The CSI Journal on
Computer Science and Engineering
Vol. 17, No. 1, 2019
Pages 34-48
Regular Paper

An Improved Back-Propagation with PSO in MLP-ANNs for Authorship Identification

Azadeh Salamzadeh, Farhad Soleimanian Gharehchopogh

Department of Computer Engineering, Urmia Branch, Islamic Azad University, Urmia, IRAN

Abstract

Scientific theft and plagiarism (academic papers and books), publishing inappropriate texts, or threatening letters. Therefore, the language is necessary for a security aspect that is recognized by the author. In this paper a larger data set containing 2500 texts for training and 2500 texts for testing are used. A new model for improving Back-Propagation (BP) has been presented with a Particle Swarm Optimization (PSO) algorithm in the Multi-Layer Perceptron Artificial Neural Network (MLP-ANN) for authorship identification of the new model. In this paper, we have used features such as lexical, syntactic and structural features for authorship identification. Finally, different criteria such as the number of correct and incorrect classified data, precision, and recall percentage have been used. The obtained precision and the recall percentage for proposed models are equal to 0.9992 and this criterion in BP is equal to 0.9849 and 0.9588. The above-mentioned results indicate the proposed model is superior to BP.
Keywords: Authorship identification, Natural language process, Artificial neural networks, Multi-layer perceptron, Particle swarm optimization**.**

1. Introduction

The development of network technologies, especially the Internet, has created new ways to share information. Although the use of internet and computer networks has increased the quality of life in many aspects, moreover, it has increased criminal activity. These criminal activities are called cybercrimes. The unauthorized publication is an example of these cybercrimes. Internet criminals use many web-based channels such as emails, websites, internet news, internet nodes, and internet chat rooms to publish unauthorized contents. The plagiarism in scientific papers, such as academic papers and books, is other example of cybercrimes [1, 2]. In other words, the problem of authorship identification of texts is very necessary to enhance the security of cyberspace. The problem of authorship identification has been derived from the topic of the text category, which is in it a branch of the natural language. In recent years, practical applications of this issue has increased significantly in cases such as intelligence, criminal law, civil and computer security, spam filtering, cybercrime, anti-plagiarism, author recognition, web application management [3], source code authorship identification, malicious code detection, software intellectual property infringement, software maintenance and update [4, 5].

The purpose of this study is to adapt the anonymous texts to the author of those texts by using a series of similarity criteria that are derived from the training of other texts written by the same author. For the purposes of the similarity criteria, the same styles of writing or hidden features written in the text in order to distinguish the author from others. In this regard, after the extraction of features, the discussion of the classification of texts is raised. Text documents categorizing is an important issue in librarianship science as well as computer science. The main task of the classification is to divide the documents into one or more categories. This can be done either manually or by using a computer. Classification is done manually, so that is done in a library, and books are divided into different

categories. Classifying by computer happens while documents related to computer science, are classified, this type of classification falls into the category of data analysis and it is carried out in the form of machine learning. Subject or a specific attribute (such as the author's name) may categorize documents. In the machine learning process, the system typically carries out such learning that has been taught on a set of pre-tagged texts, and then the classification of new texts takes place by using models derived from the training phase [6, 7].

The purpose of classification is to label the texts based on their authors. In fact, this kind of classification is an observer classification. A set of text documents that have been identified by their authors are provided as training data for the proposed model so that they can assign new entries to one of these authors by learning from this collection. The high number of correct assignments indicates that its precision is more than the proposed model. So far, authors have been identified by types of ways, but more research is necessary to achieve a precision and higher-speed technique. Each author follows a particular style in his writings and speeches, which it is called as the author's writing style [2, 3] Writing style is used as a distinguishing feature for the discovery of text authors. In such a way, that the texts of authors are specified, they can be classified as educational data into the model and after learning, we can observe them in the proposed classification model for texts that the author cannot be specified by trained classification algorithm to one of the existing authors of the assignment. In total, the authorship identification takes place in four stages as shown below [8, 9]:

Data Collection: There are valid datasets for authorship identification that can be used for the proposed model and the author identifies the texts.

Feature Extraction: Features that are often used for authorship identification of texts including: lexical features, syntactic features, structural features, application-specific attributes. After extracting the desired features, each text is displayed as a vector of those features.

Model Generation: The techniques used to generate a model for authorship identification are divided into two categories algorithms statistical methods and machine learning methods. After selecting the model, the data collected by these methods is divided into two training sets and a testing set, and the model is applied to them.

Authorship Identification: The model is used for authorship identification, and for each test text, an author is determined. A larger number of texts, the author recognizes correctly, namely, the precision the model is improved.

This paper uses the PSO to determine the initial weights for BP ANN. In this paper, the PSO algorithm is easy to achieve the global optimum, high precision, and fast convergence which make up for the lack of BP ANN. The initial threshold and the weights of the BP ANN are optimized by the improved PSO algorithm, and precision of the BP ANN are improved. BP ANN structure was constructed according the number of input data, and PSO algorithm is used to get the optimal initial weights and biases of BP ANN, which can effectively overcome the shortcoming of BP ANN that it is easy to fall into local optimal solution, and increase the convergence speed of BP ANN.

The remainder of this paper is organized as follows: Section 2 discusses about the studies related to Authorship Identification. The proposed model and its framework are presented in Section 3. Analysis and discussion are presented in Section 4. Finally, the conclusions and future works are presented in Section 5.

2. Background and Related Works

2.1. Artificial neural networks

MLP-ANNs belong to a large collection of leading ANNs and are one of the most popular and most prolific ANNs in life today. These networks consist of a set of neurons called 'perceptron'. In these types of networks, the neurons are grouped in several layers. The first layer, which indicates the input called input layer and the last layer that represents the outputs, is called output layer. Usually these networks, in addition to the two layers above, also include one or more hidden layers [10].

If we assume that our ANN has L layer, the first layer, the input layer and the L^{th} layer, the output layer, and the layers 2 to L-1, (h_1, h_2, ..., h_N), are hidden layers. For connections among neurons, the neurons of the lower layers are higher to the neurons of the higher layers, and there is no connection among the neurons of a layer. In this network, all neurons, except the input layer neurons, receive inputs from other neurons in different layers. Moreover, the outputs of each neuron are connected to other neurons or out of the ANN.

The number of neurons in the input layer is as large as the size of the input pattern and the number of neurons in the output layer as much as the number of classes that have been categorized. The inputs (x_i) are connected to the neurons of the first layer and each neuron is connected to the other neurons through weighted verticals (w_{ij}). In order to make this ANN easier to use, each neuron, in addition to the inputs, adds another fixed value called the bias (B_i), which is usually considered as the number one. These biases are connected to the neurons through a weighted edge (w_{ij})[11].

How to calculate the output (y_i) in each neuron that the inputs of each neuron are multiplied by their respective weight and the results are summed together (Σ). The sum of the multiplications is passed through an activation function (f) and finally the output of the neuron is calculated. These outputs act as inputs for neurons of later layer. Figure 1 shows the structure of the ANN of the proposed model and the calculation method of the outputs in each neuron [12].

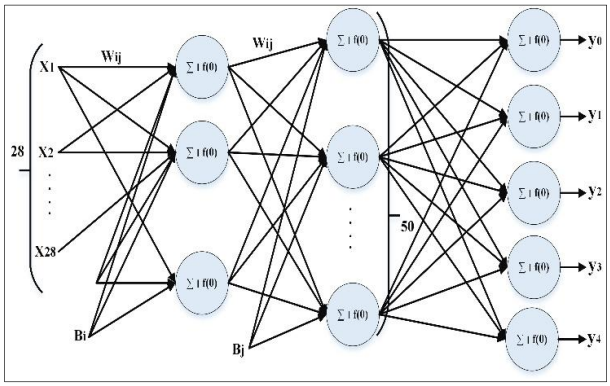

Fig. 1: MLP-ANN structure of the proposed model and how to calculate outputs per neuron.

The total output of each neuron is calculated from Eq. (1) in which w_i corresponds to the input and the desired neuron, x input vector, b is the bias variable and f is the activation function.

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \quad (1)$$

In general, ANNs are divided into two categories at the moment of learning techniques:

1. Supervised Learning Networks: In this type of network at the training stage, the ideal output corresponding to each sample is predefined, and in fact, all samples are tagged in the training step.
2. Unsupervised Learning Networks: In this type of network, the output of each instance is not already known and it is based on a criterion (for example, distance) and also it is based on a type of competition, outputs are in separate classes.

