ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, Issue 2, February 2013

Web Text Classification Using Genetic Algorithm and a Dynamic Neural Network Model

Revathi N[1], Anjana Peter[2] & Prof.S.J.K.Jagadeesh Kumar[3]
1 & 2. PG Scholar, Dept. of Computer Science & Engg., SKCT, Coimbatore.
3. Head of Department, CSE, SKCT, Coimbatore.

ABSTRACT:

Widespread adoption of the Internet, popularity of social networking, and the digitalization of information within organizations have intensified the need for effective textual document classification algorithms. Most real life classification problems, including text classification is complex and high dimensional in nature. A web text classification method using a dynamic artificial neural network is presented in this paper. The proposed method can classify a set of English text documents into a number of given classes depending on their contents. Text documents, internet edition of news paper, are considered for classification. This paper uses genetic algorithm(GA) for feature engineering, in which most relevant features are extracted and classifies the documents using dynamic neural network, which is an effective and scalable method for text classification.

Index Terms: **Text classification, Genetic Algorithm, Dynamic Neural Network, Machine Learning, Pattern Recognition.**

INTRODUCTION:

Text categorization research has its root in the 1960s. Until the late 1980s, text categorization systems, based on knowledge-engineering, were built manually. Digitization of information in today's internet age has led to large collection of digital documents. Automatic categorization and labeling are essential in processing digital information for all applications. For textual documents, classification is an important technique often utilized by researchers and practitioners. By text classification we mean both the automated assignment of textual data to groups or classes (often referred to as categorization), as well as the use of automated techniques for discovering such classes (often referred to as clustering). An example of this approach is the categorization of news stories into a broad set of topical categories using pattern-matching techniques. This approach has been superseded by the machine learning approach in which the text classifier is automatically built by learning from an already classified text collection (the training set), as identified by domain experts. This approach is shown to be comparable with human performance while offering significant savings in costs and manpower.

In machine learning methods, unprocessed texts are first preprocessed in order to create a representation of the text in vector space. Once the document is fully preprocessed, feature engineering techniques are applied. The large number of possible features in text classification presents a problem for many classification algorithms. Feature selection is an important step in any method of text classification. In this work, an efficient method of feature selection is

used. It uses genetic algorithm for feature engineering which selects the most optimal features to include only those words in the given set of text documents that are useful for the purpose of classification. The documents are divided into training and testing datasets and the process of classifying the documents begin by feeding this into a dynamic neural network model.

The benchmark used in this research is Reuters dataset. The most prominent methods today struggle in achieving high levels of classification accuracy. In this work, the use of genetic algorithm for feature extraction and dynamic neural network for classification has paved the way for better performance.

RELATED WORK:

Web text classification problems are characterized by high dimensionality and are highly complex in nature. In order to reduce complexity, some preprocessing techniques are applied to the original text to make it into a more simplified form. Various term weighting methods such as binary weighting, the term frequency approach (TF), and the term-frequency/inverse document frequency (TF/IDF) can be used for defining a transformed document's value (Manning et al., 2008; Salton & Buckley, 1988). Software tools can be used for performing these steps. Tools from the R((Feinerer, 2009;Venables & Smith, 2009) and Weka (Hornik, Zeileis, Hothorn, & Buchta, 2006) systems are used to perform these activities. Even after preprocessing the large documents, the size of the resulting matrix is still very large. In order to reduce the size feature engineering methods are applied. In text classification models, a feature represents a variable. Most real life text classification problems, including the Reuters benchmark used in this study, often involve 1000s or 10,000s of features. Feature engineering is applied in order to reduce the number of features. This is a common step for classification methods in order to reduce complexity (Yang & Pedersen, 1997). Additionally, feature engineering serves as a means to remove noise from the input documents (Manning et al., 2008). The most widely used feature engineering approaches are the information gain (IG) (Joachims, 1998; Lewis & Ringuette, 1994; Mitchell, 1996; Moulinier, Raskinis, & Ganascia, 1996; Quinlan, 1986), document frequency thresholding (DF) (Yang& Pedersen, 1997), mutual information (MI) (Manning et al.,2008), term strength (TS) (Yang & Pedersen, 1997), latent semantic indexing (LSI) (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990), and the chi-square statistic (Yang & Pedersen, 1997; Ghiassi , Olschimke, Moon, Arnaudo,2012).

