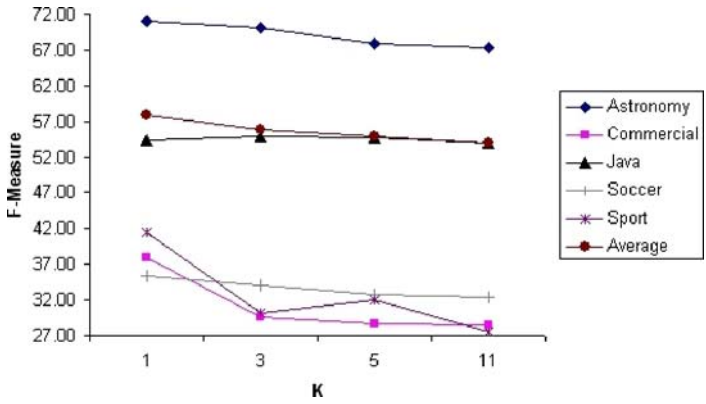

Fig. 3 F-measure values on Test Set II using the k-NN classifier for different values of k

well in some categories, and, on average, the values of the performance measurements are quite high in both two test sets with only 50 training examples per category.

7.3. The GAT versus the Naïve Bayes and k-NN classifiers

For cross-classifier comparison, the k-NN and Naïve Bayes classifiers have also been run on the BankSearch collection. The comparison took into account measurements of classification effectiveness (Pr, Rc, and F) and measurements of training and classification efficiency.

In order to classify a test document in a certain category, k-NN looks for the category of the k training documents most similar to this document. Then, the algorithm proposes to assign the category associated with the majority of the k documents to the incoming document.

Figure 3 shows the F-measure values of the k-NN classifier computed on Test Set II for all the categories of the collection and the average value with different values of k. The graphic indicates that the best F-measure value is obtained with $k = 1$ for all the categories. Moreover, the higher the value of k is, the worse is the resulting F-measure for all the categories.

Table 8 shows a comparison of the performance results for the GAT method, the Naïve Bayes classifier, and the k-NN classifier using the BankSearch collection. The training set was comprised of 50 examples, and the test set, of 250 examples (Training Set and Test Set I in Table 4, respectively). The classification performance of the k-NN classifier was calculated with $k = 1$.

Table 8 Performance results of the GAT, the Naïve Bayes classifier and the k-NN classifier

Categories	GAT			Naïve Bayes			k-NN		
Test Set II	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*	*Pr*	*Rc*	*F*
Commercial	71.69	**98.8**	83.09	**87.4**	98	**92.39**	86.29	60.40	71.06
Java	78.55	76.2	77.36	95.6	**77.2**	**85.02**	**96.72**	23.60	37.94
Astronomy	96.15	75	**84.27**	**97.7**	71.4	83.21	39.01	**89.80**	54.39
Soccer	89.76	**91.2**	**90.48**	75.44	84.8	79.84	**93.97**	21.80	35.39
Sport	**100**	**71.6**	**83.45**	69.52	84.4	76.24	**100**	26.20	41.52
Average	**87.23**	82.56	**84.83**	85.13	**83.16**	84.13	83.20	44.36	57.87

