A Novel System for Document Classification Using Genetic Programming

Saad M. Darwish, Adel A. EL-Zoghabi, and Doaa B. Ebaid
Institute of Graduate Studies and Researches, Alexandria University, Egypt
Email: Saad.darwish@alex-igsr.edu.eg, zoghabi@gmail.com, doaa_ebaid@yahoo.com

***Abstract*—With the increasing availability of electronic documents and the rapid growth of the World Wide Web, the task of automatic categorization of documents became the key method for organizing the information and knowledge discovery. Document retrieval, categorization, routing and filtering can all be formulated as classification problems. The complexity of natural languages and the extremely high dimensionality of the feature space of documents have made this classification problem very difficult. The proposed work mitigates this difficult by providing an algorithm to classify documents into more than two categories (multi-class classification) at the same time by combining multi-objective technique with the genetic programming of classifiers based on multi-tree representation of documents. This combination has the potential to attain lower errors because classification accuracy on each class is represented as a distinct objective. Empirical evaluations show encouraging results and confirm that the proposed algorithm is feasible and effective.**

***Index Terms*—document classification, genetic programming, multi-objective techniques, multi-tree representation**

I. Introduction

According to the growth in the amount of text documents over the internet and news sources which make document classification is an important task in document processing. Document classification was widely used in many contexts like document indexing, document analysis, document filtering, automatic distribution or archiving of documents [1] [2]. This process difficult to be manual with huge number of documents so automatic classification is better than manual classification because it has more accuracy and time efficiency [3] [4]. Natural language processing, data mining, and machine learning techniques work together to automatically classify documents.

Automatic text classification is the activity of assigning pre-defined category labels to natural language texts based on information found in a training set of labeled documents. A definition of a document is that it is made of a joint membership of terms which have various patterns of occurrence. Informal, document classification is to assign a document d_j to a category c_i by approximating a function $\phi: D \times C \rightarrow \{T, F\}$ by maximizing the coincidence of a function ϕ with the actual categorization Φ, where $D=\{d_1,d_2,\ldots,d_n\}$is the set of documents, n refers to the total number of documents, $C= \{c_1,c_2,\ldots,c_m\}$is the set of classes, m refers to the total number of predefined classes, T and F are boolean values for true and false respectively [5] [6] [7].

Text classification presents many challenges and difficulties. Some of them are [6]: achieving high accuracy in all application contexts, dealing with very large numbers of categories, appropriate document representation, dimensionality reduction to handle algorithmic issues, and an appropriate classifier function to obtain a good generalization and avoid over-fitting. Also, it is difficult to capture high-level semantics and abstract concepts of natural languages just from a few key words. Thus to tackle these problems a number of methods have been reported in the literature for effective text document classification.

Automatic document classification divides into two types of approaches [1] [6] [8]: Rule based and Machine learning. Rule based approach (knowledge engineering approach) where a set of rules was defined manually that convert expert knowledge on how to classify documents under the given set of categories. The advantages of this approach are human understandable, providing accurate rules and easy to control when the number of rules is small. The drawback was the high cost of human power required for defining the rules set and maintaining it.

Machine learning algorithms automatically builds a classifier by learning the characteristics of the categories from a set of classified documents, and then uses the classifier to classify documents into predefined categories. In this case, a considerable savings in terms of expert human power and accuracy is comparable to that achieved by rule based approach. For these reasons, machine learning based approach is replacing rule based approach for document classification. However, these machine learning methods have some drawbacks. In order to train classifier, human must collect large number of training text terms, the process is very laborious. If the predefined categories changed, these methods must collect a new set of training text terms, so it is difficult to improve the accuracy of these classification methods [1] [6].

The pre-defined documents are the key resource in machine learning approach. If they are not good enough to extract rules or equation correctly, the result will not

Manuscript received March 1, 2015; revised August 1, 2015.

doi: 10.12720/jait.6.4.194-200

be satisfied. According to the variety of the datasets which consist of hundreds of thousands of documents characterized by a huge number of terms, both efficiency and accuracy depend on classification systems [4]. There are many machine learning techniques which are used for document classification[4] [9] [10] [11] such as Bayesian classifier, decision tree, K-nearest neighbor, support vector machines, neural networks, genetic algorithm, and genetic programming (GP), etc. Although, this number of machine learning techniques is used in the document classification process, the process needs to improve the performance because the most of these techniques suffer from the computation time and accuracy. Besides, combinations of multiple classifiers did not always improve the classification accuracy compared to the best individual classifier [3].