The large number of possible features in text classification presents a problem for many classification algorithms. For instance the Bayesian approach will become computationally intractable for large applications (Yang & Pedersen,1997). Wiener, Pedersen, and Weigend (1995) report that their neural network based approach yielded the best average classification performance using about 20 features on the Reuters-22173 dataset, which is comparable to other research results (Apte, Damerau, & Weiss, 1994; Lewis, 1992a; Lewis & Ringuette, 1994). They also report that when using more than 20 features, the classification performance on the test set quickly decreases. The authors concluded that the neural network model[3] used in their research was not able to extract features to generalize well on the test dataset. Neural networks have often been employed in solving the problems of text document classification. M. Ruiz et.al.has suggested hierarchical neural networks for text categorization. Besides the later method being hierarchical in nature, it requires the number of classes present in the set of text documents to be known a priori. Text

classification methods using LVQ network also require the number of classes to be supplied. A web news classification technique using neural networks is proposed by A. Selamat et.al. This method is based on principal component analysis and it requires the class profile-based features to be known a priori.

PROPOSED METHODOLOGY:

Motivated by the aforementioned issues, a new system is proposed which makes use of an artificial intelligence based approach for text classification. In this work Reuters dataset is taken for experimentation. The large scale text documents are preprocessed. Even if it has undergone preprocessing still it has a large matrix. Feature engineering method is applied to reduce the number of features. Genetic Algorithm is applied to extract the most relevant features for classification. The documents are divided into training and testing datasets and the process of classifying the documents can begin. A dynamic architecture for Artificial Neural Networks, which employs a different architecture than the traditional neural network (FFBP) model is used for text classification. Pattern recognition strategy is employed in this machine learning method. The performance of the classifier is evaluated. The easiest approach to evaluate the performance of a classifier is to compare the number of right decisions (true positives and true negatives) with the number of wrong decisions (false positives and false negatives). This is defined as the error rate (a measure of accuracy).

1. Document Pre-processing:

Web text classification problems are complex in nature. Preprocessing techniques are applied to reduce the complexity of text documents. Preprocessing include some basic and standard steps. The basic steps include: (1) removing white spaces, (2) changing all words to lower case,(3) transforming the original, often XML-based files into plain text and (4) removing punctuation and numbers. The standard preprocessing steps are performed after the completion of basic preprocessing techniques. The standard steps include (1) removing stop words, (2) stemming, (3) transforming the data into the vector space, (4) term weighting. These preprocessing steps transform the documents into a simpler form and a vector representation of the documents is produced. This vector is called the bag-of-words. Suffix stripping algorithm[4] is used for preprocessing the documents. The transformed document's value for each term is defined using various term weighting methods. Common term weighting[12] methods are binary weighting, the term frequency approach (TF), and the term-frequency/inverse document frequency (TF/IDF) method[5]. Automated software tools are used to perform these activities.

2. Feature Engineering using GA:

Once the document is fully pre-processed, feature engineering techniques are applied. Most real life text classification problems, including the Reuters benchmark[7] used in this study, often involve 1000s or 10,000s of features. Feature engineering is applied in order to reduce the number of features. This is a common step for classification methods in order to reduce complexity. Additionally, feature engineering serves as a means to remove noise from the input documents.

Feature Extraction is the process of detecting and eliminating irrelevant, weakly relevant or redundant attributes or dimensions in a given dataset. The goal of feature selection is to find the minimal subset of attributes such that the resulting probability distribution of data classes is close to original distribution obtained using all attributes. In general, the computational cost of data set D is O(n x |D| x log(|D|)), where n – number of attributes, D – number of instances. The number of comparisons required for *m* attributes and *n*

instances is $m * n^2$. For a data set D, with n attributes, 2^n subsets are possible. Search for an optimal subset would be highly expensive especially when n and the number of data classes increases. Sometimes it may be infeasible.

Genetic algorithms are usually applied to spaces which are too large to be exhaustively searched[9]. A genetic algorithm is an iterative procedure that involves a population of individuals, each one represented by a finite string of symbols, known as the genome, encoding a possible solution in a given problem space. This space, referred to as the search space, comprises all possible solutions to the problem at hand. Stochastic optimization methods such as genetic algorithms perform global search and are capable of effectively exploring large search spaces[14].

