A&QT-R 2002 (THETA 13)
2002 IEEE-TTTC International Conference on Automation, Quality and Testing, Robotics
May 23 – 25, 2002, Cluj-Napoca, Romania

LEARNING WEB DOCUMENTS CATEGORIZATION BASED ON STRUCTURAL SIMILARITY

Calin CENAN, Ioan Alfred LETIA
Department of Computer Science, Technical University of Cluj-Napoca
Calin.Cenan@cs-gw.utcluj.ro, Ioan.Alfred.Letia@cs-gw.utcluj.ro

Abstract: Information Systems constitute one of the fastest evolving areas in Computer Science. In this area of research we will try to provide additional capability to retrieve information using such descriptions that are based on appearance of an HTML document. For this we will experiment using TiMBL, a program implementing several Memory Based Learning techniques. The task of the learning process will be to acquire a Web documents categorization based on the structure of the page. The results of our experiment did show that a kind of relation exist between categories and the appearance of a Web page as is reflected by the structure, however, to prove the validity of this theory, a significantly larger sample will be required.

Keywords: Machine Learning, Memory Based Learning, Categorization, Information Retrieval

1. INTRODUCTION

Information Systems comprises (but goes beyond) traditional database systems and in this area we have to consider a wide variety of information repositories from data residing not only in structured and centralized databases but also in various sites distributed on the Web. Recent revision lies in the nature of the data changing from traditional alphanumeric to multimedia, which is expensive to query and occupies a great deal of the so-called screen real estate.

A major challenge in indexing unstructured hypertext is to automatically extract meta-data that enables structured search using topic taxonomies. Such technique circumvents keyword ambiguity, and improves the quality of search and profile-based routing and filtering. Hypertext poses new challenges to automatic classifiers. The text classifier performed poorly because web documents are extremely diverse, featuring home pages, many are very short topical resource lists, other are generated automatically or active pages with scripts and links, etc.

In information retrieval systems making sense of all the data currently accessible, by using visual information and discovering its meaning, has also become a necessity. When we describe a Web page informally, we often use phrases like “it looks like…“. Unfortunately, no Web documents categorization provide the capability to retrieve information using such descriptions that are based on the appearance, i.e., structure, of the HTML document. In [6] is suggested that information, stored in the form of HTML tags, offers significant information about the nature of a Web page. Therefore we argue that it

may be used as a complementary source of information in automated search or classification systems. Visual structure clearly contain high quality semantic clues that are lost upon a purely term based classifier, but exploiting this information is not trivial. It is very difficult to classify documents according to how they look because concepts similar to "looks like … " are subjective. This paper explores new ways in which information latent in visual structure can be exposed to a suitably designed classifier. In order to use such concepts we will try to learn such categorization.

In Information Retrieval the focus is on discrete data, very large numbers of examples, many attributes of differing relevance and classification speed is a critical issue in any realistic application. Memory -Based Learning is a direct descendant of the classical *k* -Nearest Neighbor (*k*-NN) approach to classification and has proven to be successful in a large number of tasks [1]. We think it can make a useful tool for research in Information Systems domain when we try to learn from examples a classification of HTML documents based on their visual structure and not content.

2. MEMORY BASED LEARNER

We will use in our research TiMBL, the Tilburg Memory Based Learner from [7], a program implementing several MBL techniques. TiMBL stores a representation of the training set explicitly in memory, and classifies new cases by extrapolation from the most similar stored cases. On top of the classic *k*-NN classification several metrics, algorithms, and extra functions are implemented.

An MBL system [11] contains two components: a learning component that is based on memory and a performance component that is based on similarities. The learning component is based on memory as it involves adding training instances to the instance of case base without abstraction or restructuring. The performance component of an MBL system is used as a basis for mapping input to output that usually takes the form of performing classification. During classification, a previously unseen test example is presented to the system. The similarity between the new instance X and all examples Y in memory is computed using a distance metric $\Delta(X, Y)$. The extrapolation is done by assigning the most frequent category within the *k* most similar examples as the category of the new test example.