GP is a supervised machine learning technique and a powerful evolutionary algorithm widely used to evolve computer programs automatically [11]. The principle components of the GP are a set of functions and terminals that are able to represent the solution of the problem. GP works by iteratively applying genetic operators, such as crossover and mutation, to the individual programs of the population using a fitness measure. At the end of the GP run, the best individual is presented as the solution to the problem [8] [11] [12]. GP has ability to discover the underlying data relationships and express them mathematically [13] but GP classifiers still suffer from several difficulties, ineffective crossover and mutation, training time is long and error rates greater than those achieved by other classification methods on some tasks.

So to evolve GP classifiers, multi-objective techniques are used [14]. Evolutionary multi-objective (EMO) techniques have been applied to genetic programming for parsimony enforcement in order to overcome 'bloat', the tendency of genetic programs to develop unnecessary code or 'introns', and evolve programs for less computational cost. Elitist Non-dominated Sorting Genetic Algorithm (NSGA-II) is one of these techniques which evolve classifiers [15]. It has a better sorting algorithm, incorporates elitism and no sharing parameter needs to be chosen a priori.

Multi-category classification means a classification task with more than two classes. One of the most popular techniques used to deal with multi-class classification is dividing the multi-class classification into multiple two-class (binary) classification. The most common ways to decompose multi-class classification to binary classification are one-vs.-all (OVA) and one-vs.-one (OVO). In OVA approach, by assuming the number of classes is *m*, *m* binary classifiers are required to discriminate between one of the classes and all other classes. The learning phase is done using all training data, so this approach needs high memory requirement. In OVO approach, *m* (*m*-1)/2 binary classifiers are required to discriminate between different pairs of classes; this means the number of classifiers increase by increasing number of classes. The learning phase is done by using a subset from the original training data; the subset training data contain samples that belong to the two corresponding class labels [16][17] .

The main contribution of this research study is to propose a bio-inspired document classification system for that employs multi-objective GP with multi-tree representation [18]. The motivation behind multi-tree representation concept is to well balance the exploration and scalability (number of classes) capability of GP classifier for attaining better accuracy where each tree represents a classifier for a particular class. By using multi-tree representation, GP can classify all documents from several classes at the same time and in one single run. Also, the model combines GP with NSGA-II to achieve more accuracy, since using GP only do not give the desired results.

The paper is organized as follows: Section II briefly discuses some of the research related to document classification. A brief overview on the proposed document classification system is given in section III. The experimental result and evaluation of the proposed system are reported in section IV. Finally in section V, the conclusion and future work are presented.

II. Literature Review

In this section, we will cover the use of GP as a classifier in document classification in the previous works. For a comprehensive overview of document classification requirements, we refer to [2][9]. Several works have developed text classification methods using evolutionary algorithms, however, no studies can be found that uses evolutionary multi-objective for multi-class classification approaches.

The authors in [19] used an approach used to segment image document and classify the document regions as text, image, drawings and table. Document image is divided into blocks using run length smearing rule and features are extracted from every blocks. Discipulus tool has been used to construct the genetic programming based classifier model.

Using GP to classify unstructured documents is started in [20]. The aim of this model is to show that GP can be used to classify a document based on the set of words which presented in it. The agent is equipped with a dictionary of words. This dictionary contains all words used in any of the documents processed by the agent, and assigns to each word a number; this number used to refer to that word. The fitness function for the GP solutions is the number of documents which classified incorrectly. This model was created as a two class problem (interesting or uninteresting). The result showed that GP is a useful method for classification in spite of the model was needed to a large improvement.

In [8], the authors employed genetic programming to create compact classification rules using combinations of *N*-Grams (sequences of *N* letters). This system incorporated the advantageous of machine learning and rule based approaches to produce a combination of both of them. Set of features of the system includes word combinations and negative information (the features which are not found in a specific document) for discrimination purposes. The fitness of the GP individuals starts by evaluating the rule produced by the

GP against all documents in the training set, and then calculates F1-measure. The system deals with a large number of features without take in consideration the frequency of the *N*-Gram pattern. Furthermore, it is very difficult to decide the number of grams to be considered for effective document representation. Here, the GP process didn't depend on elitism which means the best individuals don't keep to the next generation, and the researchers suggested combining this model with any other classifier (e.g. in [5]) to get a practical value. In general, combinations of multiple classifiers did not always improve the classification accuracy compared to the best individual classifier.

