Impact of a New Attribute Extraction Algorithm on Web Page Classification

Göksel Biricik, Banu Diri, *Yildiz Technical University, Computer Engineering Department*

***Abstract*—This paper introduces a new algorithm for dimensionality reduction and its application on web page classification. A heterogeneous collection of web pages is used as the dataset. Selected attributes for classification are the textual content of pages. Using the offered algorithm, high dimension of attributes- words extracted from the pages- are projected onto a new hyper plane having dimensions equal to the number of classes. Results show that processing times of classification algorithms dramatically decrease with the offered reduction algorithm. This mostly relies on the number of attributes given to classifiers fall off. Accuracies of the classification algorithms also increase compared to tests run without using the proposed reduction algorithm.**

I. Introduction

THE rapid and exponential growth of web pages on the internet makes difficult to organize relevant pages together. People tend to organize data sources into categories depending on the content because it is easier to track between categories and reach the desired data. Dewey and Library of Congress Headings are the examples of classification systems used in libraries to achieve categorization. Web pages are not logically organized in the internet which makes the data access and retrieval difficult. When we look at the internet, we can find web page classification solutions in forms of web directories like Dmoz or Yahoo!. The problem is that these directories are formed by human effort- with the hands and minds of the editors. The web is in fact several times bigger than the coverage of these directories plus search engines. That's why automation of this categorization process is important. Classification algorithms may help editors to classify new coming web pages into existing categories. This is also known as web page classification. There are many solutions proposed to help solve this problem. Dumais and Chen [1] offers hierarchical classification using SVM's on LookSmart web directory. Choi and Peng [2] offers a dynamic and hierarchical classification scheme. Mladenic uses Yahoo web directory with a Naïve Bayes classifier [3]. Holden and Freitas classify their web dataset using ant-colony algorithm [4]. Many other researches can be found in literature about web page classification problem.

One of the models widely used in text classification is the vector space model in which the documents are represented as vectors described by a set of identifiers, for example, words as terms. This model is also known as bag-of-words model, every document is just a container for the words it contains. Thinking in the vector space, each term is a dimension for the document vectors. The nature of this representation causes a very high-dimensional and sparse feature space, which is a common problem to deal with when using bag-of-words model. There are two effective ways to overcome this dimensionality problem: Attribute selection and attribute extraction.

Attribute selection algorithms output a subset of the input attributes, results in a lower dimensional space. Instead of using all words as attributes, attribute selection algorithms evaluate attributes on a specific classifier to find the best subset of terms [5]. This results in reduced cost for classification and better classification accuracy. The most popular attribute selection algorithms include document frequency, chi statistic, information gain, term strength and mutual information [6]. Chi-square and correlation coefficient methods have been shown to produce better results than document frequency [7]. The lack of attribute selection algorithms is that the selection procedure is evaluated on a certain classifier. The produced subset may not be suitable for another classifier to improve its performance.

Attribute extraction algorithms simplifies the amount of resource required to describe a large set of data. The high-dimensional and sparse structure of vector space model requires large amount of memory and computation power. The aim of attribute extraction is to combine terms to form a new description for the data with sufficient accuracy. Attribute extraction can be seen as projecting the high-dimensional data into a new, lower-dimensional hyperspace. Mostly used techniques are principal components analysis (PCA), Isomap and Latent Semantic Analysis (LSA). Latent Semantic Indexing is based on LSA and it is the mostly used algorithm in text mining tasks nowadays. Our algorithm also extracts attributes from the original vector space, projects into a new hyperspace with dimensions equal to number of classes.

In section 2, we introduce our evaluation dataset and the preprocessing steps. Section 3 gives brief description about related attribute extraction algorithms and introduces the proposed method. In section 4 we discuss our experimental results. Section 5 addresses conclusions and feature work.

G.Biricik is with the Computer Engineering Department, Yildiz Technical University, Istanbul, 34349 Turkey (e-mail: goksel@ce.yildiz.edu.tr).

B.Diri is with the Computer Engineering Department, Yildiz Technical University, Istanbul, 34349 Turkey (e-mail: banu@ce.yildiz.edu.tr).

II. EVALUATION DATASET AND PREPROCESSING

A. Evaluation Dataset

Evaluation dataset is prepared from web pages under Dmoz [8] category World/Turkce. We work to develop an automated classifier for the Dmoz Turkish directory. We used the dataset that we prepared for our study because the attribute extraction algorithm we introduce in this paper is a part of our system. World/Turkce category included 28567 unique web pages in Turkish at the time we crawled. There are thirteen main categories in this set:

1. Shopping
2. News
3. Computers
4. Science
5. Regional
6. Recreation
7. Business
8. Reference
9. Arts
10. Games
11. Health
12. Sports
13. Society

Each category has several sub categories (total 758), which are not considered at this time.