In addition, providing samples to the ANN is possible in two ways [13-15]:

- Batch Method: In this method, all samples are presented to the network. Finally, the network error is calculated from the total samples and the weights are varied based on the error rate. This will continue until the error reaches an acceptable level. This method is time-consuming and requires a lot of memory as well as the ability to crash the algorithm in local minima.
- Pattern Method: In this method, at each step of the algorithm (in each epoch), the samples are given individually to the grid and the corresponding error with that sample is immediately calculated and the network weights are calculated based on that error. Then the next sample is presented to the network and the steps are repeated.

This method is more efficient than the batch method when the large data set contains bulk and redundant data or repetitive rules. On the other hand, due to the nature of the randomness of this method, the risk of falling into the local minima is eliminated. The nature of randomness can be increased by changing the order of samples in the training set.

There are many methods to train the network and modify the weights to achieve a meaningful error. Among these, PSO and BP are described below.

PSO is a global optimization algorithm[16]. This algorithm is a part of the social and population-based intelligence algorithm inspired by the collective movement of birds seeking food. Each member of this population has two attributes of speed and location, which vary depending on how each of these members moves in space and in what space they are. At the same time, each of these members has a memory that records the best position that has ever had in the search space and performs the next move based on this position. This algorithm is used due to many advantages, including simplicity and easy implementation in many areas, such as ANN training, stock optimization, pattern recognition, routing, robot motion control [17]. In this algorithm, each particle is defined as a potential solution to the problem in n-dimensional space. The position of particle i is in the form of a $X_i = (x_{i1}, x_{i2}, \ldots, x_{iN})$ vector, and the velocity of particle i is in the form of $V_i = (v_{i1}, v_{i2}, \ldots, v_{iN})$ vector and the best position of the particle is represented as a $P_i = (p_{i1}, p_{i2}, \ldots, p_{iN})$ vector. Each particle changes its position based on two factors:

- Based on the distance between your current position and the best position ever (pbest).
- Based on the distance between your current position and the best position of its neighbors (gbest).

The speed of each particle is calculated through Eq. (2).

$$\vec{v}_i(t+1) = \omega\vec{v}_i(t) + r(0,1)C_1\left(\vec{P}_{l,i} - \vec{x}_i(t)\right) + r(0,1)C_2\left(\vec{P}_g - \vec{x}_i(t)\right) \quad (2)$$

The position of each particle is represented through the Eq. (3).

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t) \quad (3)$$

In the equations (3, and 4), i, the number of each particle $i \in \{1,2,\ldots,n\}$, n the number of particles, $\vec{v}_i(t+1)$ the velocity of particle i in time (t + 1), w, the inertial coefficient, $\vec{v}_i(t)$ the particle velocity at time t, $r(0,1)$ our random number between (1 and 0), C_1 The degree of impact of the local position, C_2 the degree of impact of the general position, $\vec{P}_{l,i}$ the best position of the particle, $\vec{P}_g$ the best position of the particles, $\vec{x}_i(t)$ the current position of the particle in time (t).At each position update and the speed update, it must be checked that the particle position in the interval $[X_{min}, X_{max}]$. Moreover, the particle velocity is in the interval $[V_{min}, V_{max}]$. The inertial weight starts at 0.9 and decreases by 0.4 during the run. Large quantities of inertial weight make it possible for the PSO to first search for larger space and quickly find the situation where the optimal response exists. By reducing the inertia, the particle speed decreases to provide a more detailed and precise answer to the optimal response. This kind of selection of inertial factors will increase the convergence rate of the algorithm and will greatly improve the performance of the algorithm. An appropriate choice of inertia leads to a balance between local and global exploration and ultimately leads to find the ideal solution in a small number of epochs. Coefficients C_1,C_2 are called Learning Factors. They are weighing accelerated expressions that will take each particle to the position of pbest and gbest. Therefore, any manipulation and adjustment on them will change the amount of stress in the system. The low values of these constants make it possible for the particles to distance themselves from the target areas and migrate. This is while large quantities cause unexpected movements to target areas. The coefficient C_1 shows the tendency of particles to repeat past behaviors that succeeded, but the coefficient C_2 is the particle tendency to track the success of other particles.

Generally, C_1 and C_2 are set to the same values so that the search process of the peripheral areas by focusing on pbest and gbest is better covered, and the same setup will give the same importance to these learning factors.

All particles tend to move towards better situations. Hence, the best position (optimal answer) is ultimately obtained by combining the effort of all particles. The conditions for

stopping this algorithm are stopped after a certain number of repetitions or it will stop after reaching the optimal value. An algorithm for PSO, which is easy to understand, is called as algorithmic.

BP was proposed by Rumelhart and et al. in 1988 and therefore, it has been used extensively in feed-forward ANNs for weight training[18]. Feed-forward means that the neurons are placed in different layers and they forward their outputs, and after calculations, the calculated errors are fed back to the grid to correct the weights (BP). This algorithm is a component of learning algorithms, which means that the samples given as inputs to the algorithm are based on their respective ideal output. Therefore, the network output is compared with these ideal values and the network error is calculated. The goal of this algorithm is to reduce this error rate until all the examples are learned by the learning network[19].

The steps for implementing this algorithm are as follows.

Step 1: First, all weights of the network are based on small random values between 0 and 1.

Step 2: Follow Steps 3 to 10 as long as the stop condition occurs.

Step 3: For each sample that is paired with input and its ideal value, performance takes place through steps 4 to 9: (Feed Forward).

Step 4: Each neuron receives the component x_i of the input vector $(X_{i,} i = 1,2, \dots, n)$ from the input layer and send it to all the neurons in the higher layers (middle layer).

Step 5: For each neuron in the middle layer $(Z_j j = 1,2, \dots, p)$, the inputs of that neuron are multiplied by their respective weight, and the results together shape Eq. (4). Then, the sum of the results is applied to the activation function of that layer, and the output of the neuron is given by the Eq. (5). Finally, this output is sent to higher layer neurons (output layer).

$$z\_in_j = v_{oj} + \sum_{i=1}^{n} x_i w_{ij} \quad (4)$$

$$z_j = f(z\_in_j) \quad (5)$$

Step 6: For each neuron in the output layer $(Y_{k,} k = 1,2, \dots, m)$, the inputs of that neuron are multiplied by their respective weight, and the results are summed together in the form of Eq. (6). Then the sum of the actions is applied to the activation function of that layer and the output of that neuron is calculated in the form of Eq. (7).

$$y\_in_k = w_{ok} + \sum_{z=1}^{p} z_j w_{jk} \quad (6)$$

$$y_k = f(y\_in_k) \quad (7)$$

Then, for each neuron in the output layer, the actual value of the neuron is calculated by the network output for it. Finally, using the Eq. (8), the total network error is calculated, and the algorithm is executed in step 7 to reduce this error.

$$E = {}^1/_2 \sum_{d=1}^{n} \sum_{k=1}^{m} (o_k - y_k) \quad (8)$$

In Eq. (8), E the total network error, $(d = 1,2, \dots, n)$ the set of training samples that are trained on the network, $(k = 1,2, \dots, m)$, the set of neurons in the output layer, o_k, the ideal output corresponding to the k[th] neuron, y_k, is the real output obtained from the k[th] neuron.

Step 7: For each neuron in the output layer $(Y_{k,} k = 1,2, \dots, m)$, the error value is calculated using Eq. (9).

$$\delta_k = (t_k - y_k) f'(y\_in_k) \quad (9)$$

Therefore, in Eq. (9), t_k is the real and ideal value for k[th] neuron, y_k the value obtained from the feed-forward MLP-ANN for that neuron is $f'(y\_in_k)$, the derived activation function for the output is y_in_k. The amount of weight change for each output unit is derived from Eq. (10) and the amount of bias change for each output unit, will be calculated using Eq. (11).

$$\Delta w_{jk} = n \delta_k z_j \quad (10)$$

$$\Delta w_{ok} = n \delta_k \quad (11)$$

In Eq. (11), n is the learning rate is often an arbitrary number between 1 and 10.

Step 8: For each neuron in the hidden layer $(Z_j,\ j = 1,2, \dots, p)$, the sum of the values δ obtained from the higher layer neurons is calculated using Eq. (12).

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad (12)$$

Then, to calculate the error of each neuron in this layer, the result of the multiplication of the values δ obtained from the derivation of the activation function is calculated using Eq. (13), and then the weighting values for each hidden unit is derived from Eq. (14) and the bias changes values for Each hidden unit is obtained from Eq. (15).

$$\delta_j = \delta\_in_j f'(z\_in_j) \quad (13)$$

$$\Delta v_{ij} = n \delta_j x_i \quad (14)$$

$$\Delta v_{oj} = n \delta_j \quad (15)$$

In the Eq. (15), n is the Learning Rate, which is often considered arbitrary between (1 and 0).

