Swarm Intelligence Based Author Identification for Digital Typewritten Text

Abdul Rauf Baig
Dept. of Information Systems,
College of Computer & Information Sciences,
Al Imam Mohammad Ibn Saud Islamic
University (IMSIU), Riyadh, Saudi Arabia
raufbaig@ccis.imamu.edu.sa

Hassan Mujtaba Kayani
Dept. of Computer Science,
National University of Computer & Emerging Sciences (NUCES)
Islamabad, Pakistan
hasan.mujtaba@nu.edu.pk

Abstract—In this study we report our research on learning an accurate and easily interpretable classifier model for authorship classification of typewritten digital texts. For this purpose we use Ant Colony Optimization; a meta-heuristic based on swarm intelligence. Unlike black box type classifiers, the decision making rules produced by the proposed method are understandable by people familiar to the domain and can be easily enhanced with the addition of domain knowledge. Our experimental results show that the method is feasible and more accurate than decision trees.

Keywords—digital forensic; author identification; author verification; data mining; machine learning

I. INTRODUCTION

With the rapid proliferation of computers and networking during the last 20 years, our society has changed. Previously unknown issues have become a major concern. Cybercrimes is one such phenomenon. A crime committed with the help of computers and networking is called a cybercrime. The definition also includes those crimes that target the computer or network. Common examples of cybercrimes are fraud, identity theft, phishing, network intrusion, denial-of-service attacks, and spreading of computer viruses. One of the main causes of the rapid spread of cybercrimes is the anonymity of the users of Internet. Internet usage did not evolve with user identification as a concern. Unlike the users of many other commonly used systems, (e.g. banking, electricity, international traveling), individuals need not divulge their real identity in the cyberspace. Until our society is ready for some disciplinary measures, such as, unique identity of a person accessing the Internet (transmitted with each piece of communication), we have to find algorithmic ways of finding the identity of Internet users (particularly in situations where a cybercrime has been committed or being perpetuated).

The present research is focused on finding a method that can be used to detect the authorship of electronic text in a reliable and comprehensible (as opposed to black box) way. The key to authorship identification is hidden in the way different people write. It is a commonly observed phenomenon that people can detect when a friend suddenly write in a manner that is not his usual custom. Every person has a writing style that is usually stable across different specimens of his writing. This style can be represented by features, such as, choice and use of words, organization of sentences and paragraphs, and use of function words. These features can then be used for authorship identification.

In the current, we present a procedure for using these features for learning a classifier model and then use this model for identifying authors of unseen input text. Our classifier is based on a meta-heuristic known as Ant Colony Optimization (ACO). This meta-heuristic comes under the aegis of swarm intelligence. The main advantages of such a classifier are accuracy and comprehensibility. Unlike black box type classifiers, e.g. SVM, the decision making rules produced and used by an ACO classifier are understandable by experts of the domain.

The remaining paper is structured in the following manner. In Section 2, we present the fundamental concepts associated with authorship identification and give an overview of related work. Section 3 outlines our proposed approach. The detail of the dataset and feature set used in the experiments is given in Section 4. In the same section we present the experimental results and discuss them. In Section 5, conclusion of the present research effort is given and future research directions are presented.

II. FUNDAMENTALS AND RELATED WORK

Classification is one of the basic tasks in data mining [1]. Some of the popular and commonly used classification methods are decision trees, artificial neural networks, support vector machines (SVM), and k-nearest neighbor classifiers. Classification of text on the basis of its author (or in other words author identification) is the determination of the writer, given a closed set of writers and a sample of text. This area comes under the umbrella of stylometric studies [2].

978-1-4799-7620-1/15/$31.00 ©2015 IEEE
For classification, we need to first learn (or build) a classifier. For this purpose supervised data is required. Usually, a data sample is represented as a list of attribute-value pairs followed by the class label. The richness and appropriateness of the data attributes is one of the fundamental concerns. If an observed phenomenon is not well represented by the attributes, then the learned classifier will also not be able to correctly model that phenomenon. The same is true for authorship identification. We need to learn a classifier model from known samples of text of a given author and for this purpose we need a suitable representation that can capture the idiosyncrasies of the text. Such a representation of text is in the form of a set of attributes (called a feature set). Each attribute has a well-defined, finite number of values. If we have \( n \) predefined attributes (features), then a given sample of writing can be represented by an \( n \)-dimensional feature vector.

The choice and definition of the feature set is a crucial factor in attribute identification. Previous literature on this subject defines five categories of features, described as follows.