Previous researches reveal that all of the benefits of GP are not attained as expected in document classification problem as there are many challenges that work as barriers in success of the of GP and make it more difficult to achieve expected benefits. The previous works deal with multi-class document classification as many binary classification problems. Aiming to fill this gap in literature, this paper proposes a multi-objective GP algorithm for multi-class unstructured document classification.

III. Proposed Document Classification System

The idea developed in this paper is partially inspired by existing work in [18] that exploits GP with multi-tree representation concept to deal with all classes at the same time instead of converting multi-class classification into multi-two class classifications. To avoid large tree size of the classifier, the proposed system adds NSGA-II multi objective technique in the learning phase to improve the accuracy and reduce learning time. Combining multi-objective techniques with the genetic programming of classifiers presents opportunities for evolving smaller classifiers that should generalize better while searching the solution space more effectively. This would lead to classifiers that attain lower errors and are evolved for less computational cost than would be the case using the standard GP algorithm. Fig. 1 shows the main system components and how they are linked to each other.

The system deals with unstructured documents with no layout characteristics and uses a predefined set of documents in training phase (supervised learning technique), and the results are then tested on the testing data. The learning phase is done using all training data. Our multi-tree classifier that contains *m* trees is evolved in the same time and in a single run of GP, where *m* is the number of classes. Each model's stage is explained in the following subsections.

A. Pre-Processing

The first step is to transform documents into clear word format. The documents are represented by a great amount of features (terms, words). Each document is seen as a set of words with no mutual relations between the words. This set of words is used to classify document. Before starting in pre-processing step, all words are converted to lower case. The preprocessing steps are [21]:

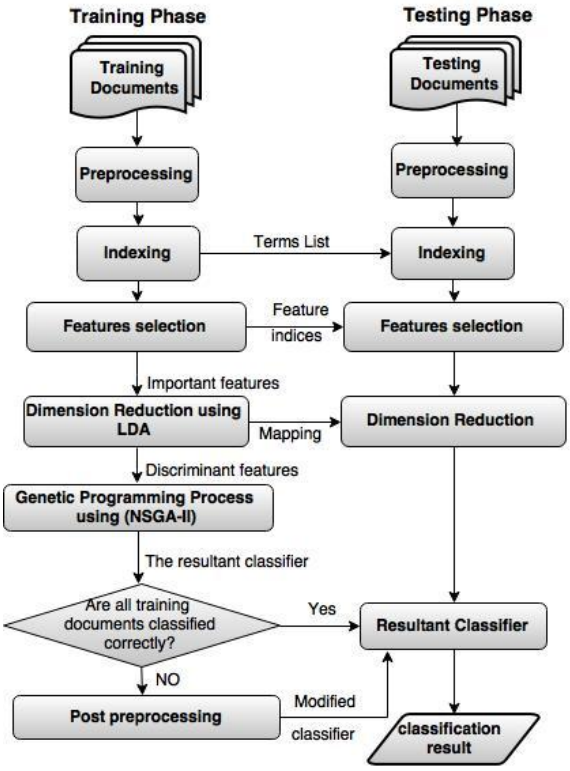

Figure 1. The proposed document classification model.

- Tokenization: a document is treated as a string, and then partitioned into a list of tokens (words).
- Removing of stop-words: words such as 'a', 'an', 'the', etc that occur commonly in all documents are removed.
- Stemming: the root of the words is transformed into an original form, e.g. connection to connect, computing to compute. There are many rules that control the stemming step as illustrated in [22].

At the end of the preprocessing step, the terms are prepared and saved in terms list. This list is used in the next step to create term frequency matrix for documents.

