IMPACT ON PERFORMANCE OF HYPERTEXT CLASSIFICATION OF SELECTIVE RICH HTML CAPTURE

Houda Benbrahim and Max Bramer
Department of Computer Science and Software Engineering, Portsmouth University, UK
{houda.benbrahim, max.bramer}@port.ac.uk

Abstract: Hypertext categorization is the automatic classification of web documents into predefined classes. It poses new challenges for automatic categorization because of the rich information in a hypertext document. Hyperlinks, HTML tags, and metadata all provide rich information for hypertext categorization that is not available in traditional text classification. This paper looks at (i) what representation to use for documents and which extra information hidden in HTML pages to take into consideration to improve the classification task, and (ii) how to deal with the very high number of features of texts. A hypertext dataset and three well-known learning algorithms (Naïve Bayes, K-Nearest Neighbour and C4.5) were used to exploit the enriched text representation along with feature reduction. The results showed that enhancing the basic text content with HTML page keywords, title and anchor links improved the accuracy of the classification algorithms.

Key words: Machine Learning; Hypertext classification.

1. INTRODUCTION

It has been estimated that the World Wide Web comprises more than 3 billion pages and is growing at a rate of 1.5 million pages a day [1]. A recent study [2] showed that users prefer to navigate through directories of pre-classified content, and that providing a categorized view of retrieved documents enables them to find more relevant information in a shorter time. The common use of the manually constructed category hierarchies for navigation support in Yahoo [3] and other major web portals has also

demonstrated the potential value of automating the process of hypertext categorization.

Automated hypertext categorization poses new research challenges because of the rich information in a hypertext document. Hyperlinks, HTML tags, and metadata all provide rich information for classifying hypertext that is not available in traditional text categorization. Researchers have only recently begun to explore the issues of exploiting rich hypertext information for automated categorization.

There is a growing volume of research in the area of learning over web text documents. Since most of the documents considered are in HTML format, researchers have taken advantage of the structure of those pages in the learning process. The systems generated differ in performance because of the quantity and nature of the additional information considered.

Oh et al. [4] reported some observations on a collection of online Korean encyclopedia articles. They used system-predicted categories of the linked neighbors of a test document to reinforce the classification decision on that document and they obtained a 13% improvement over the baseline performance when using local text alone.

Furnkranz [5] used a set of web pages from the WebKB university corpus to study the use of anchor text and the words near the anchor text in a web page to predict the class of the target page pointed to by the links. By representing the target page using the anchor words on all the links that point to it, plus the headlines that structurally precede the sections where links occur, the classification accuracy of a rule-learning system improved by 20%, compared with the baseline performance of the same system when using the local words in the target page instead.

Slattery and Mitchell [6] used the WebKB university corpus, but studied alternative learning paradigms, namely, a First Order Inductive Learner which exploits the relational structure among web pages, and a Hubs and Authorities style algorithm exploiting the hyperlink topology. They found that a combined use of these two algorithms performed better than using each alone.

Yang, Slattery and Ghani [7] have defined five hypertext regularities which may hold in a particular application domain, and whose presence may significantly influence the optimal design of a classifier. The experiments were carried out on 3 datasets and 3 learning algorithms. The results showed that the naïve use of the linked pages can be more harmful than helpful when the neighborhood is noisy, and that the use of metadata when available improves the classification accuracy.

Attardi et al. [8] described an approach that exploits contextual information extracted from an analysis of the HTML structure of Web

documents as well as the topology of the web. The results of the experiments with a categorization prototype tool were quite encouraging.

Chakrabarti et al. [9] studied the use of citations in the classification of IBM patents where the citations between documents were considered as hyperlinks, and the categories were defined on a topical hierarchy. Similar experiments on a small set of pages with real hyperlinks were also conducted. By using the system-predicted category labels for the linked neighbors of a test document to reinforce the category decision on that document, they obtained a 31% error reduction, compared to the baseline performance when using the linked documents, treating the words in the linked documents as if they were local. This approach increased the error rate of their system by 6% over the baseline performance.

This paper deals with web document categorization. Two issues will be considered in depth: (i) the choice of representation for documents and the extra information hidden in HTML pages that should be taken into consideration to improve the classification task, and (ii) how to deal with the very high number of features generated when processing text. Finally, data collected from the web will be used to evaluate the performance of the different classification methods with different choices of text representation and feature selection strategy.

Document representation is described in Section 2. Some classification algorithms used for hypertext are reviewed in Section 3. Section 4 presents experiments and results, comparing different classification algorithms with different webpage representation techniques.

