International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:1, No:5, 2007

Hybrid Machine Learning Approach for Text Categorization

Nerijus Remeikis, Ignas Skučas, Vida Melninkaitė

***Abstract*—** Text categorization - the assignment of natural language documents to one or more predefined categories based on their semantic content - is an important component in many information organization and management tasks. Performance of neural networks learning is known to be sensitive to the initial weights and architecture. This paper discusses the use multilayer neural network initialization with decision tree classifier for improving text categorization accuracy. An adaptation of the algorithm is proposed in which a decision tree from root node until a final leave is used for initialization of multilayer neural network. The experimental evaluation demonstrates this approach provides better classification accuracy with Reuters-21578 corpus, one of the standard benchmarks for text categorization tasks. We present results comparing the accuracy of this approach with multilayer neural network initialized with traditional random method and decision tree classifiers.

***Keywords*—** text categorization, decision trees, neural networks, machine learning.

I. Introduction

As the volume of information continues to increase, there is growing interest in helping people better find, filter, and manage these resources. Text categorization - the assignment of natural language documents to one or more predefined categories based on their semantic content - is an important component in many information organization and management tasks. Automatic text categorization task can play an important role in a wide variety of more flexible, dynamic and personalized tasks as well: real-time sorting of email or files, document management systems, search engines, digital libraries.

A number of statistical classification methods and machine learning techniques have been applied to text categorization, including techniques based on decision trees [6], neural networks [11], Bayes probabilistic approaches [6]. However there is still need more accurate text classifiers based on new learning machine learning approaches.

The purpose of the current work is to describe ways in which hybrid machine learning method can be applied to the problem of text categorization, and to test its performance relative to a number of other text categorization algorithms. In this paper, we introduce the use of a hybrid decision tree and neural network technique to the problem of text categorization, because hybrid approaches can simulate human reasoning in a way that a decision tree learning is used to do qualitative analysis and neural learning is used to do subsequent quantitative analysis. This approach is based on hybrid algorithm, which was described in paper [1] and has been shown to perform well on standard machine learning tasks. The main idea of this method is to transform decision trees to neural networks. Hybrid machine learning method for text categorization task constructs the networks by directly mapping decision nodes or rules to the neural units and compresses the network by removing unimportant and redundant units and connections.

N. Remeikis is with the Faculty of Informatics, Vytautas Magnus University, Vileikos str. 8, LT-44404 Kaunas, Lithuania (e-mail: Nerijus_Remeikis@fc.vdu.lt).

I. Skučas is with the Faculty of Informatics, Vytautas Magnus University, Vileikos str. 8, LT-44404 Kaunas, Lithuania (e-mail: Ignas_Skucas@fc.vdu.lt).

V. Melninkaitė is with the Faculty of Informatics, Vytautas Magnus University, Vileikos str. 8, LT-44404 Kaunas, Lithuania (e-mail: Vida_Melninkaite@fc.vdu.lt).

II. Problem Definition

Automatic text categorization has always been an important application and research topic since the inception of digital documents. The text classification task can be defined as assigning category labels to new documents based on the knowledge gained in a classification system at the training stage. A wide range of supervised machine learning algorithms has been applied to this area using a training data set of categorized documents. Although text classification performance results has been quite encouraging, but there is still need more accurate text classifies based on new learning machine learning approaches [9].

This paper presents our attempt to improve text classification accuracy by hybrid decision tree and neural network approach and to compare classification performance to previous researches results.

III. an Adapted Hybrid Approach for Text Categorization

A. Decision tree construction algorithm

Algorithm constructs the decision tree with a *divide and conquer* strategy. Each node in a tree is *associated* with a set of cases. At the beginning, only the root is present, with associated the whole training set and with all case weights equal to 1. At each node the following *divide and conquer* algorithm is executed, trying to exploit the locally best

choice, with no backtracking allowed.

Let T be the set of cases associated at the node. The weighted frequency $freq(C_i, T)$ is computed of cases in T whose class is C_i, $i \in [1, N]$.. If all cases in T belong to a same class Cj (or the number of cases in T is less than a certain value) then the node is a leaf, with associated class C_j.