B. Preprocessing

The dataset is crawled in raw HTML format. It is then parsed with HTML Parser[1] to fetch the content of web pages.

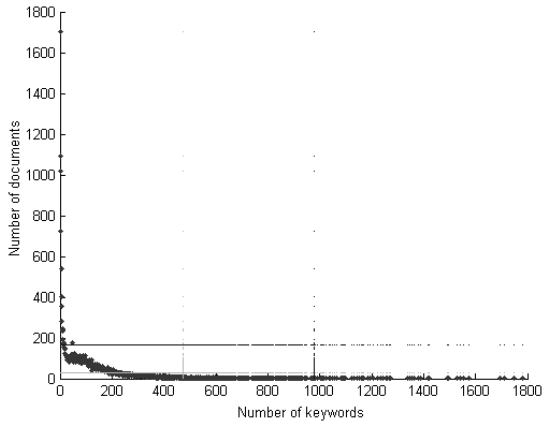

Fig.1. Distribution of terms and documents in the dataset. We can see that most of the dataset elements contain less than 200 keywords.

Term statistics are calculated to see the outlying documents. Fig.1 shows the distribution of terms and documents, in other words, the number of words documents have and the number of documents with a certain amount of terms. This plot shows us that some documents contain lots of keywords, while some have very few.

A filter similar to Box-Plot is used to find the outliers in this distribution. The mean and standard deviations of the axes are calculated and a box is drawn on the distribution with the center ($mean_x$, $mean_y$) and boundaries (σ_x, σ_y). The area within the box is used as the input dataset; the samples outside this box are considered as outliers and cleared. The center of the box and the boundaries are also shown in Fig.1.

The dataset fetched out of Dmoz's World/Turkce category is in Turkish. Because Turkish is a suffix language, meanings of the words change with suffixes. To get rid of this effect, all of the terms are stemmed into their roots by using a Turkish stemmer tool named Zemberek[2].

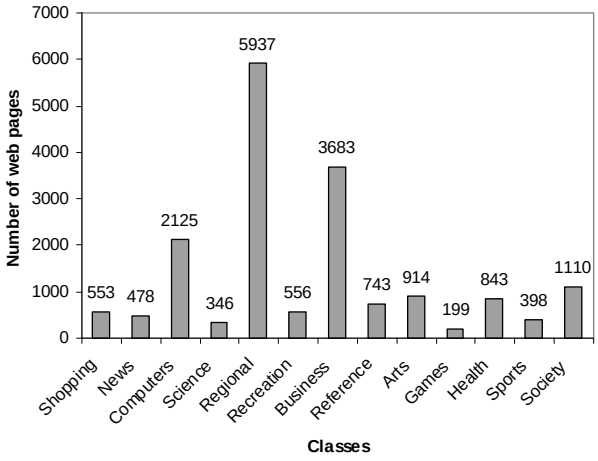

Fig.2. Distribution of web pages among classes after preprocessing step.

The final preprocess step is cleaning up the Turkish stop words. After the preprocessing step, we have 17.885 documents (web pages) and 18.363 unique keywords. Looking from the bag-of-words view, we have a term-document matrix with 17.885 rows and 18.363 columns. Because any of the documents will contain only a tiny subset of our terms, our matrix is very sparse. The cells of the matrix contain tf-idf values of terms calculated by (1), (2) and (3), where n_i is the number of occurrences of the considered term in document d_j, $|D|$ is the total number of documents, $|\{d_j:t_i \in d_j\}|$ is the number of documents where term t_i appears..

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \quad (1)$$

$$idf_i = \log \frac{|D|}{|\{d_j : t_i \in d_j\}|} \quad (2)$$

$$tfidf_{i,j} = tf_{i,j} \times idf_i \quad (3)$$

Our matrix leads us to a 18.363 dimensional vector space,

[1] Available at http://htmlparser.sourceforge.net/

[2] Available at http://code.google.com/p/zemberek/

which is hard to deal with because running classification algorithms on such a multidimensional space is very time and memory consuming. The distribution of web pages among 13 classes is also uneven, which causes another problem for classification algorithms. Number of web pages per class after preprocessing step is given in Fig.2.