Step 9: For each neuron in the output layer $(Y_{k,} k = 1,2, \dots, m)$, the weights are updated using Eq. (16) and biases are calculated by using Eq. (17).

$$w_{jk}(new) = w_{jk}(old) + \Delta w_{jk} \quad (16)$$

$$w_{ok}(new) = w_{ok}(old) + \Delta w_{ok} \quad (17)$$

In addition, for each neuron in the hidden layer $(Z_j j = 1,2, \dots, p)$, the weights are updated by using Eq. (18) and biases are calculated by using Eq. (19).

$$v_{ij}(new) = v_{ij}(old) + \Delta v_{ik} \quad (18)$$

$$v_{oj}(new) = v_{oj}(old) + \Delta v_{oj} \quad (19)$$

The condition for stopping the algorithm is checked. Various conditions can be used to stop the utilization of this algorithm, such as: stopping after the epoch for certain times, stopping when the error is reduced from a given value, stopping when occurs that the error in the instruction examples follows a certain rule.

2.2. Related works

Although the problem of authorship identification is a new issue in the processing of natural language and the classification of texts, but many studies have been conducted on which results have been fruitful. Each method has advantages and disadvantages, and in the proposed model, we tried to offer a better solution for testing algorithms. Here are some of the researches that are carried out before our study in this area.

Zheng et al. [1] proposed the ANN for authorship identification on two datasets, English and Chinese. The English data sets consist of an email message collection and an Internet newsgroup message collection. The Chinese data set consists of a Bulletin Board System (BBS) message collection. Three learning algorithms (classifiers) were used in the experiments for comparison purposes, including decision tree (DT), BP ANNs, and support vector machines (SVM). The amount of precision and recall for the English data set using the ANN is 90.10 and 91.43 and for the Chinese data is 82.40 and 82.13 respectively. They achieved average prediction accuracies of 80%-90% for email messages, 90%-97% for the Newsgroup messages, and 70%–85% for BBS messages. However, we have used the standardized data for standard methodology, including 2500 text and 50 different authors, and the value obtained for the two factors mentioned in it was 99.92%.

De VEL et al. [20] studied authorship identification on e-mail collections using SVM. The collections of e-mail documents used in the experimental evaluation of author-topic categorization contained a total of 156 documents sourced from three native language (English) authors, with each author contributing e-mails on three topics (approx. 12,000 words per author for all topics). The topics chosen are movies, food, and travel. Total of 170 features are used in the experiment based on style marker attribute type (i.e.: Number of blank lines/total number of lines, Average sentence length, Average word length (number of characters)). Total of 21 features are used in the experiment based on structural attribute type (i.e.: Has a greeting acknowledgment, uses a farewell acknowledgment, Contains signature text, Number of attachments, Position of requited text within e-mail body, HTML tag frequency distribution/total number of HTML tags). The amount of precision, recall, and accuracy obtained from their research for the third author, which has 63 messages, is 93.80, 89.60, and 91.60 respectively. Nevertheless, the proposed model uses 50 authors and 50 texts per author, and the values obtained for the two above-mentioned factors are more precise.

In [21], data mining techniques have been used to authorship identification. In order to solve the problem of document authors' detection, 10,918 documents were collected from Associated Press (AP) and 365 different Function Word. They used five classification techniques such as Naive Bayesian (NB), Bayesian Networks (BN), K-Nearest-Neighbor (KNN), three nearest-neighbor (3NN), and DT. The two Bayesian approaches are based on probabilities. The KNN methods use vector differences. Decision trees are based on classifying training data by their distinguishing features. Results for NB, BN, NN, 3NN, and DT are equal to 85.53, 90.46, 85.77, 85.53, and 84.53, respectively.

In [22], researches have investigated the authorship attribution of Turkish texts using various feature vectors on different datasets. Dataset I consists of 630 singly-authored documents written by 18 different authors, with 35 different texts written on different topics. Dataset II is formed from 315 singly-authored documents written by 9 different authors, with 35 different texts written on the same topic, while Dataset III consists of 315 singly-authored documents written by 9 different authors, with 35 different texts written on different topics. Function words, lexical, statistical, grammatical, and n-gram features are automatically extracted from documents and formed different feature vectors. They applied 5 classification algorithms that are Naive Bayes (NB), Random Forest (RF), MLP, SVM, and KNN. Ten different feature vectors are obtained from authorship attributes, n-grams and various combinations of these feature vectors that are extracted from documents, which the authors are intended to be identified. Comparative performance of every feature vector has been analyzed by applying NB, SVM, KNN, RF, and MLP classification methods. The most successful classifiers are MLP and SVM. The best performance in Dataset I, 92.5%, is obtained from SVM. The best result in Dataset II, 95.4%, is achieved from MLP while maximum success ratio in Dataset III, 96.9%, is obtained from MLP.

In [23], a text-independent method which uses Gradient features and ANN classifier has been proposed for Persian writer identification. The proposed method has been tested on a Persian handwritten database. At first, the feature extraction phase that is based on Gradient features and secondly the classification step that is based on training an MLP using feature vectors. Experiments have been done on 250 documents (50 writers and 5 pages per each). Accuracy of their research on these texts, using the MLP is 94%.

N.M. DEMIR [24]proposed a method to classify the texts of authors using a set of lexical descriptors based on self-organizing Map (SOM). The SOM is a type of ANN. The SOM has been designed to classify the books of three writers. As a clustering technique, the SOM method finds similar data to another or dissimilar data to another data. It is based on competitive learning. The SOM method is used to choose different data samples and then choose one sample for data set and calculate distance between all neuron vectors. The rate of precision is equal to 69.1 on 1755 documents.

Gazzah and Amara in [25] described the design and implementation of a system that identify the writer using off-line Arabic handwriting that is 180 text samples. This approach is based on the combination of global and structural features. They used Genetic Algorithm (GA) for feature subset selection in order to eliminate the redundant and irrelevant ones. The MLP classifier has been used for classification. Experiments have shown writer identification accuracies

reach acceptable performance levels with an average rate of 94.73% using optimal feature subset.

Nirkhi et al. [26] proposed Stylometric approach for author identification of online messages. The various steps for calculating Authorship is as follows: Step1 is calculating frequencies of words and identifying most frequent words from entire corpus. Then step2 calculates Normalized frequency by calculating total percentage of the most frequent word in that document with respect to entire corpus. In step3 Z-score approach is used. Step4 calculates distance table by finding distance between two matrices. Thus, by representing text into numeric representation that is feature extraction, clustering and classification techniques of machine learning can be applied. Reuter_50_50 Data set is used for experiments. It consists of total 50 authors and 50 documents per author. Therefore, training corpus consists of 2,500 texts and test corpus also consists of 2500 text which is non-overlapping with training texts. They achieved precision 85.01 for the unigram features by the SVM which is more in compared to Delta and KNN classifier.

Text classification for authorship attribution using Naive Bayes (NB) classifier with limited training data has been proposed[27]. In first, the data is sent to pre-processing algorithm, for tokenization, punctuation marks removal, and normalization. This step of text pre-processing is crucial in determining the quality of the next stages, feature extraction and classification stage. The dataset is collected from an Arabic website. Also, all the texts that have been collected are about Travel subject. Then NB classifier is employed in order to classify the texts to their authors. Accuracy of good attribution using NB Classifier is equal to 96.67%.

Other studies on Reuter_50_50 datasets by M.S. NIRKHI et al. in 2015 were performed for authorship identification of texts [28]. They used a set of vocabulary and character-based features, using SVM and KNN algorithms for categorization, and eventually they achieved 93.3% precision by SVM algorithm at the level of one-gram word and 74.2% precision by SVM algorithm at the level of Character Bi-gram and Character Tri-gram.

In Table (1), comparison of the proposed models for Authorship Identification by researchers is shown.