3. Predictive Modeling using a dynamic artificial neural network model:

The general philosophy of the dynamic neural network model is based upon the principle of learning and accumulating knowledge at each layer, propagating and adjusting this knowledge forward to the next layer, and repeating these steps until the desired network performance criteria are reached. As in classical neural networks, the dynamic neural network architecture is composed of an input layer, hidden layers and an output layer. The input layer accepts external data to the model. In dynamic neural network, unlike classical neural networks, the number of hidden layers is not fixed a priori. They are sequentially and dynamically generated until a level of performance accuracy is reached. Additionally, the proposed approach uses a fixed number of hidden nodes (four) in each hidden layer. This structure is not arbitrary, but justified by the estimation approach. At each hidden layer, the network is trained using all observations in the training set simultaneously, so as to minimize a stated training accuracy measure such as mean squared error (MSE) value or other accuracy measures. Fig. 1 presents the overall architecture of dynamic artificial neural network model. In fig. 1, each hidden layer is composed of four nodes. The first node is the bias or constant input node, referred to as the C node. The second node is a function that encapsulates the ‘‘Current Accumulated Knowledge Element’’ (CAKE node) during the previous training step. The third and fourth nodes represent the current residual (remaining) nonlinear component of the process via a transfer function of a weighted and normalized sum of the input variables. Such nodes represent the ‘‘Current Residual Nonlinear Element’’ (CURNOLE nodes). In Fig. 1, the ‘‘I’’ node represents the input, the ‘‘C’’ nodes are the constant nodes, the ‘‘G_k’’ and ‘‘H_k’’ nodes represent CURNOLE nodes, and the ‘‘F_k’’ nodes are the CAKE nodes. The final CAKE node represents the dependent variable or the output. At each layer, the previous four nodes (C, G_k, H_k, and F_{k-1}) are used as the input to produce the next output value (F_k). The parameters on the arcs leading to the output nodes, (a_k, b_k, c_k, d_k), represent the weights of each input in the computation of the output for the next layer. The parameter connecting the CURNOLE nodes, μ_k, is used as part of the argument for the CURNOLE nodes and reflects the relative contribution of each input vector to the final output values at each layer. The training process begins with a special layer where the CAKE node captures the linear component of the input data. Thus, its input (content) is a linear combination (weighted sum) of the input variables and a constant input node. These weights are easily obtainable through classical linear regression. If the desired level of accuracy is reached, we can conclude that the relationship is linear and the training process stops. This step is used as the starting point.

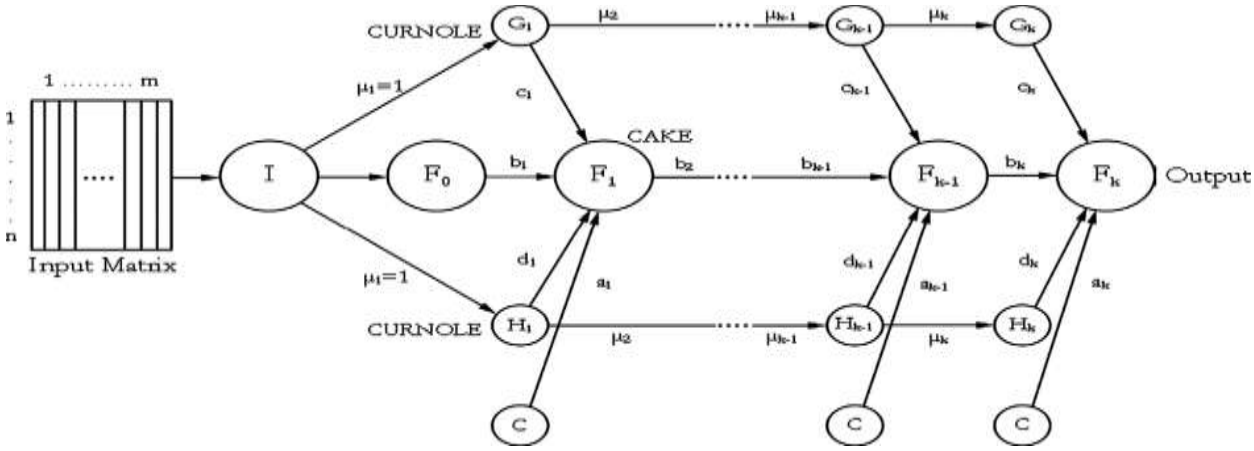

Fig. 1: Architecture of dynamic artificial neural network model.