III. Attribute Extraction

Attribute extraction algorithms simplifies the amount of resource required to describe a large set of data. The aim of attribute extraction is to combine terms to form a new description for the data with sufficient accuracy. In this section, commonly used extraction algorithms -principal component analysis, PCA and latent semantic analysis, LSA- and the proposed method are introduced. Our algorithm projects attributes into a new hyperspace with dimensions equal to number of classes.

A. Related Research

1) Principal Component Analysis: PCA transforms correlate variables into a smaller number of correlated variables- principal components. Invented by Pearson in 1901 [9], it is mostly used for exploratory data analysis.

PCA is used for attribute extraction by retaining the characteristics of the dataset that contribute most to its variance, by keeping lower order principals, which tend to have the most important aspects of data. This is done by a projection into a new hyper plane using Eigen values and Eigen vectors.

PCA is a popular technique in pattern recognition, but its applications are not very common because it is not optimized for class separability [10]. It is widely used in image processing,

2) Latent Semantic Analysis: Patented in 1988, LSA is a technique that analyzes relationships between a document set and the terms that they contain by producing a set of concepts related to them [11]. As the name suggests, singular value decomposition breaks our matrix down into a set of smaller components.

LSA uses singular value decomposition (SVD) method to find the relationships between documents. Singular value decomposition breaks term-document matrix down into a set of smaller components. The algorithm alters one of these components (reduces the number of dimensions), and then recombines them into a matrix of the same shape as the original, so we can again use it as a lookup grid. The matrix we get back is an approximation of the term-document matrix we provided as input, and looks much different from the original.

LSI is mostly used for web page retrieval, document clustering purposes. It is also used for document classification via voting, or information filtering. Many algorithms are combined with LSI to improve performance by working in a less complex hyperspace.

B. Proposed Attribute Extraction Method

Our attribute extraction algorithm neither uses Eigen values, Eigen vectors, nor singular value decomposition. Weights of terms and their probabilistic distribution over the classes are taken into account. We project term probabilities to classes, and sum up those probabilities to get the impact of each term to each class [15].

Assume we have I terms, J documents and K classes. Let $n_{i,j}$ be the number of occurrences of term t_i in document d_j and N_i be the total number of documents that contain t_i. We calculate the total number of occurrences of a term t_i in class c_k with (4). We calculate the weight of term t_i (in a document d_j) that affects class c_k with (5).

$$nc_{i,k} = \sum_j n_{i,j}, \qquad d_j \in c_k \tag{4}$$

$$w_{i,k} = \log\left(nc_{i,k} + 1\right) \times \log\left(\frac{N}{N_i}\right) \tag{5}$$

We calculate the total effect of terms in a document over a class c_k with (6). At the end, I terms are projected onto number of classes; we have K new terms in hand for a document. Repeating this procedure for all documents gives us a reduced matrix with J rows (one row per document) and K columns (number of extracted features equals the number of classes). Finally, we normalize new terms by using (7).

$$Y_k = \sum_i w_{i,k}, \qquad w_i \in d_j \tag{6}$$

$$newTerm_k = \frac{Y_k}{\sum_k Y_k} \tag{7}$$

We give a brief example to explain the algorithm comprehensibly. Assume we have 8 terms, 6 documents and 2 classes given in Fig.3. The corresponding term-document matrix is in Table 1. The cells of Table 1 correspond to term weights calculated with (4).

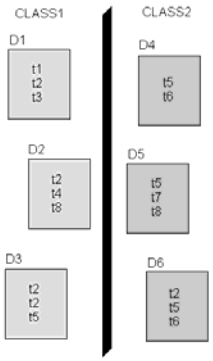

Fig.3. Sample dataset to explain the proposed extraction algorithm comprehensibly.

TABLE I
TERM-DOCUMENT MATRIX OF THE SAMPLE DATASET

	D1	D2	D3	D4	D5	D6
t1	1	0	0	0	0	0
t2	1	1	2	0	0	1
t3	1	0	0	0	0	0
t4	0	1	0	0	0	0
t5	0	0	1	1	1	1
t6	0	0	0	1	0	1
t7	0	0	0	0	1	0
t8	0	1	0	0	1	0

We calculate term weights over classes with (5). This produces *I* by *K* matrix, given in Table 2. After applying (6) and (7), we have the reduced *J* by *K* matrix with the extracted features of the processed documents. Result matrix is given in Table 3. This matrix is visualized in Fig.4. We can also read the extracted values as the membership probabilities of the documents to classes, as seen on Fig.4. For example, the content of D4 tells us that it is 84% Class2 and 16% Class1. Thus, *the extraction method we propose may also be used as a stand-alone classifier*. When we feed a new document, the system decides which class this document belongs to by calculating (6) and (7) with the new document's content.