B. Indexing

A text document is typically represented as a vector of term weights (word features) from a set of terms (dictionary), where each term occurs at least once in a certain minimum number of document. In our case, we ignore the structure of a document and the order of words in the document. The feature vectors represent the words observed in the documents. The choice of document representation has a profound impact on the quality of the classifier. So, we use term frequency– inverse document frequency (TF-IDF) weighting method for document representation [1]. TF-IDF approach is commonly used to weight each word in the text document according to how unique it is. In other words, the TF-IDF approach captures the relevancy among words, text documents and a particular category to select informative features. TF-IDF is defined as [23]:

$$w_{ij} = tf_{ij} \times log\left(\frac{n}{1+df_{ij}}\right) \quad (1)$$

where w_{ij} is the weight of word in the document d_j, tf_{ij} codes term frequency that measures how frequently a term t_i occurs in a document d_j, idf_{ij} is inverse document frequency that measures how important a term is, n characterizes the total number of documents and df_{ij} is the document frequency that measures the number of documents in which t_i occurs. As a result of this step, the following document-term matrix D is appeared using TF-IDF weighting method:

$$D = \begin{pmatrix} w_{11} & w_{21} & \cdots & w_{|T|1} \\ w_{12} & w_{22} & \cdots & w_{|T|2} \\ \vdots & \vdots & \vdots & \vdots \\ w_{1n} & w_{2n} & \cdots & w_{|T|n} \end{pmatrix} \quad (2)$$

where $|T|$ denotes the length of terms vector. Each column represents a specific word and each row represents specific document [21]. After applying this stage on the training documents, the terms list passes to the test phase to build the document-term matrix for the test documents.

C. Features Selection

Document classification is often characterized by the high dimensionality feature space and a relatively small number of training samples. The number of potential features often exceeds the number of training documents. Feature selection is used to select a subset of features from the original documents. FS is performed by keeping the words with highest score according to the pre-determined measure of the importance of the word [21]. Information gain (*IG*) method is one of feature selection methods that measures the amount of information obtained for category prediction by knowing the presence or absence of a term in a document. The information gain of term t_j , j=1,2,…$|T|$ is defined as [24]:

$$\begin{aligned} IG(t_j) = -\sum_{i=1}^{m} P(c_i) \log P(c_i) + P(t_j) \sum_{i=1}^{m} P(c_i \mid t_j) \log P(c_i \mid t_j), \\ + P(\overline{t_j}) \sum_{i=1}^{m} P(c_i \mid \overline{t_j}) \log P(c_i \mid \overline{t_j}), \end{aligned} \quad (3)$$

where c_i represents the i^{th} category, $P(c_i)$ is the probability of the i^{th} category, $P(t_j)$ and $P(\overline{t_j})$ are the probabilities that the term t_j appears or not in the documents, respectively, $P(c_i \mid t_j)$ is the conditional probability of the i^{th} category given that term t_j appeared, and $P(c_i \mid \overline{t_j})$ is the conditional probability of the i^{th} category given that term t_j does not appeared [23]. After finishing this loop, each term in the terms set has its information gain value .Terms whose information gain is less than some predetermined threshold are removed from the feature space [24]. In other words, the document-term matrix will be sparse after this step by removing the terms which have information gain value less than the predetermined threshold. We assume that D converts to X. After applying this stage, the selected feature indices pass to test phase to remove unwanted features from the test document-term matrix.

D. Dimension Reduction

Dimensionality reduction is a very important step in text classification, because irrelevant and redundant features often degrade the performance of classification algorithms both in speed and classification accuracy and also its tendency to reduce over fitting. The proposed system utilizes Linear Discriminant Analysis (LDA) as a well-known method for dimension reduction of matrix X [25]. Before applying LDA, training data vectors are normalized with zero-mean.

LDA starts by finding eigenvectors of $S_w^{-1}S_b$, where S_b is the between-class variance and S_w is the within-class variance. LDA tries to compute a transformation that maximizes the ratio of S_b to S_w. Eigenvectors of $S_w^{-1}S_b$ form the transformation matrix W in which $Y=W\times X$ [26]. W is the transformation matrix, and Y is the new compact matrix that represents the old matrix X. It means that each new feature is a combination of the original features. To apply the same step for test data, we need to save eigenvectors and mean of all classes; we refer to that by mapping. After applying this stage, mapping passes to test phase to reduce the dimension of the test samples matrix.