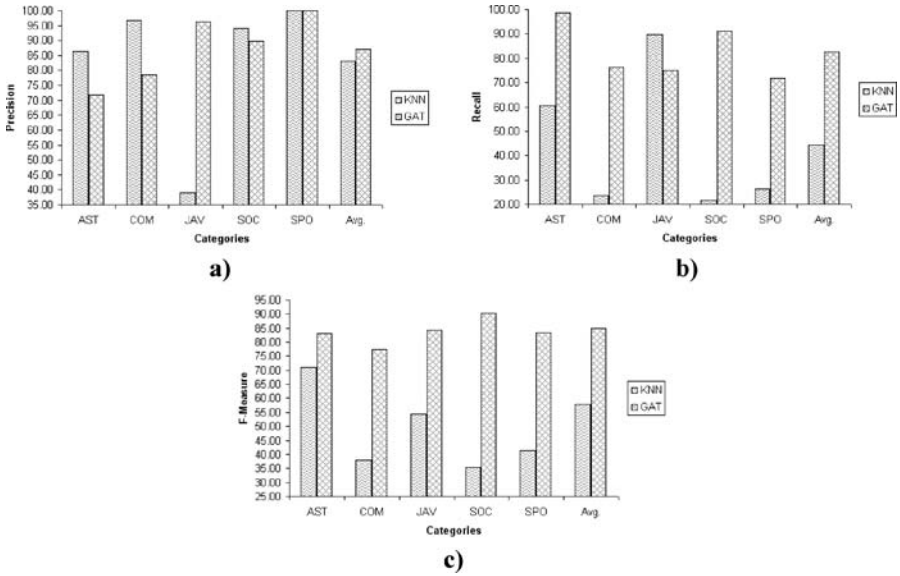

Fig. 4 (a) Precision, (b) Recall, and (c) F-measure values on Test Set II for the k-NN classifier, the Naïve Bayes classifier and the GAT

Due to the high dimensionality of the word space, a dimensionality reduction technique was used to select 20% of the best ranked words before applying the Naïve Bayes classifier. Words were scored by computing the gain information statistical measurement (Yang et al., 1997).

The bar charts in Fig. 4 show the same comparative performance results among the different classifiers for each performance measure. The results obtained highlight the successful classification performance of the genetic-based model when working with a small number of training documents. Although the Naïve Bayes classifier also behaves very well, the GAT method has obvious advantages in terms of interpretability, since the quantitative nature of the probabilistic learner produces results not easily interpretable by humans. The k-NN classifier yields the poorest behavior of the three.

Another interesting comparison of evaluation of these classifiers has been carried out using two measures alternative to effectiveness: learning efficiency (the average time it takes to build a classifier for a category from a certain training set) and classification efficiency (the average time it takes to classify a new document under a certain category). Table 9 shows the CPU time consumed in the training and classification stages of the different classifiers using the BankSearch collection. The CPU time of the GAT in the training stage has been computed for 10 generations. The training stage of the GAT and the Naïve Bayes classifier were performed off-line. The computation times indicate that the GAT has the best classification time, and the k-NN classifier is the slowest.

Table 9 Efficiency results of the GAT, the Naïve Bayes classifier and the k-NN classifier

	GAT	k-NN	Naïve Bayes
Training stage	17’	–	8’
Classification stage	0.37”	21”	11”

8. Conclusions

A genetic algorithm for texts has been proposed for obtaining centroid documents that describe text categories by first learning the centroids and then using them to classify documents. This technique consumes little time in the classification stage. The system requires no computations to find the similarity between new documents and the documents stored in the repository or the case base, but only to find the similarity between new documents and learned centroids. The classification results have shown that the technique works very well using very few training documents and at most two centroids per category. The classification results can even be improved by fine-tuning the algorithm parameters and perhaps by selecting more representative training examples.

In both experiments, a small number of training examples was used and the centroid-based method was proved to have a higher classification efficiency than the other learning approaches that were tested.

The learned centroids are easily interpreted by humans, who can read the words in the centroids and modify the centroids on the basis of their own knowledge about the categories, in order to improve the quality of the description of the categories given by the centroids.

9. Future work

Basically, future work will focus on three points:

Fine-tuning of algorithm parameters to improve classification results.
Research into other fitness functions, maybe partially semantics-based functions.
Use of the centroid for multidocument automatic text summarization, where the objective is to generate a text document that is smaller than the original documents and summarizes the main details of the original texts.

References

del, Castillo M. D., & Serrano, J. I. (2004). A multistrategy approach for digital text categorization from imbalanced documents. *ACM SIGKDD Explorations, 6*, 70–79.