2. TEXT REPRESENTATION

In order to apply machine-learning methods to document categorization, consideration first needs to be given to a representation for HTML pages. A pre-processing stage is used to remove potentially distracting information before a document is presented to a classifier. First, HTML tags, digits and punctuation marks are removed.

Next, an indexing procedure that maps a text into a compact representation is applied to the dataset. The most frequently used method is a *bag-of-words* representation where all words from the set of documents under consideration are taken and no ordering of words or any structure of text is used.

There are two ways in which the words (features) can be chosen in order to classify a set of documents. The words can be selected to support classification under each category in turn, i.e. only those words that appear

in documents in the specified category are used (the *local dictionary* approach). This means that the set of documents has a different feature representation (set of features) for each category. Alternatively, the words can be chosen to support classification under all categories, i.e. all the words that appear in any of the documents are used (*global feature selection*). In this paper, the local dictionary approach is adopted, as it has been reported to lead to better performance [10] and [11].

Assuming there are N features for a particular classification, the documents are represented using a vector space model (VSM) [12]. The i^{th} document O_i is represented as an N-dimensional vector $O_i = (X_{i1}, X_{i2}, X_{i3}, \ldots X_{iN})$, where X_{ij} stands for the *weight* of word t_j, and measures the importance of t_j in the document.

There are a number of ways to compute the weights of words. The most commonly used methods are *binary* (where 1 denotes presence and 0 absence of the term in the document), *term frequency* in the document, and *Term Frequency-Inverse Document Frequency* (TF-IDF). The TF-IDF method has been reported to lead to better performance [13] and has been used in this paper.

The weights X_{ij} are computed using the formula:

$(TF\_IDF)_{ij} = tf_{ij} * idf_i$ where $idf_i * \log_2 (n / df_j)$

tf_{ij} is the number of times term t_j occurs in document O_i

df_j is the number of training documents in which word t_j occurs at least once

n is the total number of training documents.

This weighting function encodes the intuitions that the more often a term occurs in a document, the more it is representative of the document's content, and that the more documents in which a term occurs, the less discriminating it is. In order to make weights fall in the [0,1] interval and for documents to be represented by vectors of equal length, the weights resulting from the function $(TF\_IDF)_{ij}$ are normalized by 'cosine normalization', given by:

$$X_{ij} = \frac{(TF\_IDF)_{ij}}{\sqrt{\sum_{j=1}^{N} (TF\_IDF)_{ij}^2}}, \quad 1 \le i \le n \text{ and } 1 \le j \le N$$

where N is the number of terms that occur at least once in the set of training documents.

With the bag-of-words approach for text representation, it is possible to have tens of thousands of different words occurring in a fairly small set of documents. Using all these words is time consuming and represents a serious

obstacle for a learning algorithm. Moreover many of them are not really important for the learning task and their usage can degrade the system's performance. Many approaches exist to reduce the feature space dimension, the most common ones are: (i) the use of a stop list containing common English words, (ii) or the use of stemming, that is keeping the morphological root of words, or (iii) the use of feature selection algorithms such as information gain.

In the experiments reported here, stop words are removed and IG (information gain) criterion is used as it has been reported to outperform many of the common existing feature selection methods [14]. The IG criterion measures the number of bits of information gained for category prediction by knowing the presence or absence of a feature in a document. The IG of a term t and for a class C_k ($1 \le k \le c$, where c is the number of target classes) is defined as:

$$IG(t,C_k) = -P(C_k)\log_2 P(C_k) + P(t)P(C_k/t)\log_2 P(C_k/t) + P(\bar{t})P(C_k/\bar{t})\log_2 P(C_k/\bar{t})$$

$P(C_k)(or P(\bar{C}_k))$ is the probability of having class C_k (or not having C_k).
$P(t)(or P(\bar{t}))$ is the probability of having term t (or not having term t).
$P(C_k/t)$ is the probability of having class C_k given that term t is observed in the document.
$P(C_k/\bar{t})$ is the probability of having class C_k given that term t is not in the document

These probabilities are estimated by counting occurrences in the training set of documents. The IG function captures the intuition according to which the most valuable terms for categorization under C_k are those that are distributed most differently in the sets of positive and negative examples of the category.

In this work a procedure according to which all terms are ranked based on their IG(t, C_k) is used for feature selection. For every class C_k, only the features (from the training documents belonging to C_k) with the highest (say) 20, 50 or 100 values of IG(t, C_k) are selected. The other features are ignored.