Table 1: comparison of the proposed models for Authorship Identification

Refs	Authors	Algorithms	Datasets	Precision	Recall	Accuracy	FS
[1]	Zheng et al.	DT	English data (English Email Messages and English Internet	79.03	78.27	77.14	√
			Newsgroup Messages (Chinese BBS Messages)	71.83	72.37	72.58	
		BP	English data (English Email Messages and English Internet	90.10	91.43	90.00	
			Newsgroup Messages(Chinese BBS Messages)	82.40	82.13	82.25	
		SVM	English data (English Email Messages and English Internet)	91.23	91.74	91.43	
			Newsgroup Messages(Chinese BBS Messages)	83.92	81.88	82.58	
[20]	De VEL et al.	SVM	E-MAIL CORPUS	93.80	89.60	91.60	X
[21]	Zhao & Zobel	NB	AP	85.53	-	-	√
		BN		90.46	-	-	
		NN		85.77	-	-	
		3NN		85.53	-	-	
		DT		84.53	-	-	
[22]	Turkoglu et al.	MLP	315 documents	-	-	96.90	√
[23]	Ram and Moghaddam	MLP	250 documents	-	-	94.00	√
[24]	DEMIR	SOM	1755 documents	69.1	-	-	X
[25]	Gazzah et al.	GA-MLP	180 text samples	-	-	94.73	√
[26]	Nirkhi et al.	SVM	Reuter_50_50	85.01	-	-	√
[27]	Howedi and Mohd	NB	Arabic Text	-	-	96.67	√
[28]	Nirkhi et al.	SVM	Reuter_50_50	93.3	-	-	X
		KNN		74.2	-	-	
-	Proposed Model	BP-PSO	Reuter_50_50	99.92	-	-	√

3. Proposed Model

Given the importance of the problem of authorship identification of texts in the field of natural language processing, in this paper, a solution for the problem is presented; a new solution based on BP improvement using PSO in the MLP-ANN is presented. Since BP has a great deal of localized capability and also it is ineffective in locating a global optimum, as well as it has a low search speed, we propose to improve the performance of this algorithm using the PSO, which is a global algorithm and has the ability to find global optimizations. We have used PSO to optimize weights and biases of MLP. The proposed model used to optimize weights and architecture of MLP based on the PSO. In this paper, the main idea of PSO-BP ANN is to use the global search ability of PSO algorithm to obtain the initial weights and biases of BP ANN, so as to overcome the problem that BP ANN is easy to fall into local optimal solution, and improve the convergence speed of BP ANN.

ANN is initially trained by the PSO until the proportion of the function in different epochs does not change or it is outstandingly a little different. There are training and testing phases in this strategy and the use of MLP ANN with supervised learning technique. In this way, in the training phase, 2500 texts that have been identified their author as training data are given individually to the proposed model, and their errors is calculated and the weights of network change based on error rate. Then the next text is presented to the proposed model and steps are repeated until the conditions for stopping the algorithm to be prepare. In the test phase of 2500 texts that are completely different from the texts in the training set, is given to proposed model and the system is based on the values of the weights and biases obtained from the training that each new text from the test set is attributed to one of the authors of the training set.
In the proposed model, in order to increase the precision of classification, a larger data set named $Reuter\_50\_50$ containing 2500 texts for training and 2500 texts for testing are used for authorship identification, the features are extracted from the texts in the training and testing set, and then each text is represented as a vector of these features. In the proposed model, the lexical features (character-based and word-based), syntactic features (using punctuation marks and function words), and structural features are used; these cases are shown in Table (2) [29].

Table 2: Features used in the Proposed Model

Feature Type	Feature title
Lexical Features (Character based) (F1)	Total number of characters / Total number of letters Total number of characters / Number of capital letters Total number of characters /Number of numeric characters Total number of characters / Number of empty space characters
Lexical Features (word based) (F1)	Total number of words / Total number of words Number of characters/ Total number of words The number of words in length from 7 to 20 / Total number of words The number of words in length from 1 to 3 / Total number of words The total number of different words/ Total number of words Number of words used once / Total number of words Number of words used twice / Total number of words
Punctuation (Syntactic Features) (F2)	Total number of characters /Number of characters used (-) Total number of characters / Number of characters used (") Total number of characters /Number of characters used (&) Total number of characters /Number of characters used (')') Total number of characters /Number of characters used (')') Total number of characters / Number of characters used (,) Total number of characters / Number of characters used (.) Total number of characters / Number of characters used (/) Total number of characters / Number of characters used (;)
Function words (Syntactic Features) (F3)	Total number of sentences / Number of repetitive words The, A, An Total number of sentences / Number of short answers yes, no Total number of sentences / Number of pronouns Total number of sentences / Number of letters added Total number of sentences / Number of conjunctions Total number of sentences / Number of auxiliary verbs
Structural Feature (F4)	Total number of sentences

The task of the authorship identification can be addressed as follows. Let A be a set of authors, D be a set of texts in which each text is written by an author in A. $A = \{A_1, A_2, \dots, A_q\}$ is the set of author groups, $\{D_1, D_2, \dots, D_m\}$ is a collection of documents in the corpus, $V = \{t_1, t_2, \dots, t_n\}$ is a collection of vocabulary terms for analysis. Each term $t_i \in V$ is represented as a vector t_{ij}, i.e., $t_{ij} = \{t_{i1}, t_{i2}, \dots, t_{iq}\}$, where the dimension t_{ij} represents the term t_i weight on the author group A_j. Term weighting is an important concept in the modern information analysis. For word frequency, Eq. (20) is used [3]. Eq. (20) is a word frequency determination method suitable for word extraction due to low time complexion. In

Eq. (20), the parameter (t_n , d_i) is the frequency of each t_n feature in the document d_i.

$$w_{ni} = tf(t_n, d_i) = \begin{cases} (t_n, d_i) & t_n \in vector\,of \\ 0 & t_n \notin vector\,of \end{cases} \quad (20)$$

After extraction of features, normalization of the extracted features takes place. Because of the use of the MLP-ANN with the sigmoid activation function in the proposed model, the values of the features between zero and one are normalized using Eq. (21). Where x is the amount that needs to be normalized, d_L the smallest value of x, d_H‘ the maximum value of x, n_L, is less than the value to which it is normalized (in the proposed model is zero), n_H, the highest value to which it is normalized (In the proposed model it is equal to one).

$$f(x) = \frac{(x-d_L)(n_H-n_L)}{(d_H-d_L)} + n_L \quad (21)$$

After normalizing the structure of the ANN, the MLP-ANN is determined. In the proposed model due to the existence of 50 different authors, if we use an ANN with 50 neurons in the output layer, the time needed to classify will increase and thus the classification precision will decrease, hence, 10 ANNs with 5 neurons in the output layer has been used and these values are obtained through the error test.

In all ANNs, due to the extraction of 28 characters (features) from the texts, the number of neurons in the input layer was 28 and the number of neurons in the middle layer is considered equal to 50 neurons and the sigmoid activation function has been used in both hidden and output layers. We have used the proposed model of the learning networks with the observer and the model pattern method. MLP procedure is developed through the summation of each layer’s outputs by biases and applying transfer function in order to convert and send them to the next layer. Neurons in the hidden layer can use different active functions in order to produce the output. The most common ones are sigmoid activation functions, tangent sigmoid and linear active functions. In this paper, we have used sigmoid activation function that is calculated according to the Eq. (22).

$$\varphi(v) = \frac{1}{1+\exp(-av)} \quad (22)$$

In Eq. (22) $\varphi(v)$ is the value of sigmoid active function that is applied to the neurons in hidden layer in order to achieve the proper output.

We use the proposed model to get the least error value from the first condition, which is the stop after epoch in certain times, and we count the number of epochs of BP 900 times. In the proposed model, after determining the structure of the MLP-ANN, initialization of position and velocity vectors of particles with random values between 0 and 1 is performed. (Position vectors and velocity vectors to the total length of the weights and biases in the MLP-ANN) and initializing the fit function for each particle is carried out through the error value obtained from the feed-forward MLP-ANN. For each particle, the total network error (MSE) is calculated based on all training data and this error is considered as the initial value for the fit function for that particle. After initializing, in each epoch, the velocity, and position of each particle, which has actually the same weights and biases in the network, is calculated based on Eqs. (4 and 5), the total network error for each particle is calculated for new values of position and velocity for all training data and is considered as a fitness function for that particle. This value is compared with the best position of that particle (pbest). If the value of the fitness function is less than pbest, it replaces the value of pbest. Then, the particle is identified in the group with the best position (gbest) and its position is compared with the fitness function. If the value of the fitness function is less than gbest, it will replace gbest. These steps will continue until the stopping conditions of the algorithm (stopping after 20 times the epoch) are maintained. Once the algorithm stops, the best values are stored for weights and biases (gbest). Then, BP, based on the weights and biases derived from the PSO, begins to reduce the total ANN error in the training set and changes the weights and bias of the network until the network error is minimized.