TABLE II
TERM WEIGHTS OVER CLASSES FOR THE SAMPLE DATASET

	C1	C2
t1	0.234	0
t2	0.123	0.053
t3	0.234	0
t4	0.234	0
t5	0.053	0.106
t6	0	0.228
t7	0	0.234
t8	0.144	0.144

TABLE III
TERM WEIGHTS OVER CLASSES FOR THE SAMPLE DATASET

	C1	C2
D1	0.918	0.082
D2	0.718	0.282
D3	0.585	0.415
D4	0.137	0.863
D5	0.289	0.711
D6	0.313	0.687

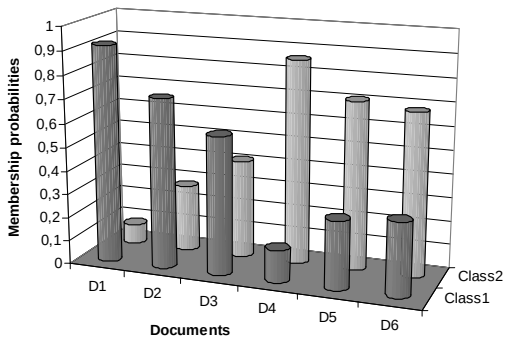

Fig.4. Visualization of Table III. Extracted values can be taken as the membership probabilities of the documents to the classes.

IV. EXPERIMENTAL RESULTS

We prepare our evaluation dataset and test our attribute extraction algorithm's performance using 5 different classifiers. We use 10-fold cross-validation for testing.

We use Naïve Bayes as a simple probabilistic classifier, which is based on applying Bayes' theorem with strong independence assumptions [12]. We choose J48 Tree, an implementation of Quinlan's C4.5 decision tree algorithm [13] for a basic tree based classifier, and a random forest with 100 trees to construct a collection of decision trees with controlled variations [14]. We choose radial basis functions network for a function based classifier and K-nearest neighbor algorithm with K=10 to test instance-based classifiers.

We evaluate the classifiers' performance with their classification accuracy which is calculated by (8) and F-Measure by using (9) .The classification performance results are given in Table 4. We can see that K-nearest neighbor classifier fails because our documents are unevenly distributed between classes. *Regional* and *Business* classes have much more instances than the other classes; this is the reason why an instance-based learner fails. RBF network also fails because the distribution of the attributes can not be easily modeled with Gaussian local models.

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{8}$$

$$F = \frac{2 \times TP}{(2 \times TP + FP + FN)} \tag{9}$$

TABLE IV
RESULTS FOR THE CLASSIFIERS EVALUATED ON OUR DATASET WITHOUT USING ATTRIBUTE EXTRACTION

	Classification Accuracy	F-Measure
Naïve Bayes	55.7%	0.591
J48 Tree (C4.5)	57.7%	0.572
Radial Basis Functions Network	32.4%	0.221
K-Nearest Neighbor (K=10)	31.8%	0.245
Random Forest (with 100 trees)	61.4%	0.583

TABLE IV
RESULTS FOR THE CLASSIFIERS EVALUATED ON OUR DATASET USING THE PROPOSED ATTRIBUTE EXTRACTION METHOD

	Classification Accuracy	F-Measure
Naïve Bayes	67.5%	0.675
J48 Tree (C4.5)	58.7%	0.589
Radial Basis Functions Network	70.6%	0.705
K-Nearest Neighbor (K=10)	74.3%	0.740
Random Forest (with 100 trees)	75.5%	0.753

We give the classification performance results of the classifiers using our proposed attribute extraction algorithm in Table 5. We can see that our attribute extraction algorithm increases the classification accuracy by 11.8% when using Naïve Bayes, 1.0% using J48, 38.2% using RBF network, 42.5% using K-nearest neighbor, and 14.1% using a random forest. Another outcome of our method is that it decreases classification times on both classifiers dramatically.

Fig.5 compares the classification accuracies and Fig.6 compares F-measures with and without using our extraction method. We can see that both classification accuracies and F-measures increase. Our method highly improves the performance of K-nearest neighbor classifier on unevenly distributed dataset. The performance of RBF network is also boosted because the final attributes tend to have a Gaussian like distribution.