**Lexical features.** These are character based and word based features and they depict an author’s lexicon-related writing style. Also sentence length and vocabulary richness can be included. For example, a set of high-frequency words can be used or the monitoring of short words and usage of words beginning with a vowel can be done.

**Syntactic features.** These features are used to express the writing style of an author at the sentence level (e.g. punctuation and function words such as “while” and “upon”). They are often derived from an author’s habit of organizing sentences. Examples of relatively complicated syntactic features are passive count and part-of-speech tags. These features are often independent of the content of the sample.

**Structural features.** The root of these features is the manner in which a sample of writing is organized. Different authors have different styles related to their habit of structuring their writing. Examples of structural features are average length of paragraphs, and usage of indentation.

**Idiosyncratic features.** These features represent the writing habits of a writer. They include misspellings, grammatical mistakes, words shouted in CAPS and so forth.

**Content-specific features.** They relate to frequency of words in a specific topic (keywords). For example a discussion on sports will have different keywords than a discussion on habits of a writer. They include misspellings, grammatical habits of a writer. They include misspellings, grammatical

These 5 types of feature set are combined to form the global feature vector for representing an author’s piece of writing.

Most of the previous work uses a fixed feature set for all the data samples. However, in [3], the authors propose a Genetic Algorithm (GA) based method for discovering the optimal sub-set of features for the data of a given author from the complete set of features. A detailed discussion of previous work can be found in [4]. Three scenarios are considered: the profiling problem in which there is no candidate set of authors at all; the needle-in-a-haystack problem in which there are several possible authors and for each one of them there may be a paucity of writing samples; and the verification problem in which there is no closed set of candidate authors but we would like to verify if a certain known person is the author of a given writing sample. For each of these problems, it has been shown that data mining algorithms can be successfully adapted to provide a solution. Another recent survey on author identification for e-mails is [5].

Given a set of writing samples labelled along with the names of their authors, we can use supervised classifier learning methods (e.g. decision trees, artificial neural networks, and SVM) to generate a classifier model. This model can then be deployed to establish the class label of a previously unseen writing sample (i.e., in our case, establish the author of a given piece of writing). SVMs have been often used in research on authorship identification e.g. [6]. The manner in which a classifier model reasons to arrive at a decision is important for users in many applications. Classifiers such as artificial neural networks and SVMs are not useful in these situations because their processing logic is nontransparent and they are black boxes for the users. A decision tree is an example of a comprehensible classifier whose logic and reasoning is easily understandable by users. In fact any model based on classification rules has the benefit of easy user understandability.

Relatively new comers, the swarm intelligence algorithms [7] are a category of algorithms based on the utilization of a population of solutions, and have their foundations in the prototyping of the joint behavior of small, non-complex beings. One of popular swarm intelligence components is a meta-heuristic called ant colony optimization (ACO) [8]. ACO has been derived from the study of food foraging behavior of ants. After the ants find a food source, they soon discover an optimal path between that place and their ant hill. This is done by means of a secreted chemical (pheromone). This chemical is deposited by the ant on any path that it is traversing and the other ants are less likely to follow a path with zero or lower concentrations of pheromones. The ACO modeled from this behavior can be used as a potent method of heuristic search for combinatorial optimization. These include complex and difficult problems related to data mining, particularly classification [18].

Classification rule discovery algorithms based on ACO have given encouraging results in terms of accuracy and other metrics [19-23]. In this paper we use a recently proposed algorithm called AntMiner-CC [24] for building classifiers aimed at author identification. The algorithm builds classifiers with high accuracy and also has the advantage of automatically selecting a sub-set of features appropriate for the dataset being used.

III. DISCOVERY OF CLASSIFICATION RULES

A. Ant Colony Optimization

During the last 20 years or so, population based heuristic algorithms have gained a lot of popularity due to their capability of solving complex optimization problems. In such algorithms a population of solutions is generated and evolved. At termination the best solution among the evolved population
is retained as the algorithmic output. Genetic algorithm is an example of such algorithms. These algorithms have grown in numbers and can be categorized into different branches. One such branch is called swarm intelligence algorithms. A component group of swarm intelligence algorithms is the ACO algorithms based on a family of meta-heuristics. The meta-heuristic was proposed by Dorigo [8, 9] in early 1990. It models the behavior exhibited by ants while foraging for edibles.