These steps continue until the BP stopping condition (stop after repeated times) is fixed. Finally, the MLP-ANN starts labeling and categorizing the datasets of the test data according to the final values for weights and biases. Figure 2 shows the Flowchart of the proposed model.

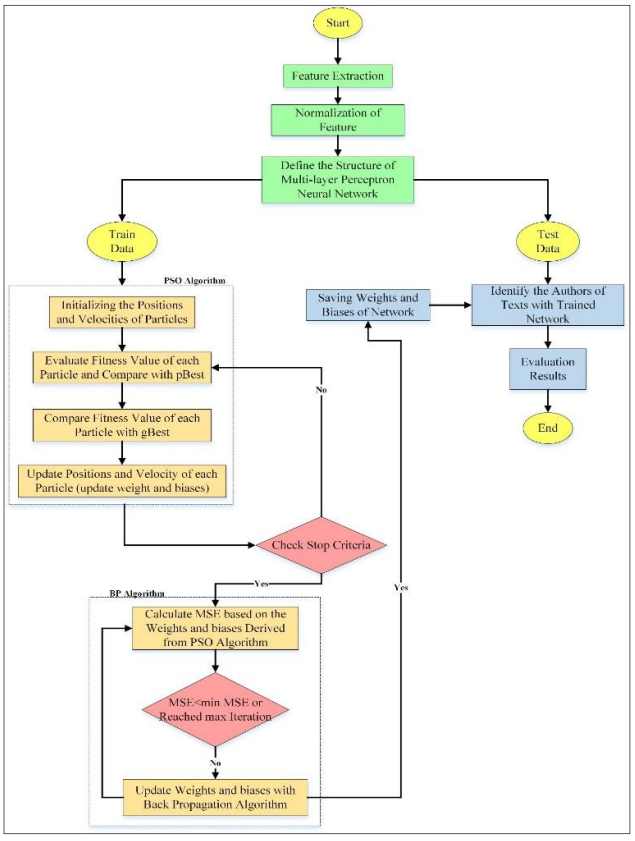

Fig. 2: Flowchart Proposed model.

To reduce randomness, the PSO is adopted to optimize the

weight and threshold coefficients of the BP ANN to avoid the local extremism and enhance the learning capability. The MSE of forecasting is used as the fitness function of PSO, which is defined as follows:

$$MSE = \frac{1}{N}\sum_{t=1}^{N}(\bar{X}(t) - X(t))^2 \quad (23)$$

Where $\boldsymbol{X(t)}$ represents the actual values, $\bar{\boldsymbol{X}}\boldsymbol{(t)}$ represents the predicted values and N represents the total number of data. In the first step of the PSO, the weights and thresholds are initialized. The fitness value of each particle is calculated by Eq. (23) and compared with the best solution. In the second cycle, the information of particles is updated by the PSO, and the next generation of particles is generated. The PSO continues searching the best solution until the target minimum MSE is reached.

4. Analysis and Discussion

In this section, the proposed model and other algorithms are applied to Reuter_50_50 datasets. This dataset is a subset of the Reuter text collection dataset and is publicly available at the University of California, Irvine (UCI) machine-learning repository. The dataset contains 5000 documents and 50 writers. The training corpus has 2,500 documents and test corpus has 2,500 documents (50 documents for each author in train and test) with an average of 500 words per document.

The assessment criteria used in our experiments are classified by 10-fold cross-validation, as used in the study [30].10-fold cross-validation is the technique by which the dataset is split into 10 subsets. Proposed model run for ten rounds; one subset is used for testing in each round, while the other nine are used for training. In each round, a different subset is taken for testing. This technique of assessment is deemed to be more efficient because each document is deemed to be forecast by its author, so each text is used precisely once in the test collection and nine times in the train collection for each collection of experiments.

To evaluate the proposed model, the precision criteria, which is computed using Eq. (24) and Recall (that is calculated by Eq. (25)), and therefore, the number of classified correct and incorrect datasets are used. To better understand the concept of the two precision and recall criteria, we use the confusion matrix described in Table (3), in which False Negatives (FN) represents the number of texts whose actual category is positive and its category classification algorithm is mistakenly shown as negative, False Positive (FP) represents the number of texts whose actual category is negative and its category classification algorithm is wrongly detected as positive. The True Positive (TP) represents the number of texts whose actual category is positive and the classification algorithm categorizes them correctly as positive, and True Negative (TN) represents the number of texts whose real category is negative and the classification algorithm categorizes them correctly as negative.

$$Precision = \frac{TP}{TP + \mathrm{FP}} * 100 \quad (24)$$

$$Recall = \frac{TP}{TP + \mathrm{FN}} * 100 \quad (25)$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} * 100 \quad (26)$$

Table 3: Confusion Matrix

Predicted Class		
Negative	Positive	Actual Class
FN	TP	Positive
TN	FP	Negative

Table (4) presents parameters of proposed model. The selection of parameters plays a crucial role in the optimization of the training process. The maximum number of epochs before the validation stops was set to 900, Number of neurons of input layer (28) and Number of neurons of output layer (50). The learning rate [0.1-0.9], for PSO the used parameters includes cognitive acceleration coefficient (c1: 1.3), social acceleration coefficient (c2: 1.3) and the number of particles used in the swarm (3000). Inertia weight linearly varied from 0.9 to 0.4 per grouping.

Table 4: Parameters of proposed model

Parameters	Value
Number of particles	3000
Inertia Weight(w)	(0.4,0.9)
V_{min}	-1
V_{max}	+1
X_{min}	-100
X_{max}	+100
C_1	1.3
C_2	1.3
Number of neurons of input layer	28
Number of neurons of output layer	50
Number of Epochs	100 - 900
Learning rate	0.1 - 0.9
momentum	0.1 - 0.9

4.1. Comparison of proposed model with BP for different momentum and learning rates

The results of the implementation of the proposed model and BP on the test dataset for the different learning is rated and it's 900 times epoch are shown in Tables 5 to 13. The results indicate that the proposed model in terms of performance and in terms of time factor is faster than BP. By increasing the amount of momentum and learning rate, the error rate and the number of incorrect texts categorized by the proposed model increased and the rate of precision and recall of the proposed model reduced. The best results are obtained for momentum = 0.1 and n = 0.1 at 900 times epoch. With BP, it was found that the time taken of simulation is about 47.79 second for Learning Rate (0.1) and momentum (0.1) and the MSE of testing error is

0.00001205. In contrast, time taken for simulation is about 22.89 second for Learning Rate (0.1) and momentum (0.1) and the MSE of testing error is 0.0000012 with proposed model.

Table (6) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9731, 0.9380, and 0.00001609 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9513, 0.8832, and 0.000029 respectively. The Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9992, 0.9992, and 0.0000012 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9627, 0.9384, and 0.0000163 respectively.

Table (7) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9513, 0.8832, and 0.000029 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9074, 0.8424, and 0.0000385 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9984, 0.9984, and 0.0000014 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9389, 0.9056, and 0.000029 respectively. From Table (7), it is clear that the proposed model shows higher performances than the BP in terms of MSE, Precision, and Recall.

Table (8) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9833, 0.9572, and 0.00001209 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9690, 0.9420, and 0.0000159 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9977, 0.9976, and 0.0000016 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9844, 0.9828, and 0.0000056 respectively.

Table (9) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9833, 0.9572, and 0.0000110 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9390, 0.8852, and 0.000029 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9980, 0.9980, and 0.0000015 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9144, 0.7848, and 0.0000541 respectively.

Table (10) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9789, 0.9528, and 0.0000125 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9221, 0.8344, and 0.0000425 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9976, 0.9976, and 0.0000016 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9719, 0.9552, and 0.0000125 respectively.

Table (11) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.9593, 0.9172, and 0.0000210 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9260, 0.8656, and 0.0000345 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9980, 0.9980, and 0.0000015 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9940, 0.8824, and 0.0000299 respectively.

Table (12) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.7933, 0.6584, and 0.0000860 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.8712, 0.7124, and 0.0000720 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9966, 0.9964, and 0.000018 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9824, 0.9804, and 0.000051 respectively.

Table (13) shows that the Precision, Recall, and MSE, on BP with n=0.1 are 0.8697, 0.7396, and 0.0000661 respectively. The Precision, Recall, and MSE, on BP with n=0.9 are 0.9150, 0.8160, and 0.0000470 respectively. Also, the Precision, Recall, and MSE, on proposed model with n=0.1 are 0.9597, 0.9684, and 0.0000084 respectively. The Precision, Recall, and MSE, on proposed model with n=0.9 are 0.9862, 0.9844, and 0.0000041 respectively.