E. Multi-Objective Genetic Programming

The aim of this stage is to build a classifier that can classify the test data automatically. A classifier is a mapping, $D_m : R^p \rightarrow N_m$ where R^p is the p-dimensional real space and N_m is the set of label vectors for m-class problem and is defined as [18]:

$$N_m = \left\{ y \in R^m : y_i \in \{0,1\} \forall i, \sum_{i=1}^{m} y_i = 1 \right\} \quad (4)$$

For any vector (training data) x $\in R^p$, D_m(x) is a vector in m-dimension with only one component as 1 and all others as 0 where m is the number of classes. In this paper, our objective is to find a D_m using the combination between GP and NSGA-II. We shall use a multi-tree concept for designing classifier. The beauty of using this concept is that we can get a classifier for the multi-class problem in a single run of GP. In general, the application of GP to binary classification problems has been demonstrated with reasonable success.

In multi-objective optimization, the goodness of a solution is determined by the dominance [27]. Let x and y are two solutions, x dominates y, if and only if, $\boldsymbol{x}$ is less than or equal y in all objectives and $\boldsymbol{x}_1$ is strictly better than $\boldsymbol{x}_2$ for least one objective. A set of all the solutions that are not dominated by any other solution is called the non-dominated solution set. In NSGA-II, ranking the individuals is based on non-dominated-fronts. The selection in NSGA-II is done using tournament selection method; a solution from the lowest front is selected, if there are more than one individual, the individual with a higher crowding distance (measure of how close an individual is to its neighbors) is selected [27].

Regarding the initialization parameters of GP; terminal set T={feature variables, R}, where R contains random

generated real numbers in rang [0.0 10] and function sets F= {+,-,/,*}. The trees are initialized by using the ramped half and half method. This method is the most common way of creating the initial GP populations that produces diverse trees [12]. Individuals or chromosomes are represented by using multi-tree concept where each individual consists from m trees. Each tree represents a classifier for a specific class [18].

The fitness function determines how an individual is able to classify the documents. In our approach, we apply the multi-objective GP to improve the classifiers that suffer from large tree size. Here, fitness objectives are: (1) classification accuracy that equals the number of correctly classified documents to the total number of documents during training phase. (2) Tree complexity that is calculated by counting the number of the nodes in the tree. Each individual is evaluated against these two fitness objectives. In our case we need maximum accuracy with lowest tree size. All objectives are of the minimization type, it means a minimization type objective can be converted to a maximization type by multiplying negative one [27].

Tournament selection uses to select individual for crossover or mutation. It generates offspring with crossover and mutation and selects the next generation according to non-dominated sorting and crowding distance comparison [15]. After tournament selection, we will use roulette wheel selection to select tree from the individual, which is selected by tournament selection to apply crossover and mutation depend on the probability of unfit tree that used to give more change to the trees with low performance. Here, we modified the crossover and mutation to be appropriate with multi-representation as clarified in the following steps where K is the number of NSGA-II fronts and g number of generation:

Step1: Create a random initial population P_0 of size N.
Step2: P_0 is sorted based on the non-domination solution. Fitness is assigned to each solution according to its non-domination level. Set $g = 0$.
Step3: Apply tournament selection to select parents from P_0 for crossover, and mutation to create offspring population Q_0 of size N.
Step4: If one of the stopping condition is occurred, stop and return P_g.
Step5: Combine parent P_g and offspring Q_g populations, Set $R_g = P_g \cup Q_g$, where R_g is of size $2N$.
Step 6: R_g is sorted according to non-domination level, identify the non-dominated fronts F_1, ..,F_k in R_g.
Step 7: For i = 1,…, k do following:
- Calculate crowding distance of the solutions in F_i.
- Create P_{g+1}
Case 1: If $| P_{g+1} | + | F_i | \leq N$, then
$P_{g+1} = P_{g+1} \cup F_i$;
Case 2: If $| P_{g+1} | + | F_i | > N$, then
$P_{g+1} = P_{g+1} \cup (N - | P_{g+1} |) F_i$;

Step 8: Use crowded tournament selection based to select parents from P_{g+1}. Apply crossover and mutation to P_{g+1} to create offspring population Q_{g+1} of size N.
Step 9: Set $g = g + 1$, and go to Step 4.

The termination condition determines when GP process is stopped. Here we set two stopping condition, if all training samples N_{tr} are classified correctly or after predefined number of generation M.