For nonlinear relations additional hidden layers are required. At each subsequent layer the input to the CAKE node is a weighted sum (linear combination) of the previous layer's CAKE, CURNOLE, and C nodes.

Throughout training, the CAKE nodes carry an adequate portion of learning achieved in previous layers forward. This process ensures that the performance or knowledge gained so far is adjusted and improved but not lost. This property of dynamic neural network introduces knowledge memorization to the model. This algorithm ensures that during network training, the residual error is reduced in every iteration and the accumulated knowledge is monotonically increased. The training process defines creation of partitions among classes that could include linear and nonlinear components. The linear component of the input data is captured in the first CAKE node using ordinary least squares (OLS) or other simple and easy to compute approaches. The algorithm next transforms the input dataset to model the nonlinearity of the process in subsequent iterations. DAN2 uses a vector projection approach to perform data transformation. The transformation process defines a reference vector R = $\{r_j; j = 1, 2, \ldots, m\}$, where m represents the number of attributes of the observation records, and projects each observation record onto this vector to normalize the data as discussed in Ghiassi and Saidane. This normalization defines an angle, a_i, between record i and the reference vector R. DAN2 uses the set of a_i's to train the network, and updates their values in every iteration. In Ghiassi and Saidane [2] the authors show that this normalization can be represented by the trigonometric function Cos ($\mu_k \alpha_i + \theta_k$). In every hidden layer k of the architecture we vary ($\mu_k \alpha_i + \theta_k$) and measure the impact of this change on the output value. The modification of the angle ($\mu_k \alpha_i + \theta_k$) is equivalent to rotating μ_k and shifting θ_k, the reference vector, thus changing the impact of the projected input vectors and their contribution to the output for that iteration. The Cos($\mu_k \alpha_i + \theta_k$) uses two (nonlinear) parameters, μ_k and θ_k. The use of the latter can be avoided through the expansion of the cosine function in the form: A Cos($\mu_k \alpha_i$) + B Sin ($\mu_k \alpha_i$). We use this functional form as the transfer function in our model. The two CURNOLE nodes in Fig. 1 represent this formulation. At any given hidden layer k, if the Cos($\mu_k \alpha_i + \theta_k$) terms captured in previous layers do not adequately express the nonlinear behavior of the

process, a new layer with an additional set of nodes is automatically generated, including a new $Cos(\mu_k\alpha_i + \theta_k)$ term. This process is analogous to how the Fourier series adds new terms to improve function approximation. Therefore, the number of layers in the DAN2 architecture is dynamically defined and depends upon the complexity of the underlying process and the desired level of accuracy. Thus, the output of this model is represented by the linear combination of the constant, CAKE, and CURNOLE nodes.

3.1 The dynamic learning algorithm:

Define the input matrix $X=\{X_i; i=1,2,\ldots n\}$ as n independent records of m attributes[1]. Let $X_i =\{X_{ij} ; j=1, 2, . . . ,m\}$. Also define a reference vector $R=\{r_j; j=1,2,\ldots,m\}$ as a vector of constant values.

The algorithm steps are as follows:

1. The initial linear layer:

$$F_0(X)=a_0+\sum b_{0j}x_{ij}. \quad \text{----------} \quad (1)$$

2. Subsequent hidden layers' CAKE node (at iteration k):

$$F_k(X_i)=a_k+b_kF_{k-1}(X_i)+c_kG_k(X_i)+d_kH_k(X_i). \quad \text{----------} \quad (2)$$

3. The CURNOLE node's input and transfer function at iteration k (k=1, 2, . . ,K; where K is the maximum sequential iterations or number of hidden layers) is defined as:

a. Specify a random set of m constants representing the reference vector R(default $r_j=1$ for all j=1; 2; . . . ;m).

b. For each input record X_i, compute the scalar product: $(R*X)=\sum_j r_jx_{ij}$.

c. Compute the length (norm) of the vector R and a record vector

X_i : $||R||= sqrt(\sum_j r_j^2)$; $||X_i||= sqrt(\sum_j x_{ij}^2)$.