4.2. Comparison of proposed model with BP based on epochs

By implementing the proposed model and BP for momentum = 0.1 and n = 0.1 in different epochs, different results were obtained that are shown in Table (14). The results show that with increasing the number of epochs, the error rate and the number of incorrect categories classified in the proposed model decreases, and the value of Precision and Recall criteria and the number of correct texts categorized in the proposed model increases. The results of various experiments showed that in all the epochs, the proposed model was better and faster than BP, indicating an improvement in the BP is based on PSO.

The epochs of the network reflect the performance of the algorithm. In order to compare the epoch's capability, a MSE threshold is established. When the algorithm reaches this threshold, it terminates learning and the number of iterations required is recorded, as shown in Table 14.

Table 5. Impact of Learning Rate on momentum = 0.1

Learning Rate	BP						Proposed Model					
	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2397	103	0.9849	0.9588	0.00001205	47.7917335	2498	2	0.9992	0.9992	0.0000012	22.8958667
n=0.2	2386	114	0.9804	0.9544	0.00001303	48.0327473	2482	18	0.9933	0.9928	0.0000021	23.016373
n=0.3	2372	128	0.9754	0.9488	0.00001401	48.7007855	2472	28	0.9896	0.9888	0.0000029	23.3503927
n=0.4	2368	132	0.9739	0.9472	0.00001408	47.9337416	2466	34	0.9874	0.9864	0.0000035	22.9668708
n=0.5	2360	140	0.9708	0.944	0.00001502	49.5228325	2450	48	0.9826	0.9850	0.0000041	22.7614162
n=0.6	2359	141	0.9705	0.9438	0.00001505	49.1808130	2451	49	0.9822	0.9804	0.0000054	22.5904065
n=0.7	2358	142	0.9701	0.9432	0.00001506	48.8467939	2242	258	0.9529	0.8968	0.000021	23.4233969
n=0.8	2162	338	0.9409	0.8648	0.00003105	49.4808301	2220	280	0.9410	0.888	0.000025	23.7404150
n=0.9	2158	342	0.9392	0.8632	0.00003109	50.0698638	2209	291	0.9314	0.8836	0.000029	24.0349319

Table 6. Impact of Learning Rate on momentum = 0.2

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2345	155	0.9731	0.9380	0.00001609	56.4472286	2498	2	0.9992	0.9992	0.0000012	26.2236143
n=0.2	2388	112	0.9813	0.9552	0.00001502	53.6520688	2493	7	0.9973	0.9972	0.0000015	25.8260344
n=0.3	2376	124	0.9767	0.9504	0.00001508	57.1962715	2474	26	0.9903	0.9896	0.0000027	27.59813575
n=0.4	2370	130	0.9745	0.948	0.00001601	57.736023	2454	46	0.9832	0.9816	0.0000038	26.8680115
n=0.5	2216	284	0.9545	0.8864	0.000028	01:00.4414571	2460	40	0.9858	0.984	0.0000033	29.5486269
n=0.6	2357	143	0.9698	0.9428	0.00001603	48.7147683	2301	199	0.9655	0.9204	0.0000194	21.35738415
n=0.7	2364	136	0.972	0.9456	0.00001608	58.3163355	2362	138	0.9652	0.9448	0.0000140	27.15816775
n=0.8	2357	143	0.9698	0.9428	0.00001603	01:01.0874940	2319	181	0.9571	0.9276	0.0000185	28.5693989
n=0.9	2208	292	0.9513	0.8832	0.000029	49.6768413	2346	154	0.9627	0.9384	0.0000163	23.83842065

Table 7. Impact of Learning Rate on momentum = 0.3

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2295	292	0.9513	0.8832	0.000029	54.9211413	2496	4	0.9984	0.9984	0.0000014	26.46057065
n=0.2	381	119	0.9786	0.9524	0.00001504	01:00.0634355	2387	113	0.974	0.9548	0.0000120	27.6555486
n=0.3	2372	128	0.975	0.9488	0.000016	55.1371536	2475	25	0.9905	0.99	0.0000029	25.5685768
n=0.4	2366	134	0.9729	0.9464	0.00001605	57.9243131	2310	190	0.9687	0.924	0.0000195	25.96215655
n=0.5	2216	284	0.955	0.8864	0.000028	01:01.3155070	2460	40	0.9855	0.984	0.0000041	28.527894
n=0.6	2310	190	0.9597	0.924	0.00001905	01:01.2815051	2454	46	0.9831	0.9816	0.0000054	29.12897455
n=0.7	2307	193	0.9584	0.9228	0.00001906	01:02.7385884	2353	147	0.9628	0.9412	0.0000136	27.6658849
n=0.8	2078	422	0.9388	0.8312	0.00004503	01:02.4415715	2449	51	0.9811	0.9796	0.000065	26.5487264
n=0.9	2106	394	0.9074	0.8424	0.0000385	49.6888420	2264	236	0.9389	0.9056	0.000029	22.844421

Table 8. Impact of Learning Rate on momentum = 0.4

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2393	107	0.9833	0.9572	0.00001209	53.2940482	2494	6	0.9977	0.9976	0.0000016	23.6470241
n=0.2	2374	126	0.9759	0.9496	0.00001302	54.0550918	2488	12	0.9955	0.9952	0.0000020	25.0275459
n=0.3	2227	273	0.959	0.8908	0.0000285	55.33441655	2496	31	0.9889	0.9876	0.0000042	26.667208275
n=0.4	2359	141	0.9704	0.9436	0.0000156	53.3670525	2362	138	0.9649	0.9448	0.0000145	24.68352625
n=0.5	2364	136	0.9725	0.9456	0.0000143	59.4834022	2463	37	0.9867	0.9852	0.0000041	27.7417011
n=0.6	2190	310	0.9445	0.876	0.0000320	01:03.2506177	2442	58	0.979	0.9768	0.0000062	29.7550045
n=0.7	2357	143	0.9694	0.9428	0.0000157	48.4787728	2199	301	0.9416	0.8796	0.0000310	22.2393864
n=0.8	2359	141	0.9702	0.9436	0.0000153	49.0678066	2296	204	0.9478	0.9184	0.0000210	21.5339033
n=0.9	2355	145	0.969	0.942	0.0000159	48.7457881	2457	43	0.9844	0.9828	0.0000056	23.37289405

Table 9. Impact of Learning Rate on momentum = 0.5

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2393	107	0.9833	0.9572	0.0000110	52.0079747	2495	5	0.998	0.998	0.0000015	10.186447025
n=0.2	2380	120	0.9782	0.952	0.0000135	52.6620121	2486	14	0.9947	0.9944	0.0000023	25.33100605
n=0.3	2368	132	0.974	0.9472	0.0000146	50.6048945	2464	36	0.9869	0.9856	0.0000041	23.30244725
n=0.4	2364	136	0.9728	0.9456	0.0000148	50.5638920	2374	126	0.9607	0.9496	0.0000135	24.281946
n=0.5	2360	140	0.9712	0.944	0.0000156	50.5268900	2452	48	0.9826	0.9808	0.0000059	24.263445
n=0.6	2356	144	0.9696	0.9424	0.0000159	50.3538801	2453	47	0.9829	0.9812	0.0000056	22.17694005
n=0.7	2349	151	0.9666	0.9396	0.00001608	53.7300732	2300	200	0.9492	0.92	0.000021	23.8650366
n=0.8	2310	190	0.9602	0.924	0.0000198	52.6410105	2351	149	0.9611	0.9404	0.0000150	25.32050525

n=0.9	2213	287	0.939	0.8852	0.000029	50.1008656	1962	538	0.9144	0.7848	0.0000541	24.0504328

Table 10. Impact of Learning Rate on momentum = 0.6

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2382	118	0.9789	0.9528	0.0000125	50.9539144	2494	6	0.9976	0.9976	0.0000016	23.4769572
n=0.2	2209	291	0.9349	0.8836	0.0000298	52.8720241	2339	161	0.9795	0.9356	0.0000172	25.43601205
n=0.3	1954	546	0.8824	0.7816	0.0000551	51.7019572	2357	143	0.9631	0.9428	0.0000156	21.8509786
n=0.4	2219	281	0.9555	0.8876	0.0000293	53.0840363	2474	26	0.9904	0.9896	0.0000035	24.54201815
n=0.5	2344	156	0.9645	0.9376	0.0000164	52.4089976	2458	42	0.9848	0.9832	0.0000046	23.2044988
n=0.6	2336	164	0.9616	0.9344	0.0000174	57.7430167	2360	140	0.9647	0.944	0.0000154	26.87150835
n=0.7	2270	230	0.9421	0.908	0.0000241	53.0190325	2450	50	0.9821	0.98	0.0000062	25.50951625
n=0.8	2134	366	0.9285	0.8536	0.0000373	52.6510114	2449	51	0.9818	0.9796	0.0000063	25.3255057
n=0.9	2086	414	0.9221	0.8344	0.0000425	50.0438624	2388	112	0.9719	0.9552	0.0000125	23.0219312