F. Post Preprocessing

The aim of this stage is to enhance resultant classifier if it could not classify all training samples correctly. In text classification, a text document may partially match many categories. A classifier may assign a document to multiple classes or single class depending on the constraints. We need to find the best matching category for the text document.

1) Improving using OR-ing technique

After getting the best classifier (BCF), it is possible that the terminal GP population contains two chromosomes (individuals) one of them is good for a particular segment of the feature space, while another one is good for another segment of the feature space. To get benefit from that since the overall performance, in terms of misclassification of the two individuals could be comparable or different. Therefore, combining the two individuals by OR-ing may result in a much better classification tree as comprehensively explained in [18]. The combined classifier which correctly classified a higher number from the training samples is taken as a resultant classifier *CF*. Using this method may increase the number of correctly classified samples and may not which means the best individual through the GP process is the best at all.

2) Conflict resolution

It can be happened that more than one tree of *CF* show positive responses. In this case, we face a conflicting situation. To solve that, we use a set of heuristic rules followed by a weighting scheme as employed in [18]. The aim of the heuristic rules is to identify the typical situations when the classifier cannot make unambiguous decisions and utilize that information to make decisions; the decision here means to assign class label to a document in the conflict cases. May be heuristic rules can't solve conflict situation if there is no heuristic rule at all. So, a weighting scheme is used. The objective of this matrix is to calculate the false positive cases and the false negative cases which will use to enhance the classifier *CF* behavior according to them; see [18] for more details.

It is possible that neither the weight-based scheme nor the heuristic rule is able to assign a class label to the conflicted samples. Note that the OR-ing operation, the heuristic rule generation, and computation of the weights are done only once after the GP terminates. Consequently, the additional computation involved for the post-processing, as we shall see later, is much less than the time required for the GP to evolve.

IV. Experimental Results

To evaluate the performance of the proposed model, we conduct a set of experiments to measure its efficiency and compare it with standard GP classifier and previous work in [8]. Our experiments were implemented using MatLab

2011b. All the experiments were based on a PC with Windows 7 Ultimate 64-bit, with Intel (R) Core(TM) i3 CPU, 2.53GHz and 4GB RAM. Experiments were performed using the Reuters-21578 dataset (http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html). We used the top 6 categories and discarded documents with no label or with multiple labels. The distribution of documents into the categories is unbalanced in this dataset. Furthermore, 20 Newsgroups dataset (https://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html) is used also to show the ability of the system to deal with balanced dataset (roughly equal number of documents inside each category).

To evaluate the performance of the proposed classifier, *micro-average* and *macro-average* F-measure are commonly used [28], where TP_i is the number of correctly classified documents for a class i , FP_i is the number of documents wrongly classified for a class i, and FN_i is the number of documents which belong to class i but not assigned to it. Here, preparing the training documents affect dramatically in the classification result.

$$F_{(macro-avg)} = \frac{\sum_{i=1}^{m}\left(\frac{2*\rho_i*\pi_i}{\rho_i+\pi_i}\right)}{m}, \quad F_{(micro-avg)} = \frac{2*\rho*\pi}{\rho+\pi} \quad (5)$$

$$\pi_i = \frac{TP_i}{TP_i+FP_i}, \ \rho_i = \frac{TP_i}{TP_i+FN_i} \quad (6)$$

$$\rho = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + FN_i}, \ \pi = \frac{\sum_{i=1}^{m} TP_i}{\sum_{i=1}^{m} TP_i + FP_i} \quad (7)$$

Many experiments were done to determine the best parameters values in each proposed system's step to achieve the best classification performance. In the experiments, the best LDA dimension to reduce the data is m-1. For Reuter's dataset, the best information gain threshold is 0.2 and for 20news-group dataset is 0.1. We train the classifier with 90% from the dataset and the rest for testing. This large percentage for training the classifier is used because the proposed model classifies documents for all classes at the same time. Furthermore, increasing number of generations more than 30 do not affect on the result of the classifier and also increasing population size for more than 250 do not influence on the process; it takes more time only without increasing the classification accuracy. Since each word in a document is a key to classify it, tokenizing the document to a set of single (gram) words is more efficient than using bi-gram tokenization, this gives more accuracy.