d. Normalize (R*X) to compute $(R*X)_N$: $(R*X_i)_N =(R*X_i)/||R||*||X||$. Recall that $(R*X_i)=(||R||*||X_i||)*cos\{angle(R,X_i)\}$ thus $cos\{angle(R,X_i)\}=(R*X_i)/||R||*||X||_i)= (R*X_i)_N$.

e. Compute angle(R,X_i): $angle(R,X_i)= Arc\{Cos(R*X_i)_N\}=\alpha_i$ for i=1, 2, . . . , n.

f. Compute the transferred nonlinear component of the signal as:

$G_k(X_i)=cos(\mu_k * \alpha_i)$. $H_k(X_i)=sin(\mu_k * \alpha_i)$, where μ_k is a constant multiplier for iteration k.

g. Replacing $G_k(X_i)$ and $H_k(X_i)$ in Eq. (2) will result in

$$F_k(X_i)=a_k+b_kF_{k-1}(X_i)+c_k cos(\mu_k* \alpha_i)+d_k sin(\mu_k * \alpha_i).$$

RESULT AND DISCUSSION:

The significance of this methodology is in classifying the web text into various categories. This classification helps in processing the digital documents which are used for various applications. The preprocessing method utilizes a suffix stripping algorithm which is much useful for improved performance. The application of feature selection method using GA can reduce the number of attributes by 75%, which contributes much for classification accuracy. The dynamic neural network model for prediction has better performance. Throughout training, the CAKE nodes carry an adequate portion of learning achieved in previous layers forward. This process ensures that the performance or knowledge gained so far is adjusted and improved but not lost. This property of dynamic neural network introduces knowledge memorization to the model. The algorithm ensures that during network training, the residual error is reduced in every iteration and the

accumulated knowledge is monotonically increased, thereby increasing accuracy rate.

CONCLUSION:

This paper presented a genetic algorithm and dynamic neural network based approach for web text classification. The introduction of GA in this system has helped in retrieving the most optimal features. The dynamic neural network is scalable and can be used for document text classification without requiring experimentation with parameter settings or network architectural configurations. Overall, results show that DAN2 performs better than both kNN and SVM, for Reuters-21578 dataset. The excellent uniform performance of dynamic neural network indicates the robustness of the approach and its generality. Finally, the transformation and dimensionality reduction property of dynamic neural network makes its application to text classification especially important.

REFERENCES:

[1] Ghiassi, M., & Saidane, H. (2005), " A dynamic architecture for artificial neural Network", Neurocomputing, 63, 397–413.

[2] M. Ghiassi , M. Olschimke, B. Moon & P. Arnaudo. "Automated text classification using a dynamic artificial neural network model", Expert Systems with Applications 39 (2012) 10967–10976.

[3] D. Saha, "Web Text Classification Using a Neural Network", 2011 Second International Conference on Emerging Applications of Information Technology.

[4] M.F.Porter, "An algorithm for suffix stripping",pp.313-316.

[5] Feinerer, I., "A framework for text mining applications within R". http://cran.rproject.org/web/ packages/tm/index. html. Accessed 7 June 2009.

[6] Lewis, D. D., "Text representation for intelligent text retrieval: A classification-oriented view", In Proceedings of the 15th annual international SIGIR '92 (pp. 37–50), Denmark.

[7] Lewis, D. D., " Reuters-21578 text categorization test collection", http:// kdd.ics.uci.edu/ databases/reuters21578/ README.txt. Accessed 7 June 2009.

[8] Yang, Y.,& Wilbur, J., "Using corpus statistics to remove redundant words in text categorization. Journal of the American Society for Information Science, 57(5), 357–369.

[9] Tom V. Mathew, "Genetic Algorithm".

[10] Mitchell, T. (1996). Machine learning. Columbus, OH: McGraw Hill.

[11] Moulinier, I., Raskinis, G., & Ganascia, J., "Text categorization: a symbolic approach". In Proceedings of SDAIR-96, 5th annual symposium on document analysis and information retrieval (Las Vegas, NV).

[12] Salton, G., & Buckley, C. (1988). Term weighting approaches in automatic text retrieval. Information Processing & Management, 24(5), 513–523.

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

[14] D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning. Reading, MA: Addison-Wesley, 1989.