CASIS-25 Authorship Attribution Challenge

Janzaib Masood
Department of Electrical and Computer Engineering
Auburn University
jzm0144@auburn.edu

***Abstract*—Authorship attribution systems are machines that help us discriminate in the writing patterns of different authors. It is fairly easy to build an authorship attribution system, but it is very challenging to make it secure in every way possible. The goal of this study is to build a secure authorship attribution framework that is not vulnerable to all types of cyber-attacks and fooling. We experiment with CASIS-25 dataset which has 4 sample texts of one author, and contains a total of 25 authors. We use an evolutionary feature selection method to find the most discrimiating features and use multiple models for the decision making that vote for the classification decision. These models are trained for robustness by our perturbed dataset derived form the original data.**

***Index Terms*—authorship attribution, feature selection, support vector machine, multilayer perceptron**

I. Introduction

An authorship attribution system (AAS) basically investigates the properties of the text written by different authors [1] [2] [3] [4]. The goal usually is the identification of the author of a message, letter or any text pattern. It can be used to check for plagiarism, forensics and several other applications. Due to the nature of the applications of these softwares, it is very important to make these models safe from all kinds of fooling.

In this project, we put our effort to build an secure AAS that is safe from the can do the right decision when confused by an attacker. We innvestigate the basic building blocks the appropriate measures that should be taken to produce effectively a secure AAS. We do our experiment with the CASIS-25 dataset. This dataset contains the text patterns of 25 authors (for every author there are four samples given).

We use a genetic and evolutionary feature selection method(GeFeS) to evolve a mask that reduces the size of our feature vectors for the model. The GEFeS [5] [6], used in the study is a steady state genetic algorithm. With a small feature length, it becomes easier and quicker to train the model and it helps get better accuracy with our less data. Initially we train only a multilayer perceptron, and support vector machines (with two different activation functions) as our classifers.

We find the most consistent features using the results of the GeFeS masking. We take our Casis-25 dataset and perturb the most important features in a stochastic fashion to generate a new dataset. The purpose of the our artificial dataset is to train the model for robustness. Using this perturbed dataset we train 5 new machine learning models that our adversary does not know of. It is harder for an adversary to attack a model, with an unknown architecture.

Section II gives an explaination of the techniques and strategies we use in our work. Section III describes the low level experimental details. Section IV shows the results and discusses why our model should perform well in the AAS challenge.

II. Methodology

A. Dataset

The dataset that we use in this work is called CASIS-25. It is composed on blog entries from 25 authors. For each author there are four writing samples, making it 100 writing samples altogether. The problem with this dataset set is only 4 examples for each author. Each example in the dataset is of length 95.

B. Feature Selection and Mask Design

Feature selection is the process of finding the important features that play an role in the classification. With better feature selection we can bring the focus of the classification algorithm to the right information and can work with less data. Here, we use a GeFeS as our feature selection method and uses a steady state genetic algorithm (SSGA) [5]. We use this GEFeS to generate feature masks which we used to turn on/off on certain features. The fitness function we use here supposed to help evolving feature mask by looking at the classifier accuracy. At the end of the GeFeS run, we get a mask that passes only the features that are important for the Casis-25 classification.

C. Preprocessing

The performance of machine learning models can be significantly improved if the input data is in a friendly format. For this purpose, we use three preprocessing steps given below.

a) Term Frequency-inverse Document Frequency (TFIDF): is a weight usually used in information retrieval and text data mining. It is a statistical metric, used to evaluate the significance of a word, in a document.The significance increases with the frequency of the word in the document but is offset by the frequency of the word in the corpus.

b) Standardization and Normalization: The standardizing of the features brings the value of each feature in dataset to a zero mean value and a standard deviation of one. And normalization brings the mean of each feature to zero and scales the dataset to have values between zero and one. The benefit of these two steps is the speedup of the convergence, when we train a classifier. Because in the search space, the data points tend to look more and more similar to each other and have a normal distribution.