If T contains cases belonging to two or more classes, then the *information gain* of each attribute is calculated:

$$I = H(T) - \sum_{i=1}^{s} \frac{|T_i|}{|T|} \times H(T_i) \quad (1)$$

where

$$H(T) = -\sum_{j=1}^{n} \frac{freq(C_j,T)}{|T|} \times \log_2(\frac{freq(C_j,T)}{|T|}), \quad (2)$$

is the entropy function.

While having an option to select information gain, by default, however, C4.5 [7] considers the *information gain ratio* of the splitting $T_1,...,T_s$ which is the ratio of information gain to its split information:

$$S(T) = -\sum_{i=1}^{s} \frac{|T_i|}{|T|} \times \log_2(\frac{|T_i|}{|T|}) \quad (3)$$

B. Training multilayer neural network

The neural network is a machine learning method, which has a powerful ability to separate non-linear classes. In this research, a feedforward network with back- propagation [3, 8], the most widely used learning algorithm, is implemented. Each neuron j in the hidden layer sums up its input x_i after weighting them with the strengths of the respective connections w_{ji} from the input layer and computes its output y_j as a function f of the sum as follows

$$y_j = f(\sum w_{ji} x_i) \quad (4)$$

where f can be a simple threshold function, a linear or a sigmoid function.

Back-propagation training algorithm uses a gradient-descent algorithm to minimise the mean squared difference between the neural network output and the desired output. The change in weight between neurons i and j is as follows

$$\Delta w_{ji}(k) = \eta \delta_j x_i + \alpha \Delta w_{ji}(k-1) \quad (5)$$

where η is a parameter called the learning coefficient, α is the momentum coefficient, and δ_j depends on output neuron or a hidden neuron. For output neurons

$$\delta_j = \frac{\partial f}{\partial net_j}(y_j - y_{net-j}) \quad (6)$$

where $net_j = \sum_i x_i w_{ji}$, y_j, y_{net-j} are the target and the neural outputs for neuron j, respectively. For hidden neurons

$$\delta_j = \frac{\partial f}{\partial net_j} \sum_q w_{qj} \delta_q \quad (7)$$

C. Hybrid approach

If we compare decision trees and neural networks we can see that their advantages and drawbacks are almost complementary. For instance humans easily understand knowledge representation of decision trees, which is not the case for neural networks. Decision trees have trouble dealing with noise in training data, which is again not the case for neural networks, decision trees learn very fast and neural networks learn relatively slow, etc. The main idea of this method was to combine decision trees and neural networks in order to combine their advantages. First is build a decision tree that is then used to initialize a neural network. Such a network is then trained using the same training objects.

The source decision tree is converted to a disjunctive normal form, which is a set of normalized rules. Then the disjunctive normal form serves as source for determining the neural network's topology and weights. The neural network has two hidden layers. The number of neurons on each hidden layer depends on rules in the disjunctive normal form. The number of neurons in the output layer depends on how many outcomes are possible in the training set. The conversion is described in the next steps (see Fig. 1):

1) Build a decision tree, using any available approach.
2) Every path from the root of the tree to every single leaf is presented as a rule.
3) The set of rules if transformed into the disjunctive normal form, which is minimal representation of original set of rules.
4) In the input layer create as many neurons as there are attributes in the training set.
5) For each literal in the disjunctive normal form there is a neuron created in the first hidden layer (literal layer) of a neural network.
6) Set weights for each neuron in the literal layer, that represents a literal in the form (attribute > value) to $w_0 = -\sigma * value$ for each literal in the form (attribute ≤ value) to $w_0 = \sigma * value$. Set all the remaining weights to $+\beta$ or $-\beta$ with equal probability. Constant σ is usually a number larger then 1 (usually 5) and constant β is a number close to 0 (usually 0.05).
7) For every conjunction of literals create a neuron in the second hidden layer (conjunctive layer).
8) Set weights that link each neuron in the conjunctive layer with the appropriate neuron in the literal layer to $w_0 = \sigma * (2n-1)/2$, where n is a number of literals in the conjunct. Set all the remaining weights to $+\beta$ or $-\beta$ with equal probability.
9) For every possible class create a neuron in the output layer (disjunctive layer).
10) Set weights that link each neuron in the disjunctive layer with the appropriate neuron in the conjunctive layer to $w_0 = -\sigma * (1/2)$ Set all remaining weights to $+\beta$ or $-\beta$ with equal probability.
11) Train the neural network using the same training objects as were used for training the decision tree.