2.1. Algorithms

All the algorithms incorporated in TiMBL store some representation of the training set explicitly in memory and the main differences lie in the definition of similarity, the way the instances are stored in memory, and the way the search through memory is conducted. We can choose between the standard IB1 (nearest neighbor algorithm), the decision tree-based optimization IGTREE and the hybrid of the two: TRIBL.

It should be stressed that the choice of representation for instances remains the key factor determining the strength of the approach. Due to the tree storage structure instances with identical feature values are collapsed into one path and only their separate class

information needs to be stored in the distribution at the leaf node. Also instances that share a prefix of feature values also share a partial path.

In order to improve performance we can use Information Gain rather than un-weighted Overlap distance to define similarity in IB1 [12]. This positive effect direct to an alternative approach in which the instance memory is restructured in such a way that it contains the same information as before, but in a compressed decision tree structure, algorithm which is called IGTREE. In this algorithm, similar to the tree structured instance base described above, instances are stored as paths of connected nodes, which contain classification information. Nodes are connected via arcs denoting feature values. Information Gain as defined in [10] is used to determine the order in which instance feature values are added as arcs to the tree.

The application of IGTREE on a number of common machine learning data sets suggested that it is not applicable to problems where the relevance of the predictive features cannot be ordered in a straightforward way, e.g. if the differences in Information Gain are only very small. For this reason we can use TRIBL, a hybrid generalization of IGTREE and IB1. TRIBL allows you to exploit the tradeoff between optimization of search speed (as in IGTREE), and maximal generalization accuracy. When the Information Gain of a feature is below a given threshold, and the node is still ambiguous, tree construction halts and the leaf nodes at that point represent case bases containing subsets of the original training set. During search, the normal IGTREE search algorithm is used, until the case base nodes are reached, in which case regular IB1 nearest neighbor search is used on this sub case base.

2.2. Similarity metric

The most used metric that works for patterns with symbolic features is the Overlap metric given in equations (1) and (2); where $\Delta(X, Y)$ is the distance between patterns X and Y, represented by n features, and δ is the distance per feature. The distance between two patterns is simply the sum of the differences between the features. The distance metric in equation (2) simply counts the number of exact (mis)matching feature values in both patterns.

$$\Delta(X,Y) = \sum_{i=1}^{n} \delta(x_i, y_i) \quad (1)$$

$$\delta(x,y) = \begin{cases} \dfrac{x-y}{\max(x)-\max(y)}, & if\ x\ and\ y\ numerical \\ 0, & if\ x = y \\ 1, & if\ x \neq y \end{cases} \quad (2)$$

2.3. Feature weighting

Using this package we can choose from various weighting techniques to deal with features of different importance: information gain, gain ratio, chi-squared and shared variance. This feature weighting methods are used in the metric of IB1 and in the ordering of the IGTRE. In the absence of information about feature relevance we can presume that all the features have the same importance and the reasonable choice is of using no

weighting. Otherwise, we can add domain knowledge bias to weight or select different features. We can compute statistics about the relevance of features by looking at which features are good predictors of the class labels [3]. Information Theory gives us a useful tool for measuring feature relevance in this way. Information Gain weighting, looks at each feature in isolation, and measures how much information it contributes to our knowledge of the correct class label. The Information Gain of feature *i* is measured by computing the difference in uncertainty, entropy, between the situations without and with knowledge of the value of that feature.

Information Gain tends to overestimate the relevance of features with large numbers of values. To normalize Information Gain for features with different numbers of values, Quinlan has introduced a normalized version, called Gain Ratio, which is Information Gain divided by *si(i)* (split info), the entropy of the feature values. Unfortunately, the Gain Ratio measure, just as all information based measures, does also have a bias towards features with more values. The reason for this is that the Gain Ratio statistic is not corrected for the number of degrees of freedom. To solve this problem in statistical studies is proposed a feature selection measure based on the Chi squared value.