Cohen, W. W., & Singer, Y. (1999). Context-sensitive learning methods for text categorization. *ACM Trans. Inform. Systems, 17*(2), 141–173.

Cohen, W. W. (1995). Learning to classify English text with ILP methods. In L. De Raedt, (Ed.), *Advances logic programming* (pp. 124–143). Amsterdam: IOS Press.

Doan, A., Domingos, P., & Halevy, A. (2003). Learning to match the schemas of data sources: a multistrategy approach. *Machine Learning, 50*, 279–301.

Dumais, S. T., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representation for text categorization. In *Proceedings of the CIKM-98, 7th International Conference on Information and Knowledge Management* (pp. 148–155). Bethesda.

Goldberg, D. (1989). *Genetic algorithms in search, optimization & machine learning*, (ed.) Addison-Wesley Publishing Company, Inc.

Godoy, D., & Amandi, A. (2000). PersonalSearcher: an intelligent agent for searching web pages (pp. 43–52). LNAI, 1952. Springer-Verlag.

Grobelnik, M., & Mladenic, D. (1998). Efficient text categorization. In *text mining workshop on the 10th european conference on machine learning* (pp. 1–10). Chemnitz.

Han Eui-Hong, S., Karypis, G., & Kumar, V. (2001). Text categorization using weight adjusted k-nearest neighbor classification. In *PAKDD'2001* (pp. 53–65). Springer-Verlag, LNAI 2035.

Hull, D. A. (1994). Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval* (pp. 282–289). Dublin.

Joachims, T. (1998). Text categorization with support vector machines. In *Proceedings of ECML-98 10th European Conference on Machine Learning* (pp. 137–142). Chemnitz.

Lenz, M., Hubner, A., & Kunze, M. (1998). Textual CBR. In M. Lenz, B. Bartsch, B. D. Burkhard, and S. Wess (Eds.), *Case-based reasoning technology* (pp. 115–138). Springer-Verlag, LNAI 1400.

Lewis, D. D. (1998). Naïve Bayes at forty: The independence assumption in information retrieval. In *Proceedings of ECML-98, 10th European Conference on Machine Learning* (pp. 4–15). Germany.

Lewis, D. D., & Gale, W. A. (1994). Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of SIGIR-94, 11th International Conference on Research and Development in Information Retrieval* (pp. 3–12). Dublin.

Lewis, D. D., & Ringuette, M. (1994). A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval* (pp. 81–93). Las Vegas.

Mitchell, T. M. (1997) *Machine learning*. The McGraw- Hill Companies.

Porter, M. F. (1980) An algorithm for suffix stripping. *Program, 14*(3), 130–137.

Ritcher, M. M. (1995). The knowledge contained in similarity measures. In *Invited Talk at ICCBR-95*.

Ruiz, M. E., & Srinivasan, P. (1997). Automatic text categorization using neural networks. In *Proceedings of the 8th ASIS/SIGCR Workshop on Classification Research* (pp. 59–72). Washington.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys, 34*(1), 1–47.

Sinka, M. P., & Corne, D. W. (2002). A large benchmark dataset for web document clustering. In A. Abraham, J. Ruiz-del-Solar, and M. Koeppen (eds.), *Soft computing systems: design, management and applications* (pp. 881–890). (Volume 87 of Frontiers in Artificial Intelligence and Applications, 2002).

Weiss, S. M., Apté, Damerau, F. J., Johnson, D. E., Oles, F. J., Goezt, T., & Hampp, T. (1999). Maximizing text-mining performance. *IEEE Intelligent Systems, 14*(4), 63–69.

Yang, Y., & Pedersen, J. P. (1997). A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning* (pp. 412–420). Nashville.

Zechner, K. (1997). A literature survey on text summarization. Paper for Directed Reading (Fall 1996), Carnegie Mellon university. Computational Linguistics.