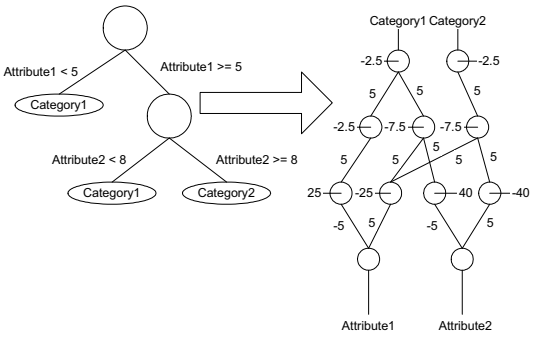

Figure 1. Transformation of decision tree to neural network

Such network is then trained using backpropagation. As it is seen in Fig. 2, mean square error of such network converges toward 0 much faster than it would in the case of randomly set weights in the network.

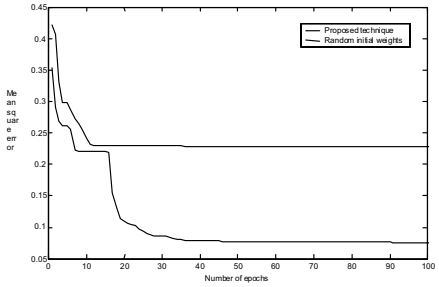

Figure 2. Mean square error convergence.

Even if we would use neural network before the backpropagation stage, it would already give good results.

IV. Experiment and Results

A. Feature Selection and Extraction

In text categorization, features are often measures of frequencies of words appearing in a document. Feature selection chooses which features to be used in classification. It is preferable to use less features than the raw measurements (say, frequency of each word), so that classification will be performed in a feature space of a lower dimensionality. By reducing the dimensions of the feature space, it not only increases the efficiency of the training and test processes, but also reduces the risk of overfitting the model to data. Feature extraction computes the chosen features from an input document. In statistical classification, features are represented in a numerical vector, which is subsequently used by the classifiers. Feature selection involves stop word removal, stemming, and term selection:

1) **Stop Word Removal.** Words used in text indexing and retrieval are called terms. According to the term discrimination model, moderate frequency terms discriminate the best. High frequency words, which are called *stop words,* have low information content, and therefore have weak discriminating power. They are removed according to a list of common stop words.
2) **Stemming.** Stemming reduces morphological variants to the root word. For example, "asks", "asked", and "asking" are all reduced to "ask" after stemming. This relates the same word in different morphological forms and reduces the number of distinctive words. The *Porter stemmer* is a commonly used stemmer [2].
3) **Term Selection.** Even after the removal of stop words and stemming, the number of distinct words in a document set may still be too large, and most of them appear only occasionally. In addition to removing high frequency words, the term discrimination model suggests that low frequency words are hard to learn about and therefore do not help much. They should be removed to reduce the dimensions of the vector space as well. We used information gain [4] to select the 1000 best features. This choice is made for compatibility with the Joachims [12], Yang and Liu [5] results.
4) **Feature Extraction.** After the terms are selected, for each document a feature vector is generated whose elements are the feature values of each term. We use $tf * idf$ (term frequency - inverse document frequency) weighting [13]: $tf_{ij} * \log_2 \frac{N}{n}$, where tf_{ij} - frequency of term T_j in Document D_i, N - number of documents in collection, n - number of documents where term T_j occurs at least once.

B. Classification

A number of classifiers have been tried on text categorization. In our experiment, we focused on the evaluation of the hybrid decision tree classifier on text categorization. We compare its accuracies to those of classical decision tree C4.5 [7] and feedforward backpropagation neural network.

C. Evaluation

The performance of category ranking can be evaluated in terms of *precision* and *recall,* computed at any threshold on the ranked list of categories of each document. The category

TABLE I
Contingency Table

	YES is correct	NO is correct
Assigned YES	a	b
Assigned NO	c	d

assignment of a binary classifier can be evaluated using a two-way contingency table (see Table I).