Ants excrete a chemical, called pheromone, on the path that they are following. This chemical acts as an information signal for other ants. Other ants are more likely to take a trail with higher concentrations of pheromones. They themselves also spread pheromones, thus increasing its concentration. In this way, a path becomes popular and the other ants know that it will be worth their while to follow it. The pheromone decreases with the passage of time. Thus, if the number of ants traversing a path decreases, the path is likely to become unused after some time. This indirect signaling by pheromones helps a colony of ants to set up a feasible and short track between their ant hill and a source of food. Those ants that take a shorter path return sooner and the pheromone gets reinforced on that path as compared to a longer path. If the path becomes cluttered by an obstacle, the ants soon establish another efficient path by this mechanism. When the food source gets depleted, the path goes out of service.

The ACO meta-heuristic models some characteristics of this behavior of ants shown in quest for food. Suppose we have a problem whose solution can be put together as an amalgamation of constituent parts (e.g. a function of 5 variables: \( a^2 + b^2 + c^{1/2} + d + e^{1/3} \approx 25 \) whose correct solution is a combination of correct values of variables \( a, b, c, d, \) and \( e \)). In ACO paradigm, an artificial ant develops a solution by probabilistically selecting the solution components from the available options. Employing the vocabulary used for graphs, we designate vertices as feasible elements of the solution and the edges to be hypothetical links or paths between these elements. All edges have a pheromone value linked to them. An ant probabilistically chooses the components of a solution, one after another. The probability that the next component \( j \) will be selected after component \( i \) has been selected, is given by:

\[
P_j(t) = \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{j'=1}^{\text{total components}} x_j \{ \tau_{ij'}^\alpha \eta_{ij'}^\beta \}}
\]

where \( x_j \) is 1 if the component \( j \) has not previously been used in the current solution and otherwise 0. The powers of pheromone and heuristic values are given by the parameters \( \alpha \) and \( \beta \) and they govern the comparative significance of the pheromone and heuristic information.

The fitness or quality of the constructed solution is then calculated with the help of some performance measure. This fitness is used to calculate the pheromone value that is to be deposited on those components that contributed in the composition of the solution. A colony of ants constructs the solutions with the number of ants in the colony depending upon the problem being solved. The solutions are assessed and the pheromone is upgraded once all the ants have finished constructing their solutions.

We can have different policies for update of pheromones. One such policy is to update pheromone for each of the generated solutions. For an edge that links two components \( i \) and \( j \), the pheromone \( \tau_{ij} \) is modified as:

\[
\tau_{ij} \leftarrow (1 - \rho) \cdot \tau_{ij} + \frac{m}{\Delta T_{ij}}
\]

where \( \rho \) is a variable called rate of evaporation, and the number of ants is given by \( m \). The amount of pheromone deposited on an edge \((i, j)\) by an ant \( t \) is symbolized by \( \Delta \tau_{ij} \) and it is subjected to the fitness of the particular solution found by that ant.

The components found in a good solution have a greater probability of being selected again by ensuing ants. These are the components that have larger pheromone values and they frequently appear in the new solutions. If heuristic values can be assigned to the components then that is also factored in the probabilistic selection of the components. Normally, after a sufficient number of repetitive cycles, the ants are able to find an optimal, or at least a good enough, solution.

There are three things that are required for applying ACO to a given problem:

- ability to represent the solution to the problem as an amalgamation of discrete constituents.
- a procedure to evaluate the worth (i.e. quality) of the constructed solution (called fitness function).
- a heuristic function for evaluating the different elements of the solution (desirable but not necessary).

Currently, the most commonly used ACO algorithms are Ant System, MAX-MIN Ant System, and Ant Colony System [9]. Their primary dissimilarity is in the way new solutions are constructed (the manner in which probability is calculated) and the manner in which pheromone is updated.

Irrespective of the variant used, one pass of the algorithm consists of a colony of ant doing the following steps: solution construction, followed by its evaluation, and subsequent proportional update of pheromone values. The algorithm repeats these steps several times. In each of the iterations, the ants are guided by the experience obtained (in the form of pheromone values) in previous iterations. The output of the algorithm is the top-most solution constructed during any of the iterations and this solution is returned as the output (solution to the input problem).

ACO inherently provides a convenient solution for combinatorial optimization problems. From its genesis, ACO has been successfully applied to numerous problems [9]. Examples include travelling salesman, quadratic assignment, protein folding, job scheduling, routing of vehicles, set problems, network routing, power load dispatch, and bioinformatics. It has also been employed for discovery of classification rules. In this paper we present an ACO based
method for discovering classification rules governing author identification.