3. CLASSIFICATION ALGORITHMS

3.1 Naïve Bayes (NB)

Naïve Bayes (NB) is a widely used model in machine learning and text classification. The basic idea is to use the joint probabilities of words and categories in the training set of documents to estimate the probabilities of categories for an unseen document. The term 'naïve' refers to the assumption that the conditional probability of a word is independent of the conditional probabilities of other words in the same category.

A document is modeled as a set of words from the same vocabulary, *V*. For each class, C_j, and word, $w_k \in V$, the probabilities, $P(C_j)$ and $P(w_k|C_j)$ are estimated from the training data. Then the posterior probability of each class given a document, D, is computed using Bayes' rule:

$$P(C_j \mid D) = \frac{P(C_j)}{P(D)} \prod_{i=1}^{|D|} P(a_i \mid C_j)$$

where a_i is the i^{th} word in the document, and $|D|$ is the length of the document in words. Since for any given document, the prior probability $P(D)$ is a constant, this factor can be ignored if all that is desired is ranking rather than a probability estimate. A ranking is produced by sorting documents by their odds ratios, $P(C_1|D) / P(C_0|D)$, where C_1 represents the positive class and C_0 represents the negative class. An example is classified as positive if the odds are greater than 1, and negative otherwise.

3.2 K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) is a well-known statistical approach in pattern recognition. KNN assumes that similar documents are likely to have the same class label. Given a test document, the method finds the *K* nearest neighbors among the training documents, and uses the categories of the *K* neighbors to weight the category candidates. The similarity score of each neighbor document to the test document is used as the weight of the categories of the neighbor document. If several of the K nearest neighbors share a category, then the per-neighbor weights of that category are added together, and the resulting weighted sum is used as the likelihood score of that category with respect to the test document. By sorting the scores of candidate categories, a ranked list is obtained for the test document. By thresholding on these scores, binary category assignments are obtained. The decision rule in KNN can be written as:

$$y(\vec{x}, c_j) = \sum_{\vec{d_i} \in KNN} sim(\vec{x}, \vec{d_i}) y(\vec{d_i}, c_j) - b_j$$

where $y(\vec{d_i}, c_j) \in \{0,1\}$ is the classification for document $\vec{d_i}$ with respect to category cj (y = 1 for Yes, and y = 0 for No); $sim(x, \vec{d_i})$ is the similarity between the test document x and the training document $\vec{d_i}$; and b_j is the category specific threshold for the binary decisions.

3.3 Decision Tree Classification (C4.5)

C4.5 is a decision tree classifier developed by Quinlan. The training algorithm builds a decision tree by recursively splitting the data set using a test of maximum gain ratio. The tree is then pruned based on an estimate of error on unseen cases. During classification, a test vector starts at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. This process is then repeated for the subtree rooted at the new node until a leaf is encountered, at which time the pattern is asserted to belong to the class named by that leaf.

4. EXPERIMENTS

4.1 Dataset

To test the proposed algorithms for hypertext classification, datasets were needed that reflected the properties of real world hypertext classification tasks.

The major practical problem in using web document datasets is that most of the URLs become unavailable. The well-known dataset WebKB project at CMU [15] is outdated since most of its web pages are no longer available.

The data used for the experiments comprises a set of HTML web documents. The dataset was provided by Reading University, UK [16]. The Open Directory Project and Yahoo! categories were used to provide web pages that have already been categorized by people. The considered dataset consists of 11,000 pages. The web pages were distributed over 11 different categories under 4 distinct themes. The dataset consists of some sets of categories that are quite distinct from each other, as well as other categories that are quite similar to each other. Table 1 gives a summary of the dataset.

Table 1. Dataset Summary

Category' ID	Category' Name	Associated Theme

Category' ID	Category' Name	Associated Theme
A	Commercial Banks	Banking and Finance
B	Building Societies	Banking and Finance
C	Insurance Agencies	Banking and Finance
D	Java	Programming Languages
E	C/C++	Programming Languages
F	Visual Basic	Programming Languages
G	Astronomy	Science
H	Biology	Science
I	Soccer	Sport
J	Motor Sport	Sport
K	Sport	Sport

4.2 Performance Measures

The evaluation of the different classifiers is measured using four different measures: recall (R), precision (P), accuracy (Acc), and F1 measure [17]. These can all be defined using the 'confusion matrix' shown as Table 2.