Table 11. Impact of Learning Rate on momentum = 0.7

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	2293	207	0.9593	0.9172	0.0000210	55.2911624	2495	5	0.998	0.998	0.0000015	25.6455812
n=0.2	2105	395	0.9161	0.842	0.0000398	53.6140666	2382	118	0.9545	0.9528	0.0000123	24.8070333
n=0.3	2356	144	0.9692	0.9424	0.00001508	49.2518171	2452	48	0.9825	0.9808	0.0000051	23.62590855
n=0.4	2349	151	0.9671	0.9396	0.00001601	51.3169352	2442	58	0.9789	0.9768	0.0000059	24.6584676
n=0.5	1918	582	0.9101	0.7672	0.0000594	50.9929167	2339	161	0.9565	0.9356	0.0000172	24.49645835
n=0.6	1915	585	0.8619	0.766	0.0000596	51.9689724	2456	44	0.9841	0.9824	0.0000049	24.9844862
n=0.7	2189	311	0.9274	0.8756	0.0000321	50.5518914	2440	60	0.9785	0.976	0.0000067	23.2759457
n=0.8	1806	694	0.8271	0.7224	0.0000699	50.9089119	2463	37	0.987	0.9852	0.0000039	22.45445595
n=0.9	2164	336	0.926	0.8656	0.0000345	50.5518914	2206	294	0.994	0.8824	0.0000299	24.2759457

Table 12. Impact of Learning Rate on momentum = 0.8

	BP						Proposed model					
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	1646	854	0.7933	0.6584	0.0000860	51.9779730	2491	9	0.9966	0.9964	0.000018	24.9889865
n=0.2	1853	647	0.8793	0.7412	0.0000651	52.2629893	2466	34	0.9874	0.9864	0.000037	24.13149465
n=0.3	2070	430	0.9349	0.828	0.0000440	54.6401252	2463	37	0.9865	0.9852	0.000039	26.3200626
n=0.4	2209	291	0.9522	0.8836	0.0000299	52.9870306	2465	35	0.9868	0.986	0.000038	22.4935153
n=0.5	1976	524	0.9119	0.7904	0.0000535	52.7240156	2388	112	0.9622	0.9552	0.00000120	23.3620078
n=0.6	1861	639	0.8671	0.7444	0.0000640	52.3039916	2468	32	0.9883	0.9872	0.000038	23.1519958
n=0.7	2131	369	0.9169	0.8524	0.0000370	59.8884254	2456	44	0.984	0.9824	0.000048	26.9442127
n=0.8	1950	550	0.8649	0.78	0.0000560	53.0040317	2474	26	0.9904	0.9896	0.000027	23.50201585
n=0.9	1781	719	0.8712	0.7124	0.0000720	48.3757665	2451	49	0.9824	0.9804	0.000051	22.18788325

Table 13. Impact of Learning Rate on momentum = 0.9

						BP						Proposed model
Learning Rate	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
n=0.1	1849	651	0.8697	0.7396	0.0000661	54.5181183	2421	79	0.9597	0.9684	0.0000084	25.25905915
n=0.2	1813	687	0.8917	0.7252	0.0000691	54.8271359	2470	30	0.9896	0.988	0.0000041	25.41256795
n=0.3	2211	289	0.9532	0.8844	0.0000293	52.1199810	2468	32	0.9885	0.9872	0.0000036	25.0598905
n=0.4	1809	691	0.8484	0.7236	0.0000699	52.5380050	2456	44	0.9847	0.9824	0.0000051	24.2690025
n=0.5	1858	642	0.8357	0.7432	0.0000652	53.3580520	2361	139	0.9678	0.9444	0.0000145	24.679026
n=0.6	1865	635	0.8546	0.746	0.0000645	53.8630808	2330	170	0.9762	0.932	0.0000182	24.9355404
n=0.7	1976	524	0.9181	0.7904	0.0000532	51.7309588	2477	23	0.9917	0.9908	0.0000025	24.8654794
n=0.8	2134	366	0.9276	0.8536	0.0000372	51.5199468	2298	202	0.9462	0.9192	0.0000210	23.8654794
n=0.9	2040	460	0.915	0.816	0.0000470	48.8857961	2461	39	0.9862	0.9844	0.0000041	22.44289805

Table 14: Different Epoch Effects in the Test Dataset

						BP						Proposed model
Epochs	True Count	False Count	Precision	Recall	MSE	Time(s)	True Count	False Count	Precision	Recall	MSE	Time(s)
100	1664	836	0.7948	0.6656	0.0000845	04.8492774	2283	217	0.9461	0.9132	0.0000221	1.4246287
200	2294	206	0.9611	0.9176	0.0000210	10.7216132	2450	50	0.9800	0.98	0.0000062	4.3508166
300	2296	204	0.9693	0.9184	0.0000215	15.6588956	2459	41	0.9897	0.9836	0.0000051	5.8284378
400	2345	155	0.9731	0.938	0.0000168	20.2131561	2495	7	0.9973	0.9972	0.0000018	9.10753805
500	2346	154	0.9738	0.9384	0.0000164	25.0354320	2496	5	0.9980	0.998	0.0000015	11.507316
600	2397	103	0.9849	0.9588	0.0000123	30.0167165	2497	4	0.9984	0.9984	0.0000014	15.00835825
700	2398	102	0.9853	0.9592	0.0000121	35.4160257	2497	3	0.9984	0.9988	0.0000013	17.56801285
800	2398	102	0.9853	0.9592	0.00001205	40.2473020	2498	2	0.9992	0.9992	0.0000012	20.124651
900	2397	103	0.9849	0.9588	0.00001205	47.7917335	2498	2	0.9992	0.9992	0.0000012	22.8958667

4.3 Classification accuracy

The evaluation measure used in our study is the accuracy. The experimental results for different feature types on benchmark Reuter_50_50 datasets are given in Table 15. The results clearly depict that the proposed model achieves the highest accuracy. The results for the comparison of the different feature types and techniques are summarized in Table 15 and Figure 3. Experimental results show that proposed model has 99.22% for F1+F2+F3+F4.

Table 15: Experimental results of BP and Proposed Model for Different Feature Types

different feature types	Accuracy	
	BP	Proposed Model
F1	95.23	98.14
F1+F2	95.78	98.49
F1+F2+F3	96.03	99.12
F1+F2+F3+F4	96.51	99.22

Figure 3 shows the experimental results of BP and the proposed model for four feature types.

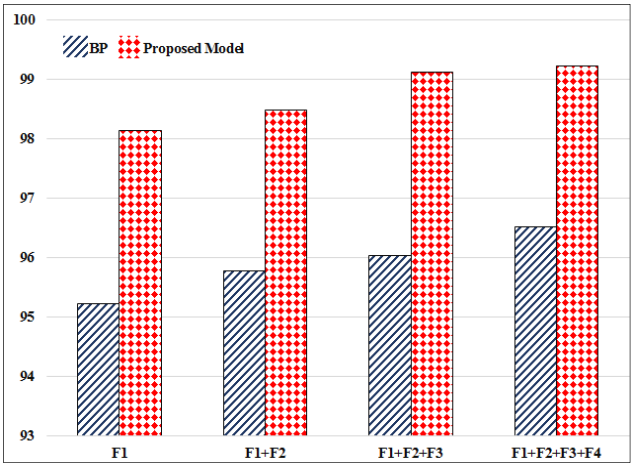

Fig. 3: Authorship Identification Accuracies for Different Feature Types.

Table 16 reveals the comparison of t-test values for Reuter_50_50 dataset based on proposed model and BP. This test shows that the proposed model outperforms in terms of accuracy. In this study, t-test is used to show the significance size of proposed model. Table 16 shows that proposed model is statistically significant.