Precision is defined as a/(a+b), and recall is defined as a/(a+c). The point at which recall equals precision is the break-even point, which is often used as a single summarizing measure for comparing results. The F_1 measure, initially introduced by van Rijsbergen [10], combines recall (r) and precision (p) with an equal weight in the following form:

$$F_1(r, p) = \frac{2rp}{r + p} \quad (8)$$

Also measure alternative to precision and recall and commonly used in the machine learning literature, such as accuracy (a+d)/(a+b+c+d), is used in our text categorization evaluation. For evaluating performance average across categories, there are two conventional methods, namely macro-averaging and micro-averaging. Macro-averaged performance scores are computed by first computing the scores for the per-category contingency tables and then averaging these per-category scores to compute the global means. Micro-averaged performance scores are computed by first creating a global contingency table whose cell values are the sums of the corresponding cells in the per-category contingency tables, and then use this global contingency table to compute the micro-averaged performance scores. There is an important distinction between macro-averaging and micro-averaging. Micro-averaging performance scores give equal weight to every document, and is therefore considered a per-document average. Likewise, macro-average performance scores give equal weight to every category, regardless of its frequency, and is therefore a per-category average.

D. Text corpora

It is hard to find standard benchmark sets for text classification, where each method can be tested and its performance compared reliably with other methods. The Reuters sets are a notable exception. This collection consists a set of newswire stories classified under categories related to economics. Although different versions are available, many researchers use it for benchmarking. We will use the ApteMod version of Reuters-21578 (Yang and Liu, 1999). The ApteMod set has 7769 documents for training and 3019 for testing, after stemming and stop word removal 24240 unique terms remain. The Aptemod version has an average of 1.3 categories per document, with a total of 90 categories that occur in both sets.

E. Results

Precision, recall, F_1, accuracy on the ten most frequent Reuters categories and micro-averaged, macro-averaged measures over all 90 Reuters categories are given in tables II, III and IV. Precision, recall, F_1 and accuracy using proposed method in comparison with decision tree and neural network approaches are higher almost for all categories. The micro-averaged for precision, recall, breakeven point, F1, and accuracy using decision tree are 81.9%, 78.3%, 79.6%, 80.1%, 99.5% as shown in Table II. The micro-averaged for precision, recall, breakeven point, F_1, and accuracy using neural network are 83.9%, 83.4%, 83.7%, 84.7%, 99.5% as shown in Table III. In comparison with the adapted hybrid decision tree and neural network approach, the micro-averaged precision, recall, breakeven point, F_1, and accuracy are 90.9%, 85.2%, 87.0%, 86.3%, 99.6% as shown in Table IV. This indicates that hybrid approach increases text categorization performance. Also macro-averaged measures are better compared to the single decision tree or neural network text classifiers.

TABLE II
THE CLASSIFICATION RESULTS USING DECISION TREE APPROACH ON THE TEN MOST FREQUENT REUTERS CATEGORIES AND MICRO-AVERAGED, MACRO-AVERAGED PERFORMANCE OVER ALL 90 REUTERS CATEGORIES

Category	P (%)	R (%)	Br. p. (%)	F_1 (%)	Acc. (%)
Earn	98.6	95.3	96.94	96.9	97.8
Acq	94.0	92.0	93.05	93.0	96.7
Crude	85.0	68.8	76.88	76.0	97.3
Money-fx	58.5	78.8	68.64	67.1	95.4
Grain	72.7	73.2	72.91	72.9	97.3
Interest	82.0	73.3	77.67	77.4	98.2
Trade	63.4	66.7	65.04	65.0	97.2
Ship	81.6	69.7	75.62	75.2	98.6
Wheat	77.4	57.7	67.55	66.1	98.6
Corn	69.6	57.1	63.35	62.7	98.7
Micro-avg	81.9	78.3	79.6	80.1	99.5
Macro- avg	63.6	46.0	58.7	47.4	99.5

TABLE III
THE CLASSIFICATION RESULTS USING NEURAL NETWORK APPROACH ON THE TEN MOST FREQUENT REUTERS CATEGORIES AND MICRO-AVERAGED, MACRO-AVERAGED PERFORMANCE OVER ALL 90 REUTERS CATEGORIES

Category	P (%)	R (%)	Br. p. (%)	F_1 (%)	Acc. (%)
Earn	94.7	97.6	96.1	96.1	97.2
Acq	95.3	93.6	94.5	94.5	97.4
Crude	82.9	82.0	82.5	82.5	97.8
Money-fx	74.2	82.1	78.2	78.0	97.3
Grain	87.2	86.6	86.9	86.9	98.7
Interest	68.6	82.1	75.3	74.7	97.9
Trade	80.2	71.0	75.6	75.3	98.0
Ship	86.1	69.7	77.9	77.0	98.8
Wheat	77.5	77.5	77.5	77.5	98.9
Corn	70.4	89.3	79.9	78.7	99.1
Micro-avg	83.9	83.4	83.7	84.7	99.5
Macro- avg	65.0	63.9	66.3	59.8	99.6