B. ACO Learning of Rule Based Classifier Model

The learning of rule based classifier models with ACO was first presented by Parpinelli, et al. [10] in the form of an algorithm called AntMiner. Since then several variants have been proposed, e.g. AntMiner2, AntMiner3 [11], and AntMiner+ [12]. In our experiments we use a recently proposed variant called AntMiner-CC [13], due to its promise of better results and short computation time. For the sake of brevity, we only provide a high level summary of AntMiner-CC. The search space for the ants is shown in Fig. 1. The interested reader is referred to [13] for more details.

![Fig. 1. Example graph used for searching by ants](image_url)

Given some data, the AntMiner-CC algorithm performs a heuristic search for learning rules that model that data. The rules discovered are of the following generic structure: “IF \(<term_1\> \text{ AND } \<term_2\> \text{ AND } \ldots \text{ THEN } \<\text{class}>>\)”. Each term is a couple comprising of an attribute and its value joined with the help of the relational operator “=”. (e.g. Shape = round, where “Shape” is the attribute and “round” is one of its many feasible values). If the data is real-valued (or continuous), then it has to be discretized before the application of the algorithm (pre-processing step).

AntMiner-CC is the type of classifier learning algorithms that are called sequential covering algorithms. Such algorithms remove those training samples from the complete training set that are correctly covered by the recently discovered rule, before attempting to discover the next rule. Correctly covered samples are those which match the rule (both antecedent and consequent are satisfied). The reduced training set is used to discover the next rule, and so on. The rule discovery procedure persists up to the point where the training set is completely devoid (or near to be empty) of training samples.

The ants construct several candidate rules (one rule per ant) and the best one from these candidates is selected to be appended in the list of discovered rules. For the construction of a rule, an ant proceeds in the following manner. First, it selects the rule consequent (one of the class labels from the class label set). Then it embarks upon its journey to construct the antecedent of the rule. It begins with zero terms and accumulates conditions one by one. The selection of a term to be appended to the incomplete rule is probabilistic and the terms with higher values of pheromone and heuristic have higher probability of being added.

The search by ant can be guided by the heuristic value of the different available options. In AntMiner-CC, the magnitude of the relationship between the chosen class label, the already chosen term, and the next candidate term is used as a heuristic. This usage of this heuristic has the side benefit of considerably reducing the search space and thus it speeds up the algorithmic search. The heuristic equation is:

\[
\eta_{k,i,j} = \frac{|term_i, term_j, class_k|^2}{|term_i, term_j|^2|term_j, class_k|}
\]

(3)

where \(\eta_{k,i,j}\) is the heuristic function determined by the number of data samples having the committed \(class_k\) and the already selected \(term\) and the candidate \(term\).

When a rule has been constructed, it is then slashed of extraneous terms (called pruning). The aim is to improve the accuracy of the rule, if possible. Even if accuracy is not improved, sometimes it is possible to reduce the size of the rule for the same accuracy. The fitness of the rule is determined after the pruning effort. The update of pheromone values is done on the basis of the fitness.

This process is then repeated and the construction of another rule is undertaken. Following the construction of a specified number of rules (each ant constructing one rule), the rule with the highest fitness is appended to the list of learned rules. The training set is depleted by trimming those training samples that are accurately predicted by the discovered rule.

Rules are continuously discovered in this manner and the list of discovered rules continues to expand until some criterion for exit is fulfilled and the algorithm stops. The ordered set of rules produced by the algorithm is a model of the input data and it can be used to predict the class labels of new, previously unseen data.

This approach, among other benefits, has the advantage of selecting a unique sub-set of the complete feature set for each individual author.

For using AntMiner-CC, each element of our feature set is considered a vertex and each author present in the dataset is represented as a class node.

IV. EXPERIMENTS AND RESULTS

This section describes the dataset and feature set used. We report our experiments and their results. The AntMiner-CC’s
performance is presented and compared with the results obtained from decision trees [14].

We have used the publically available predator dataset regarding cyber-aggression and predation [15-17]. We extracted several pieces of authorship (average 10) for 20 authors. The average length of a post is just over 10 sentences (approximate average of 70 words). The language used is English but it is of the type found on social networking sites with abbreviations and new words evolved from the concatenation of words. Repeated errors of all sorts can be found.