D. Multi-model Classification Approach

We used trained 5 types of machine learning algorithms in this work for the classification purpose. The decision making is done by carrying our a voting between the decision of the classifiers. A description of each of these models is as follows.

a) Multilayer Perceptron: Multilayer perceptron was among the first architectures that came in the beginning of the neural networks era in 1950s. It was inspired by an approximate model of the working of the real human neuron in [**?**] in equation (6). Where, W_i is the weight of i_{th} input and Y is the output of the neuron.

$$Y = activation(W_0X_0 + W_1X_1 + ... + W_nX_n) \quad (1)$$

This neuron model would take a weighted sum of its inputs (X) and pass the output through a non-linear function (an activation function). So the weights and the activation function give a neuron its main properties. Using more of these neurons in the form of stacked layers where information flows from the first layer to the output layer makes a feed-forward neural network or a multilayer perceptron [**?**]. These models are usually trained using example data, but they can also be trained using evolutionary methods. In this study we use two activation functions, the rectified linear unit (a simple positive ramp function or an ideal diode curve for electrical engineers) in equation (7).

$$Relu(x) = max(0, x) \quad (2)$$

b) Support Vector Machines: Support Vector Machine (SVM) was proposed for the first time in 1979 by Vapnik [**?**]. SVMs represent a specific instance of an entire class of algorithms called kernel methods that map input vectors through a kernel method (e.g, a Gaussian kernel) to create more accurate classifications. The central idea at the core of an SVM is to translate all input vectors into a high dimensional kernel space where the instances of each class are more easily separated by a hyperplane [**?**]. So the goal always is get to a good hyperplane that helps us separate most the of function space. In this paper we used SVMs with two activation functions. SVM with linear activation is LSVM and the other with radial basis activation function is RBFSVM.

c) Decision Tree Classifier: is an algorithm that is expressed as a partition of the input example space [7]. It consists of nodes that make up a rooted tree and these nodes are connected by directed edges. The nodes that have an input coming into them but have no output edge are called the leaves. Each normal node splits the instance into two or more subspaces according to a discrete function of the input variables.

d) Naive Bayes Classifier: is a simple probabilistic classifier that uses Bayes theorem with strong (naive) independence assumptions. Another term for this model would be 'independent feature model'. In simple terms, a Naive Bayes (NB) classifier assumes that the presence or the absence of a particular feature of a class is unrelated to the presence or absence or any other feature. Depending on the precision of the probabilistic model, NB classifiers can be trained very effectively in a supervised setting [7].

e) Random Forest Classifier: is a supervised classification algorithm as its name implies, it creates a forest of trees and makes it random. There is a direct relationship between the number of trees in the forest and the random possibilites. With more number of trees it gives better accuracy. Working of Random Forest (RF) is simple, it randomly selects *K* features from the total *m* features. Among the *K* features, it calculates the *d* using the best split point. And it splits the node in children nodes using th best split and continues doing the spilting process until an end node is reached. The entire process is repeated *n* times to create *n* number of trees for the forest. Each tree acts as a subclassfier and a voting between the decisions of all the classifiers is performed to generate the classifier decision. It falls into the category of ensemble classifier, because the model contains classifier within itself.

E. Vulnerabilities and Counter Approaches

There are a number of ways in which an AAS can be failed, to generate wrong decisions. For example: the simplest that I can think of is changing the important features of a text. Probably, the easiest thing an adversary can exploit is the training data. They can put wrong information in the training data, if the model does online learning. The statistics of the training set also can be an easy tool to find the vulnerability of a trained model. The other thing the attackers can also exploit is the knowledge of a model's architecture. For example: the attacker can exploit the rectifying nature of RELU activation somehow. An attacker can also do probing with a lot of question/answer sessions with the model to know the decision boundaries and attack it. The attacker can also use a buffer layer or a fake cover on top of the model to confuse the user with wrong decisions.

Security in general is never complete; we just do our best to find the weaknesses and figure out ways to overcome them. Here are some examples: we can stop an attacker from probing our model, with help of some delay or limit the number forward passes from the same user. We should not share the entire information about the data we train the model with. A simple thing we can do is, not let the attacker know the architecture of the AAS. We could make use of different architectures and different classification models, so that it becomes harder to attack multiple models at the same time.