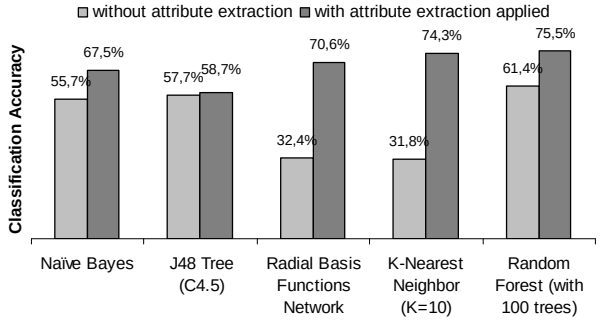

Fig.5. Comparison of classification accuracies with and without using the proposed attribute extraction algorithm.

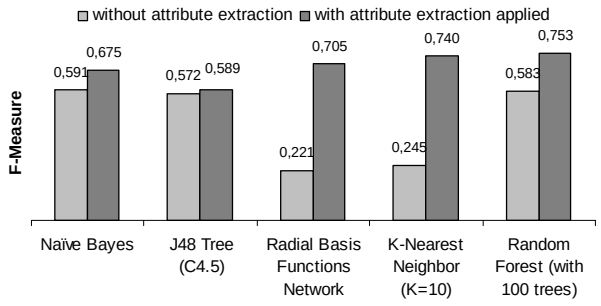

Fig.6. Comparison of F-measures of the classifiers with and without using the proposed attribute extraction algorithm.

V. Conclusion and Future Work

We introduce a new algorithm for attribute extraction and its application on web page classification. We use a heterogeneous collection of web pages crawled from Dmoz's World/Turkce as the dataset. Selected attributes for classification are the textual content of the web pages, in other words, preprocessed terms according to vector-space model. Using the offered algorithm, high dimension of attributes- terms extracted from the pages- are projected onto a new hyper plane having dimensions equal to the number of classes. Results show that processing times of classification algorithms dramatically decrease with our reduction algorithm. This mostly relies on the number of attributes given to classifiers fall off. Accuracies of the classification algorithms also increase compared to tests run without using the proposed reduction algorithm, especially on the classifiers that fail on unevenly distributed datasets.

We plan to compare our attribute extraction method with other algorithms like LSA and PCA as a future work. As we see that our method can also act as a classifier, we program to test our method as a classifier with other classification algorithms. Furthermore, we have in view to improve our method to fit hierarchical classes.

References

[1] S.Dumais and H.Chen, "Hierarchical Classification of web Content", In Proc. of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval, Athens, Greece, 2000, pp 256–63.

[2] B.Choi, X.Peng, "Dynamic and Hierarchical Classification of Web Pages", Online Information Review, vol.28, no.2, pp.139-147, 2004.

[3] D.Mladenic, "Turning Yahoo into an Automatic Web-Page Classifier", 13th European Conference on Artificial Intelligence Young Researcher Paper, 1998.

[4] N.Holden, A.A.Freitas, "Web Page Classification with an Ant Colony Algorithm", in Parallel Problem Solving from Nature – PPSN VIII, vol.3242, Springer Berlin-Heidelberg, 2004, pp.1092-1102.

[5] Y.Yiming, J.O.Pedersen, "A comparative study on feature selection in text categorization", in 14th International Conference on Machine Learning, 1997, pp.412-420..

[6] J.Zhu, H. Wang and X.Zhang, "Discrimination-Based Feature Selection for Multinomial Naïve Bayes Text Classification," in 2006 Proc. ICCPOL2006, LNAI 4285, pp.149-156.

[7] R.Jensen, Q.Shen, "Computational Intelligence and Feature Selection Rough and Fuzzy Approaches", IEEE-Wiley, New Jersey, 2008, pp.203-218.

[8] Directory Mozilla Project, DMOZ, available at: http://www.dmoz.org.

[9] K.Pearson, "On Lines and Planes of Closest Fit to Systems of Points in Space", Philosophical Magazine2(6):pp.559-572, 1901.

[10] K.Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, London,1990.

[11] T.K. Landauer, S.T. Dumais, "Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction and Representation of Knowledge." Psychological Review, 1997, vol:104, no:2, pp. 211-240.

[12] A. McCallum, K. Nigam, "A Comparison of Event Models for Naive Bayes Text Classification". In Proc. AAAI/ICML-98 Workshop on Learning for Text Categorization, 1998, pp. 41-48.

[13] J.R. Quinlan, C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

[14] L.Breiman, "Random Forests". Machine Learning, vol:45, no:1, pp.5-32, 2001.

[15] Z.Kaban, B.Diri, "Recognizing Type and Author in Turkish Documents Using Artificial Immune Systems", 16th Signal Processing and its Applications Congress, Turkey, 2008.