We use the feature set (Table I) defined by [6] in their research. There are 270 features. Eighty seven (87) of these are lexical features, 158 are syntactic features, 14 are structural features, and 11 are content-specific features. It has been experimentally shown [6] that these features combined together can adequately capture the distinct writing style of an author and thus they can be used for identification of authors. It may be useful to note that our classifier model learning algorithm automatically selects only those features from the complete feature set that are useful in accurately predicting the writing of an author.

<table>
<thead>
<tr>
<th>Features names</th>
<th>Number of features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total</td>
<td>270</td>
</tr>
<tr>
<td>Lexical</td>
<td>87</td>
</tr>
<tr>
<td>Syntactic</td>
<td>158</td>
</tr>
<tr>
<td>Structural</td>
<td>14</td>
</tr>
<tr>
<td>Content specific</td>
<td>11</td>
</tr>
</tbody>
</table>

The results generated by AntMiner-CC are compared with the results obtained for decision tree algorithm C4.5 [14]. We have implemented AntMiner-CC in C-Sharp language and we use the free available machine learning software called Weka [1] for C4.5 algorithm. The default parameters of the Weka tool were used while using C4.5.

It is important to select a performance metric for the experiments that is relevant. Classification algorithms can be evaluated using many different performance metrics. Many of these metrics are based on the number of prediction errors made by the classifier under study on unseen data. In our present experiments, predictive accuracy is retained as the main performance metric. It is a commonly used performance metric and very useful when a classifier is being evaluated or when a general comparison is being made between two classification algorithms. Predictive accuracy is simply the percentage of correct classification made by a classifier when presented with unseen testing samples.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of ants</td>
<td>1,000</td>
</tr>
<tr>
<td>Number of rules converged</td>
<td>10</td>
</tr>
<tr>
<td>Power of pheromone &amp; heuristic ($\alpha$, $\beta$)</td>
<td>1, 1</td>
</tr>
<tr>
<td>Pheromone evaporation rate ($\rho$)</td>
<td>0.1</td>
</tr>
</tbody>
</table>

The values of user defined parameters (Table II) used for the algorithm AntMiner-CC are: number of ants = 1000 ants, number of rules converged = 10, powers of pheromone and heuristic values ($\alpha$, $\beta$) are $\alpha = \beta = 1$ and pheromone evaporation rate $\rho = 0.1$.

The experimental results (Table III) show that our ACO based algorithm, AntMiner-CC, achieves a higher rate of accuracy (82.6% on the average) than the compared algorithm decision trees C4.5 (68.3% on the average).

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>AntMiner-CC</td>
<td>82.6%</td>
</tr>
<tr>
<td>C4.5</td>
<td>68.3%</td>
</tr>
</tbody>
</table>

V. Conclusion and Future Work

A swarm intelligence based method to learn author identification models has been proposed and presented. Such models are not only more accurate but also have the advantage of comprehensibility due to being rule based. Furthermore, the model learning algorithm automatically selects only a subset of the full feature set and ignores those that are not needed. Thus, during model learning, a two dimensional search is taking place: we are searching for the best feature sub-set in the space of feature set along with searching for a set of rules that represents the writing style of a given author.

The system can be applied for authorship detection of criminal or immoral e-mails, social networking texts, plagiarism check, etc.

This work can be expanded in several directions. One such direction is datasets with hierarchical and multi-label classes. Multi-label datasets are composed of data with multiple classes for a given sample. For example, a person can be a computer scientist and be a resident of New York and also play tennis. Hierarchical datasets also have multiple classes, but the classes are related to one another by a hierarchical structure. For example, a person can be a computer scientist, and an
employee, and a human being. Our method is capable of dealing with such data after suitable modifications.

Since our classifier is rule based, we can easily merge already known domain knowledge with the discovered knowledge. One straightforward way of doing this is to check for rules that are missing, redundant, and/or misleading. Another possibility is to include the hard and soft limitations imposed by our expert knowledge of the subject, directly in the classifier learning. Hard constraints can be incorporated by changing the search space to disallow impossible search possibilities. Soft constraints can be introduced by deliberate modification of the heuristic values.

This work can be expanded to include other languages, for example, Arabic. For Arabic, authorship detection for typewritten text may be easier as compared to handwritten text. Arabic handwritten text is difficult to interpret by automated means because of its cursive nature, but in type-written text there is no such problem. The bigger challenge is to identify a representative feature set. Each language has its own idiosyncrasies and the feature set used for English may not be appropriate for Arabic. Our method can be used to identify the more relevant features for any given language from a big, raw feature set of that language.

REFERENCES