TABLE IV
THE CLASSIFICATION RESULTS USING HYBRID APPROACH ON THE TEN MOST FREQUENT REUTERS CATEGORIES AND MICRO-AVERAGED, MACRO-AVERAGED PERFORMANCE OVER ALL 90 REUTERS CATEGORIES

Category	P (%)	R (%)	Br. p. (%)	F_1 (%)	Acc. (%)
Earn	99.5	97.9	97.2	97.2	98.0
Acq	98.4	94.6	96.5	96.5	98.3
Crude	86.2	82.5	84.4	84.3	98.1
Money-fx	74.4	83.2	73.8	73.8	96.9
Grain	97.7	86.9	91.8	91.4	99.2
Interest	87.6	82.3	76.3	74.6	98.1
Trade	81.0	72.1	76.5	76.7	98.9
Ship	94.0	70.8	82.4	80.8	99.0
Wheat	89.1	80.3	84.7	84.4	99.3
Corn	90.6	85.7	88.1	88.1	99.6
Micro-avg	90.9	85.2	87.0	86.3	99.6
Macro- avg	79.2	69.6	73.4	60.5	99.6

The micro-averaged breakeven point (79.6%) of decision tree is slightly higher than reported by Joachims (79.4%) [12]. The micro-averaged breakeven point (83.7%) of neural network is slightly lower than reported by Yang and Liu

(83.8%) [5]. These disagreements are possibly because of a difference in term weighting scheme for document presentation.

V. Conclusion

An adaptation of the hybrid machine learning algorithm is proposed in which a decision tree from root node until a final leave is used for initialization of multilayer neural network. Our study provides evidence that hybrid machine learning approach can be used for the construction of effective classifiers for automatic text categorization. We have presented a hybrid decision tree and neural network algorithm for building the classifier.

This paper showed that hybrid decision tree and neural network approach improved accuracy in text classification task and are substantially better than single decision tree or neural network text classifiers performance comparable to previous researches results.

Although encouraging results have been obtained using hybrid approach based classifier, there is still much work remaining to be investigated. They include to create new decision tree construction algorithm for textual data and to determine how much it has an effect on hybrid classifier accuracy. These issues are left for our future works.

References

[1] Banerji, A. (1997). Initializing neural networks using decision trees. Computational Learning Theory and Natural Learning Systems, MIT Press, IV, 3-15.
[2] Frakes, W., and R. Baeza-Yates (1992). Information Retrieval: Data Structures & Algorithms, Prentice Hall.
[3] Haykin, S. (1994). Neural Networks: A comprehensive foundation. Macmillan College Publishing Comp., New York.
[4] Yang, Y., and J. Pedersen (1997). A comparative study on feature selection in text categorization. In Proceedings of ICML-97, 14th International Conference on Machine Learning, Nashville, US, 412–420.
[5] Yang,Y., and X. Liu (1999). A re-examination of text categorization methods. In Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval, Berkeley, US, 42–49.
[6] Lewis, D.D., and M. Ringuette (1994). A comparison of two learning algorithms for text categorization. In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, 81–93.
[7] Quinlan, J.R. (1993). C4-5: Programs for machine learning, Morgan Kaufmann, San Mateo, CA.
[8] Rumelhart, D.E., and J.L. Mcclelland (1986). Parallel distributed processing 1. MIT Press, Cambridge, MA.
[9] Sebastiani, F. (2002). Machine learning in automated text categorization. ACM Computing Surveys, 34(1), 1-47.
[10] van Rijsbergen, C.J. (1979). Information Retrieval. Butterworths, London.
[11] Wiener, E.D., J.O. Pedersen, and A. S. Weigend (1995). A neural network approach to topic spotting. In Proceedings of SDAIR-95, 4th Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, 317–332.
[12] Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. In Proceedings of ECML-98,10th European Conference on Machine Learning, 137–142.
[13] Dumais, S. T. (1991). Improving the retrieval information from external soures. Behaviour Research Methods, Instruments and Computers, 23(2), 229–236.