Table 2. Confusion Matrix

	Correct Class is C_k	*Correct Class is* $\overline{C_k}$
Assigned class is C_k	a	b
Assigned class is $\overline{C_k}$	c	d

$$R = \frac{a}{(a+c)} \quad \textit{if } (a+c) > 0 \textit{ otherwise } R = 1$$

$$P = \frac{a}{(a+b)} \quad \textit{if } (a+b) > 0 \textit{ otherwise } P = 1$$

$$Acc = \frac{(a+d)}{n} \quad \textit{where } n = a + b + c + d > 0$$

$$F1 = \frac{2PR}{(R+P)} = \frac{2a}{(2a+b+c)} \quad \textit{if } (a+c) > 0 \textit{ otherwise undefined.}$$

Recall (R) is the percentage of the documents for a given category that are classified correctly. Precision (P) is the percentage of the predicted documents for a given category that are classified correctly. Accuracy (Acc) is defined as the ratio of correct classification into a category C_k.

Neither recall nor precision makes sense in isolation from the other. In fact, a trivial algorithm that assigns class C_k to all documents will have a perfect recall (100%), but an unacceptably low precision. Conversely, if a system decides not to assign any document to C_k it will have a perfect precision but a low recall. The F1 measure has been introduced to balance recall and precision by giving them equal weights.

Classifying a document involves determining whether or not it should be classified in any or potentially all of the available categories. Since the four measures are defined with respect to a given category only, the results of all the binary classification tasks (one per category) need to be averaged to give a single performance figure for a multiple class problem.

In this paper, the 'micro-averaging' method will be used to estimate the four measures for the whole category set. Micro-averaging reflects the per-document performance of a system. It is obtained by globally summing over all individual decisions and uses the global contingency table.

4.3 Design of Experiments

The classification algorithms NB, KNN and C4.5 were applied to the dataset to address the different binary classification problems. The full set of 11 categories was used. The dataset was randomly split into 70% training and 30% testing.

Two local dictionaries were then built for each category after stop word removal (using a stop list of 512 words provided by David Lewis [18]), with the option of stemming turned either on or off. Documents were represented by VSM where the weights were computed using TF-IDF. Information Gain (IG) was used to select the best features.

Three series of experiments were conducted. The documents are represented by (i) the basic content of HTML documents or (ii) the metadata (keywords and description), title and link anchors, or (iii) a combination of basic html content, metadata, title and link anchors, with the consideration of extra weight assigned to metadata, title and link anchors. Table 3 gives the number of features for each of the eleven classes (A to K) and for each choice of text representation.

For all 11 classes, the 'metadata + title + link anchors' representation (with or without stemming) requires considerably fewer features than the other representations, which all include 'basic text content', and the processing time required to construct the local dictionary is correspondingly far less. Thus, if all other factors were equal, the preferred choice of representation would be Meta or MetaStem.

Table 3. Number of features for each class and for each choice of text representation

	Ct	CtStm	Mt	MtStm	CtMt	CtMtStm
A	8587	5649	1957	1513	11350	8478
B	8667	5972	1820	1427	11173	8673
C	11533	7649	2232	1827	13943	10282
D	20444	15231	7108	5922	23271	18118
E	26429	20407	6375	5265	27675	21968
F	15805	12154	5394	4397	18562	14972
G	29165	20722	9198	7681	32015	23636
H	69181	54713	9068	7644	70762	57117
I	27928	23108	5064	4409	29080	25718
J	21955	16327	5703	4970	23224	18099
K	36569	28352	8491	7026	38606	30824

Cell abbreviation

Ct (Content): basic text content with the stemming option turned off.
Ctstm (ContentStem): basic text content with the stemming option turned on.
Mt (Meta): metadata + title + link anchors with the stemming option turned off.
Mtstm (MetaStem): metadata + title + link anchors with the stemming option turned on.
Ctmt (ContentMeta): basic text content + metadata + title + link anchors with the stemming option turned off.

Ctmtstm (ContentMetaStem): basic text content + metadata + title + link anchors with the stemming option turned on.

4.4 Results and Interpretation

The different algorithms result in different performance depending on the features used to represent the documents.

The first set of experiments evaluates C4.5, NB and KNN, for texts represented using either (i) the basic content with the stemming option turned on or off, or (ii) meta data, title and link anchors with stemming option turned on or off, or (iii) a combination of basic content, metadata, title and link anchors with extra weight assigned to metadata, title and link anchors, with stemming option turned on or off.