In Table I and Table II, the results show that combining genetic programming with NSGA-II improves the classification performance rather than using GP only. This improvement in the performance affects on GP run time. The reason behind that is the complexity of the NSGA-II which is $O(BN^2)$ where B is the number of objective and N is the population size. We run the system 10 times and calculate the average for each comparison criteria. Our proposed model deals with all classes at the same time; to examine the classification result per category against binary classification result, Table II shows that there is improving in the classification results rather than binary classification results for the model in [8].

TABLE I. RESULTS ON REUTERS-TOP (AVERAGE)

	Standard GP	Hirsch et al.[8]	Proposed system
Class	F-Measure_ average		
Earn	0.6185	0.857	0.859
Acq	0.5307	0.755	0.861
Crude	0.7941	0.826	0.697
Trade	0.6121	0.761	0.821
Money-fx	0.5183	0.612	0.767
Interest	0.2833	0.569	0.855
Micro_ average	0.614	---	0.826
Macro_ average	0.559	0.73	0.808
Average-GP run time (min)	2.7259	around 20 min/class	4.1813
Average-best individual node count	45	---	30

TABLE II. RESULTS ON 20 NEWSGROUPS- TOP (AVERAGE)

	Standard GP	Proposed system
Class	F-Measure_Average	
comp.graphics	0.500	0.640
comp.os.ms-windows.misc	0.554	0.833
comp.sys.ibm.pc.hardware	0.550	0.768
comp.sys.mac.hardware	0.611	0.855
comp.windows.x	0.692	0.550
misc.forsale	0.664	0.666
Micro_ average	0.629	0.787
Macro_ average	0.595	0.646
Average -GP run time (min)	2.011	3.288
Average –best individual node count	40	30

V. CONCLUSION AND FUTURE WORK

This paper proposes a document classification system based on multi-objective genetic programming as a supervised learning technique. The system deals with multi-class document classification without dividing it to binary classifications; unlike the majority of others classification techniques. This capability is resulting from employing multi-tree representation to represent each individual. The system achieves a good performance compared to standard genetic programming. To achieve a reasonable performance, the training data must be balanced or closer to balance. In the future, we suggest using incremental learning to improve classification results and test the system on different datasets. Also, we plan to test the system's behavior when the number of classes is more than 6.

REFERENCES

[1] F. Sebastiani, "Machine learning in automated text categorization," *ACM Computing Surveys (CSUR)*, vol. 34, no. 1, pp. 1-47, 2002.

[2] N. Chen and D. Blostein, "A survey of document image classification: Problem statement, classifier architecture and