Table 16: P-values of pairwise t-tests on accuracy using different feature types

different feature types	t-test	p-value
	Proposed Model	
F1 vs F1+F2	0.006	0.001
F1+F2 vs F1+F2+F3	0.002	0.031
F1+F2+F3 vs F1+F2+F3+F4	0.004	0.001

The paired t-test methods assess the means of two variables that stand for the same class at different times. The average values for the two variables are shown in the Paired Samples Statistics in Table 16. A low significance value for the t-test (typically less than 0.05) indicates that there is significance between the two variables. As it is seen in Table 16, the terms P-Value is (<0.05), which shows that proposed model obtained significant results.

4.4 Comparison of the proposed model with previous studies

Many studies have been conducted by researchers in the field of authorship identification of texts, each of which has examined various methods for various data. The results of comparing the proposed model with previous studies are shown in Table 17.

Table 17: Comparison of previous studies with the proposed model

Approach	Datasets	Precision
Delta [24]	Reuter_50_50	0.6700
KNN[24]	Reuter_50_50	0.6900
SVM[24]	Reuter_50_50	0.8500
SVM SMO[26]	Reuter_50_50	0.9170
SVM[27]	Reuter_50_50	0.9330
KNN[27]	Reuter_50_50	0.6150
BP	Reuter_50_50	0.9849
Proposed model	Reuter_50_50	0.9992

As shown in Table 17, the proposed model among different methods, in particular methods with the same dataset, has yielded better results and PSO has been able to greatly improve the results due to the fact that it is a global algorithm.

5. Conclusion and Future Works

Considering the importance of the problem of authorship identification of the texts, a new model based on improved BP with PSO in MLP-ANN has been presented in this paper. Since the PSO is a global algorithm and has a great amount of global capability, but in contrast to BP's ability to locate optimal localization, acts poorly in the global optimum. To solve the weaknesses of both algorithms, a new approach is presented in this paper. After extracting the lexical, syntactic, and structural features of the Reuter_50_50 datasets, we will train the proposed model based on the labeled examples. Then, at the testing stage, the trained model attempts to implement authorship identification of new texts and finally, using the criteria of the number of correct and incorrect data, Precision and Recall, the results of the proposed model are evaluated. The percentage of precision and recall obtained from the proposed model is 0.9992 and the values of these criteria in BP are 0.9849 and 0.9588, respectively. The results of the proposed model and its comparison with BP in the MLP-ANN show the efficiency, the speed, and precision of the proposed model, which are higher than that of the algorithm. It is possible to evaluate the efficiency of the proposed model for other factors such as extraction of N-gram characteristics, variation in the structure of MLP-ANN, and other algorithms can be used to train the MLP-ANN and optimize the results. Based on the simulation results, the proposed model is more precision than BP.

Therefore, it can conclude that the proposed model is a fast and accurate tool for designing problem of authorship identification.

Reference

[1] Zheng, R., et al. Authorship analysis in cybercrime investigation. in International Conference on Intelligence and Security Informatics. 2003. Springer.
[2] Kumar, P.J., G.S. Reddy, and T.R. Reddy, Document weighted approach for authorship attribution. Int. J. Comput. Intell. Res., 2017. 13(7): p. 1653-1661.
[3] Luhn, H.P., A statistical approach to mechanized encoding and searching of literary information. IBM Journal of research and development, 1957. 1(4): p. 309-317.
[4] Zhang, C., et al. Authorship identification of source codes. in Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint Conference on Web and Big Data. 2017. Springer.
[5] Tennyson, M.F. and F.J. Mitropoulos. A bayesian ensemble classifier for source code authorship attribution. in International Conference on Similarity Search and Applications. 2014. Springer.
[6] Rexha, A., et al., Authorship identification of documents with high content similarity. Scientometrics, 2018. 115(1): p. 223-237.
[7] Potha, N. and E. Stamatatos. An improved impostors method for authorship verification. in International Conference of the Cross-Language Evaluation Forum for European Languages. 2017. Springer.
[8] Posadas-Durán, J.-P., et al., Application of the distributed document representation in the authorship attribution task for small corpora. Soft Computing, 2017. 21(3): p. 627-639.
[9] Zhang, C., et al., Authorship identification from unstructured texts. Knowledge-Based Systems, 2014. 66: p. 99-111.
[10] Ramchoun, H., et al., Multilayer Perceptron: Architecture Optimization and Training. IJIMAI, 2016. 4(1): p. 26-30.
[11] Gharehchopogh, F.S. and E. Ahmadzadeh, Artificial neural network application in letters recognition for farsi/arabic manuscripts. International Journal of Scientific & Technology Research, 2012. 1(8): p. 90-94.
[12] Das, H., et al., A novel PSO based back propagation learning-MLP (PSO-BP-MLP) for classification, in Computational Intelligence in Data Mining-Volume 2. 2015, Springer. p. 461-471.
[13] Kawakami, K., Supervised sequence labelling with recurrent neural networks. Ph. D. thesis, 2008.
[14] Aghazadeh, M. and F. Soleimanian Gharehchopogh, A New Hybrid model of Multi-layer Perceptron Artificial Neural Network and Genetic Algorithms in Web Design Management Based on CMS. Journal of AI and Data Mining, 2018. 6(2): p. 409-415.
[15] Gharehchopogh, F.S., I. Maleki, and S.R. Khaze, A Novel Particle Swarm Optimization Approach for Software Effort Estimation. International Journal of Academic Research, 2014. 6(2).
[16] Gharehchopogh, F.S., I. Maleki, and S.R. Khaze, A new optimization method for dynamic travelling salesman problem with hybrid ant colony optimization algorithm and particle swarm optimization. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 2013. 2(2): p. 352-358.
[17] Bai, Q., Analysis of particle swarm optimization algorithm. Computer and information science, 2010. 3(1): p. 180.
[18] Rumelhart, D.E., G.E. Hinton, and R.J. Williams, Learning representations by back-propagating errors. Cognitive modeling, 1988. 5(3): p. 1.
[19] Craven, M.W. and J.W. Shavlik, Using neural networks for data mining. Future generation computer systems, 1997. 13(2-3): p. 211-229.
[20] De Vel, O., et al., Mining e-mail content for author identification forensics. ACM Sigmod Record, 2001. 30(4): p. 55-64.
[21] Zhao, Y. and J. Zobel. Effective and scalable authorship attribution using function words. in Asia Information Retrieval Symposium. 2005. Springer.
[22] Türkoğlu, F., B. Diri, and M.F. Amasyalı. Author attribution of turkish texts by feature mining. in International Conference on Intelligent Computing. 2007. Springer.
[23] Moghaddam, M.E. A Persian writer identification method based on gradient features and neural networks. in 2009 2nd International Congress on Image and Signal Processing. 2009. IEEE.
[24] Demir, N.M., Authorship Categorization With Neural Network. Southeast Europe Journal of Soft Computing, 2012. 1(2).
[25] Gazzah, S. and N.E.B. Amara. Writer identification using modular MLP classifier and genetic algorithm for optimal features selection. in International Symposium on Neural Networks. 2006. Springer.
[26] Nirkhi, M.S., Stylometric approach for author identification of online messages. Int. J. Comput. Sci. Inf. Technol, 2014. 5(5): p. 6158-6159.
[27] Howedi, F. and M. Mohd, Text classification for authorship attribution using Naive Bayes classifier with limited training data. Computer Engineering and Intelligent Systems, 2014. 5(4): p. 48-56.
[28] Nirkhi, S., R. Dharaskar, and V. Thakare, Authorship identification in digital forensics using machine learning approach. Int. J. Latest Trends Eng. Technol.(IJLTET), 2015. 5(1).
[29]Rexha, A., et al. Extending Scientific Literature Search by Including the Author's Writing Style. in BIR@ ECIR. 2017.
[30] Iqbal, F., et al., A unified data mining solution for authorship analysis in anonymous textual communications. Information Sciences, 2013. 231: p. 98-112.

Azadeh Salamzadeh received her MSc. degree in computer engineering from Islamic Azad University, Urmia Branch, IRAN in 2017. Her research interests are mainly in the field of Data Mining, Text Document Classification, and Information Retrieval.

Email: salamzadeh.azadeh@gmail.com

Farhad Soleimanian Gharehchopogh received his Ph.D. in computer engineering from Hacettepe University, Ankara, Turkey. He is an Asistant Professor at the Department of Computer Engineering in Islamic Azad University, Urmia Branch, Iran. His research focus is on Machine Learning, Natural Language Processing, Information Retrieval.
Email: bonab.farhad@gmail.com

Paper Handling Data:

Submitted: 12.02.2017
Received in revised form: 09.11.2019
Accepted: 09.28.2019
Corresponding author: Dr. Farhad Soleimanian Gharehchopogh
Department of Computer Engineering, Islamic Azad University, Urmia Branch, Iran