Figure 1 and Figure 2 in Appendix 1 report, respectively, the performance accuracy and F1 measure on the test set of C4.5, NB and KNN for the different text representation options. They show that the use of stemming improves the performance of C4.5, NB and KNN for all the options of text representation. They also show that C4.5 outperforms NB and KNN for texts represented by basic content or metadata in both cases of stemming switched on or off. C4.5 and KNN do closely well for text represented by basic text content and metadata. Further, enhancing the basic content of texts with metadata improves the performance of all the classifiers.

Figures 3, 4 and 5 in Appendix 1 report the average accuracy of C4.5, NB and KNN (respectively) on the test set for the different text representation options and different feature numbers. Information gain has been used to select the best 20, 50, and 100 words that have the highest IG in each local dictionary. The figures show that, for the different vector space model sizes, stemming improves the accuracy of the considered learning algorithms. They show also that choosing the best 20 features from the metadata representation for text degrades the performance of C4.5, NB and KNN considerably. However, choosing the best 100 features from the different text representations is competitive with the accuracy of the classifiers used with all the features. Also, using feature selection to reduce the feature space model gives a huge reduction in the learning time of the classifiers. Figures 6, 7 and 8 in Appendix 1 report the F1 measure of C4.5, NB and KNN (respectively) on test set with the different options of text representation and different feature numbers. They report similar conclusions to those of figures 3, 4 and 5.

5. CONCLUSION AND FUTURE WORK

In summary, a number of experiments were conducted to evaluate the performance of some well-known learning algorithms on hypertext data. Different text representations have been used and evaluated. We applied C4.5, NB and KNN to a dataset of HTML documents. Basic text content or metadata, title, and link anchors, or basic text content along with metadata, title and link anchors were used for text representation. Information Gain was used for feature selection to reduce the feature space dimension. The following conclusions can be drawn.

- The use of stemming helps C4.5, NB and KNN improve their performance over the testing set.
- The use of basic text content enhanced with weighted metadata (metadata + title + link anchors) improves the performance of the 3 classifiers.
- The use of IG as a feature selection technique did not improve the accuracy of the classifiers, but instead helped reduce the text processing and learning time needed.
- C4.5 outperforms NB and KNN for the different text representation options.
- The accuracy of C4.5 used with the stemmed metadata for text representation (case1) is competitive with that of C4.5 used with basic text content enhanced with weighted metadata as text representation (case 2). In particular, the processing and learning time for case 1 is much less than that of case 2.

The use of the extra information hidden in HTML pages improved the performance of the different classifiers. In future work, this extra information will be extended by including the information in the 'linked neighborhood' of each web page. As a next step, features from outgoing and ingoing links will be used to enhance the text representation and evaluate the classifiers. Experiments with different datasets should also be conducted before final conclusions can drawn.

REFERENCES

[1] K.Bharat and A. Broader. A technique for measuring the relative size and overlap of public web search engines. In Proc. Of the 7th World Wide Web Conference (WWW7), 1998.

[2] H. Chen and S. Dumais. Bringing order to the web: automatically categorizing search results. In proceedings of CHI-00, ACM International Conference on Human Factors in Computing Systems, p 145-152, Den Haag, NL, 2000. ACM Press, New York, US.
[3] http://www.yahoo.com
[4] H. Oh, S. Myaeng, and M. Lee. A practical hypertext categorization method using links and incrementally available class information. In proceedings of the Twenty Third ACM SIGIR Conference, Athens, Greece, July 2000.
[5] J. Furnkranz. Exploiting structural information for text classification on the WWW. Proceedings of IDA-99, Third Symposium on Intelligent Data Analysis, pp 487-497, Amsterdam, NL, 1999.
[6] S. Slattery and T. Mitchell. Discovering test set regularities in relational domains. In Seventeenth International Conference on Machine Learning, June 2000.
[7] Y. Yang, S. Slattery and R. Ghani. A study of approaches to hypertext categorization. Journal of Intelligent Information Systems (Special Issue on Automatic Text Categorization) 18 (2-3) 2002, pp. 219-241.
[8] G.Attardi, A. Gulli, and F.Sebastiani. Automatic web page categorization by link and context analysis. In Proceedings of THAI'99, European Symposium on Telematics, Hypermedia and Artificial Intelligence, 1005-119, 1999.
[9] S.Chakrabati, B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. Proceedings ACM SIGMOD International Conference on Management of Data, pages 307-318, Seattle, Washington, June 1998. ACM Press.
[10] C. Apte, F. Damereau, and S. Weiss. Automated learning of decision rules for text categorization. ACM trans.Information Systems, Vol.12, No.3, July 1994, pp. 233-251.
[11] A.Bensaid and N. Tazi. Text categorization with semi-supervised agglomerative hierarchical clustering. International Journal of Intelligent Systems, 1999.
[12] G. Salton and McGill. Introduction to modern Information Retrieval. McGraw Hill, 1983.
[13] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. Information Processing and Management, 24 (5), pp 513-523, 1988.
[14] Y.Yang and J.Pederson. A Comparative study on feature selection in text categorization. International Conference on Machine Learning (ICML), 1997.
[15] http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/data/
[16] http://www.pedal.rdg.ac.uk/banksearchdataset/
[17] Y. Yang. An evaluation of statistical approaches to text categorization. Journal of Information Retrieval, 1999.
[18] D. Lewis. Feature selection and feature extraction for text categorization. Proceedings of Speech and Natural Language Workshop, 1992, pp. 212-217.