performance evaluation," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 10, no. 1, pp. 1-16, 2007.
[3] N. VasfiSisi and M. R. F. Derakhshi, "Text classification with machine learning algorithms," *Journal of Basic and Applied Scientific Research*, vol. 3, no. 1, pp. 31-35, 2013.
[4] Nidhi and V. Gupta, "Recent trends in text classification techniques," *International Journal of Computer Applications*, vol. 35, no. 6, pp. 45-51, 2011.
[5] S. Ramasundaram and S. Victor, "Algorithms for text categorization: A comparative study," *World Applied Sciences Journal*, vol. 22, no. 9, pp. 1232-1240, 2013.
[6] P.Y. Pawar and S. Gawande, "A comparative study on different types of approaches to text categorization," *International Journal of Machine Learning and Computing*, vol. 2, no. 4, pp. 423-426, 2012.
[7] M. Ikonomakis, S. Kotsiantis, and V. Tampakas, "Text classification using machine learning techniques," *WSEAS Transactions on Computers*, vol. 4, no. 8, pp. 966-974, 2005.
[8] L. Hirsch, M. Saeedi , and R. Hirsch, "Evolving text classification rules with genetic programming," *International Journal of Applied Artificial Intelligence*, vol. 19, no. 7, pp. 659-676, 2005.
[9] Bhumika, S. S. Sehra, and A. Nayyar, "A review paper on algorithms used for text classification," *International Journal of Application or Innovation in Engineering & Management*, vol. 2, no. 3, pp. 90-99, Mar. 2013.
[10] B. Baharudin, L. H. Lee , and K. Khan, "A review of machine learning algorithms for text-documents classification," *Journal of Advances in Information Technology*, vol. 1, no. 1, pp. 4-20, 2010.
[11] D. Kumar and S. Beniwal, "Genetic algorithm and programming based classification: A survey," *Journal of Theoretical and Applied Information Technology*, vol. 54, no. 1, Aug. 2013, pp. 48-58.
[12] H. Jabeen and A. R. Baig, "Review of classification using genetic programming," *International Journal of Engineering Science and Technology*, vol. 2, no. 2, pp. 94-103, 2010.
[13] J. Kishore, L. M. Patnaik, V. Mani , and V. Agrawal, "Application of genetic programming for multicategory pattern classification," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 3, pp. 242-258, 2000.
[14] D. Parrott, X. Li, and V. Ciesielski, "Multi-objective techniques in genetic programming for evolving classifiers," in *Proc. the IEEE Congress on Evolutionary Computation*, USA, Sep. 2-5, 2005, pp. 1141-1148.
[15] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 849-858, Apr. 2000.
[16] N. Mehra and S. Gupta, "Survey on multiclass classification methods," *International Journal of Computer Science and Information Technologies*, vol. 4, no. 4, pp. 572-576, 2013.
[17] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all Schemes," *Pattern Recognition Journal*, vol. 44, no. 8, Aug. 2011, pp. 1761-1776.
[18] D. P. Muni, N. R. Pal, and J. Das, "A novel approach to design classifiers using genetic programming," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 183-196, Apr. 2004.
[19] N. Priyadharshini and V. MS, "Genetic programming for document segmentation and region classification using discipulus," *International Journal of Advanced Research in Artificial Intelligence*, vol. 2, no. 2, pp. 15-22, 2013.
[20] B. Svingen, "Using genetic programming for document classification," in *Proc. 11th International Florida Artificial Intelligence Research (FLAIRS)*, USA, May 1998, pp. 63-67.
[21] V. Korde and C. N. Mahender, "Text classification and classifiers: A survey," *International Journal of Artificial Intelligence & Applications*, vol. 3, no. 2, pp. 85-99, Mar. 2012.
[22] A. G. Jivani, "A comparative study of stemming algorithms," *International Journal of Computer Technology and Applications*, vol. 2, no. 6, pp. 1930-1938, Nov-Dec. 2011.
[23] H. Uguz, "A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm," *Knowledge-Based Systems*, vol. 24, no. 7, pp. 1024-1032, 2011.
[24] Y. Xu, G. J. Jones, J. Li, B. Wang, and C. Sun, "A study on mutual information-based feature selection for text categorization," *Journal of Computational Information Systems*, vol. 3, no. 3, pp. 1007-1012, 2007.
[25] K. Torkkola, "Discriminative features for text document classification," *Journal of Formal Pattern Analysis & Applications*, vol. 6, no. 4, pp. 301-308, 2004.
[26] G. Biricik, B. Diri, and A. C. Sonmez, "Abstract feature extraction for text classification," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 20, no. 1, pp. 1137-1159, 2012.
[27] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *International Journal of Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992-1007, 2006.
[28] A. Ozgur, L. Ozgur, and T. Gungor, "Text categorization with class-based and Corpus-based keyword selection," *Lecture Notes in Computer Science*, vol. 3733, pp. 606–615, 2005.

Adel A. El-Zoghabi received his B.Sc. in computer engineering from Alexandria University in 1987, his M.Sc. and Ph.D. in information technology from Alexandria University and Old Dominion University in 1991 & 1994 respectively. His research and professional interests include intelligent systems and machine learning, inter networking & routing protocols, and distributed systems. He has published many papers in international journals and conferences worldwide during the past three decades. Currently, he is a professor of computer science and IT and the head of Dept. of Information Technology since August 2012.

Saad M. Darwish received his Ph.D. degree from the Alexandria University, Egypt. His research work concentrates on the field of image processing, optimization techniques, security technologies, computer vision, pattern recognition and machine learning. He has published in journals and conferences and severed as TPC of many international conferences. Since Feb. 2012, he has been an Associate Professor in the department of information technology, Institute of Graduate Studies and Research, Egypt.

Doaa B. Ebaid received the B.Sc. degree in computer science and statistic from the Faculty of Science, Alexandria University, Egypt in 2011, she is doing the M.Sc. degree in the Institute of Graduate Studies and Research (IGSR), Department of Information Technology, University of Alexandria, she has been a Demonstrator in the department of information technology, IGSR. Her research and professional interests include Machine learning, Multi-objective techniques.