3. EXPERIMENTS AND RESULTS

The main idea of our paper is that knowledge, stored in the form of HTML tags, offers significant information about the nature of a Web page and, therefore, may be used as a complementary source of information in automated search or classification systems. We introduce some types of distance to measure structural similarity between Web documents. Perhaps the easiest way to compute distances between two Web documents is to eliminate everything except HTML tags and then for each HTML tag, compute its frequency (in %) in a document, and summarize the frequencies [4]. If F_{k1} is the frequency of tag k in the first document and F_{k2} is the frequency of the same tag in the second document the distance can then be computed as:

$$d = \sum_k (F_{k1} - F_{k2})^2 w_k \qquad (4)$$

We can improve this distance metric by assigning w_k, the weight of the corresponding tag, depending on how "predictable" the tag frequency is in its category.

This method relies on the assumption that tag frequencies reflect some inherent characteristics of a Web document and correlate with its structure. In other words, the assumption is that those frequencies are not aimless and they are not the same for documents with different structures.

This technique was used in an experiment that we conducted in order to explore possible correlation between structure and keywords topic. We have sampled Web pages and for each of the three topics corresponding URLs were processed. An approximate number of 600 Web pages were retrieved for an average of two thousand valid pages per topic. We take three different types of Web documents: professor’s home pages, university courses and news. The news pages were taken from two different types of sources: on-line newspapers and Internet news streams. To increase the relevance of the solution we take

documents both in English and Romanian language considering that the approach must learn the structural similarities of documents which must be the same in both languages.

We considered different types of tags in a HTML document. Some of them (<B>, <I>, <U>, <FONT>, etc.) only affect the way some fragment of the text is presented, without changing the document's format. Other tags actually induce the structure, and they are of the most interest to us (<TABLE>, <TR>, <TD>, <TH>, <DL>, <DD>, <OL>, <LI>, <UL>, <LI>, <CENTER>, <P>,
, <HR>). There are also HTML tags expected to appear only once in any Web document (<HTML>, <HEAD>, <TITLE>, <BODY>).

After we obtained these data we tried to learn a classifier using TiMBL, Memory Based Learning to decode a possible correspondence from this measures to different types of Web documents. The goal of the learning process is to obtain a classifier for the corresponding three classes of Web documents investigated: news, university courses and professor's home pages. For this we with a set off all the documents available together with the proper classification. In our experiments the learning process to obtain the classifier is based only on subsets of documents and in the final we always test the entire set.

The main result obtained is a quiet good categorization for HTML documents, based on visual structure, which can be learned with the nearest neighbor method. Even if we used for learning only 5% of the document’s set a not so bad (50%) categorization is learned. If we extend the percent of Web pages from which to learn to 10% the performance increase to an exact classification of 60-70% of the Web pages from the test set. The best results are obtained with TRIBL and IGTREE algorithms with Chi-square and Gain ration as weighting methods. These can be seen from the next table in which we describe our experiments considering different number of HTML tags, MBL algorithms and size for the learning set

No. Tags	Algorithm	Learn Set	Result
34	TRIBL, k=1, Weighting: Chi-square	60	386/575 (0.671304), 68 exact matches
34	TRIBL, k=1, Weighting: Shared Var.	60	372/575 (0.646957), 68 exact matches
34	TRIBL, k=1, Weighting: Chi-square	102	405/575 (0.704348), 119 exact matches
34	TRIBL, k=1, Weighting: GainRatio	102	391/575 (0.680000), 119 exact matches
70	IB1, k=1, Weighting: GainRatio	60	358/572 (0.625874), 60 exact matches
70	IGTREE	100	339/572 (0.592657)
70	TRIBL, k=1, Weighting: Chi-square	100	410/572 (0.716783), 148 exact matches
70	TRIBL, k=1, Weighting: GainRatio	100	407/572 (0.711538), 148 exact matches
70	IB1, k=1, Weighting: GainRatio	100	399/572 (0.697552), 148 exact matches
70	IB1, k=1, Weighting: Chi-square	100	368/572 (0.643357), 148 exact matches

In our comparison of the three learning algorithms, in all the experiments conducted, the best results were obtained with IB1 and TRIBL and less good results with IGTREE. Usually IGTREE fails to correctly classify from the test set around 10% less documents then the categorizations obtained with the IB1 and TRIBL. The results obtained goes from 90% documents correctly classified when we learn using a large number of examples to a reduction of performance to 60-70% when the set of examples for learning decrease to 5 or 10% documents from the initial set.