Appendix 1

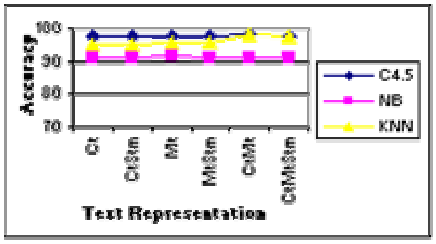


Figure 1. C4.5, NB and KNN accuracy for different choices of text representation

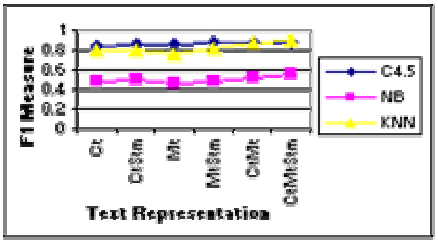


Figure 2. C4.5, NB and KNN F1 Measure for different choices of text representation

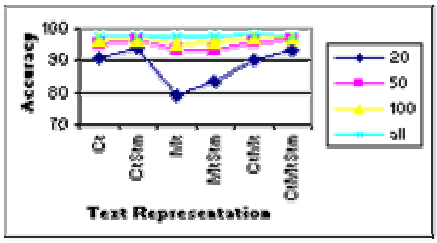


Figure 3. C4.5 accuracy for different choices of text representation

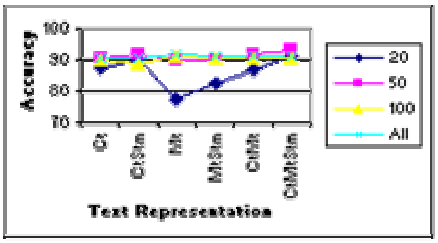
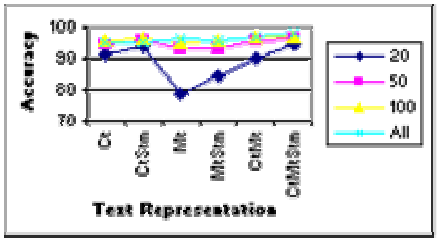


Figure 4. NB accuracy for different choices of text representation



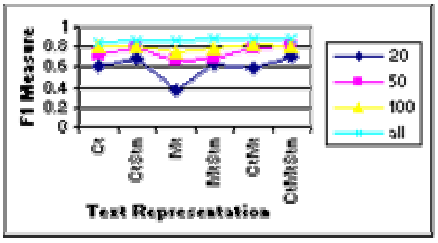


Figure 5: KNN accuracy for different choices of text representation

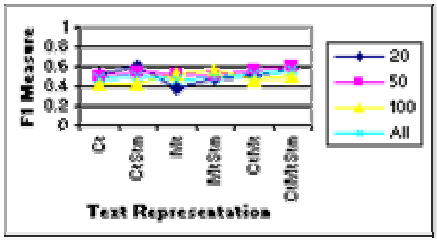


Figure 7. NB F1 Measure for different choices of text representation

Figure 6: C4.5 F1 Measure for different choices of text representation

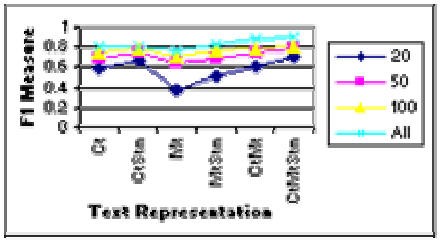


Figure 8. KNN F1 Measure for different choices of text representation