The chosen weighting methods for the TRIBL and IB1 learning algorithms has no great impact on performance. Experimenting with various values for all the rest of learning

parameters we concluded, as theory suggested, that Chi-square weighting method seems to be the best. The increase in performance for using this method is only 2-3% greater then when we use different weighting like Gain Ratio, Shared Variance or Information Gain.

As we expected the value of the parameter *k* from *k*-nearest neighbor does not have a big impact on performance. Since our experiment was carried with a limited number of classes, actual only three, increasing the value of *k* from the learning algorithm and considering more then one possible classes has a unfavorable impact for the learning performance which decrease with around 5%.

4. CONCLUSION AND FURTHER RESEARCH

While this approach certainly makes a solid argument for the possibility of a relationship between Web document appearance and the associated categorization, insufficient statistical evidence was given to prove this hypothesis. The results did show that a relationship existed between categories, however, to prove the validity of this theory, a significantly larger sample will be required. The focus of a subsequent research is to provide the statistical data necessary to show that structural analysis of a Web document can lead to the appropriate classification. We further want to devise an architecture similar to that presented in [2] and [5] for a hypertext search engine that transcend keyboard-based approach including the structure of a HTML document. We have demonstrated that memory based learned classifier, based on visual structure, can be used as an additional tool to structure the Web content.

[1] D. W. Aha, [1997], Lazy Learning, Special edition Editorial, *Artificial Intelligence Review*, No. 11, Pag. 7-10.
[2] S.Chakrabrati, M. van den Berg and B.E. Dorn [1999], Distributed Hypertext Resource Discovery Through Examples, Proceedings of Conference on Electronic Publishing, St. Malo, France.
[3] S. Cost and S. Salzberg, [1993]. A Weighted Nearest Neighbour Algorithm for Learning with Symbolic Features, *Machine Learning*, No. 10, Pag. 57-78.
[4] I.F. Cruz, S. Borisov, M.A. Marks, and T.R. Webb, [1998], Measuring Structural Similarities Among Web Documents: Preliminary Results, Lecture Notes in Computer Science, Vol. 1375, Pag. 513-524
[5] I.F. Cruz, L.L. Liu, and T.Y. Wu [2000], WEBSA: Database Support for Efficient Web Site Navigation, , *Proceedings of Conference on Visual Database Systems* (VDB5), Fukuoka, Japan, Pag. 301-320.
[6] I.F. Cruz and W.T. Lucas, [1997], A Visual Approach to Multimedia Querying and Presentation, Proceedings of the Fifth ACM International Conference on Multimedia, Seattle, USA, Pag. 109-120.
[7] W. Daelemans, [1999], Machine Learning Approaches, *Syntactic Wordclass Tagging*, Kluwer Academic Publishers, Pag. 285-304
[8] W. Daelemans, A. Van den Bosch, and A. Weijters, [1997], IGTREE: Using Trees for Compression and Classification in Lazy Learning Algorithms, *Artificial Intelligence Review*, No. 11, Pag. 407-423.
[9] T. Mitchell, [1999], *Machine Learning*, McGraw-Hill, 414.
[10] J.R. Quinlan, [1986], Induction of Decision Trees, *Machine Learning*, No. 1, Pag. 81-206.
[11] C. Stanfill and D. Waltz, [1986], Toward Memory-Based Reasoning, *Communications of the ACM*, Vol. 29, No. 12, Pag. 1213-1228.
[12] A.P. White and W.Z. Liu, [1994], Bias in information-based measures in decision tree induction, *Machine Learning*, Vol. 15, No. 3, Pag. 321-329.