III. Experiments

We first pass the CASIS-25 dataset through a GEFeS to evolve a feature mask. The SSGA, evolves binary masks of length 95 and we apply these masks on our features. We allow the GEFeS to evolve at most for 350 evaluations and a population size of 30. The mutation rate is set to 2 percent (0.02) and we halt the evolution if the fitness value goes beyond 0.79. The evolved feature masks are applied on the

CASIS-25 vectors to reduce the select on relevant or important features. The fitness function first passes the feature vector through the preprocessing steps (TFIDF, Standardization and Normalization). Secondly, the three classification models are created. The preprocessed feature vectors are used to train the classifiers. For the classifier training and testing we apply 4 fold cross validation and calculate the accuracy. Finally, the fitness value of SSGA is finally this average classification accuracy of MLP, RBFSVM and LSVM model. At the end of the GEFeS run, we get an evolved feature mask. This mask helped us identify the features that were the most significant for knowing the author of a text sample. The above procedure was repeated for 30 runs to draw meaningful conclusions.

We calculated the mean and the standard deviation of the presence of each feature. The best feature masks that we recorded from the GEGeS runs in the first experiment were used here. The results were saved in a text file. The mean indicates the relevance or importance of each feature for the classification or discrimination (on a scale 0 to 1). Where as the standard deviation, shows how the importance of a feature can vary in the dataset. In order to see the importance of each feature, we focus on the standard deviation obtained in the previous experiment. The smaller the standard deviation, the better is the consistency of a feature in the dataset.

We take an intersection of the 30 feature masks of each classifier and then take a union of the resulting 3 masks. This final mask in equation 6 is used later. Where *M* denotes and *Mask* represents the final mask obtained.

$$M_{mlp} = intersection(M_1, M_2, M_3...M_{30}) \tag{3}$$

$$M_{rbfsvm} = intersection(M_1, M_2, M_3...M_{30}) \tag{4}$$

$$M_{lsvm} = intersection(M_1, M_2, M_3...M_{30}) \tag{5}$$

$$Mask = Union(M_{mlp}, M_{rbfsvm}, M_{lsvm}) \tag{6}$$

Now we create our artificial dataset from the original CASIS-25. We pick each example feature vector in the dataset and create its 24 perturbed copies. So we get a 100 examples of each author and a total of 2500 examples. The perturbation applies applies only on the features that are not clipped by the *Mask*. The *Mask* has a length of 80. So from the *Mask*, we randomly pick 5 features and replace them based on the consistency calculated previously (using the mean and standard deviation). Equation 7 tells the update rule of the 5 selected features in the Mask. Where, *Ri* is the consistency score of the feature *i* that has a value *Fi*.

$$F_i = round(R_i).rand(0, 1).abs(1 - F_i) \tag{7}$$

Now, we discard our previously built models and train 5 new models from scratch with the new data. This time we have more data, so our model should learn to classify with a better accuracy and recall. we do all our model training in python and use scikit-learn library for the for easy training and testing of the classification models. For the MLP model, we use 3 layers (of length 80, 50, 25). The first 2 layers have RELU activation and the output layer has softmax function applied on it. The two SVM models were applied as before (1 with linear and the other with radial basis activation function). A decision tree classifier, an RF classifier with 5 trees and NB classifier was used also used. The new dataset was shuffled and then split into 10 parts and 10 fold cross validation was applied to test the model.

Finally we perform voting between the 5 classification models' decisions to generate an output. In the voting scheme all the models are treated equally.

IV. Results

Table 1 shows the mean cross validation accuracy of LSVM, RBFSVM and MLP on the original dataset. Overall we have better performance in the presence of the preprocessing steps. So, we decide to keep them.

TABLE I
Effect of preprocessing on the 4 fold cross validation accuracy of the LSVM, RBFSVM and MLP

Mean Accuracy	**Classifiers**		
	LSVM	***RBFSVM***	***MLP***
Without TFIDF	0.7	0.62	0.678
Without Standardization	0.46	0.48	0.607
Without Normalization	0.51	0.52	0.599
With All	0.7	0.62	0.672

In Table 2 we show the mean and standard deviation of the best fitness value or classifier accuracy, achieved in each of the 30 GEFeS runs. We have also shown the mean vector length after applying the best evolved masks. We can see that the best accuracy is achieved for LSVM and the standard deviation is very low. The low standard deviation in the classifier accuracy is a good sign, it indicates that the models learn well in every run of the GEFeS. The also see that not all of the features are equally important. The mean feature length is around 75 for all three models; so these seem to be the features that help the classifier discriminate between the text samples of the authors.

TABLE II
Mean and standard deviation of the fitness accuracy and feature vector length achieved using the GEFeS

Mean GEFeS Scores	**Classifiers**		
	LSVM	***RBFSVM***	***MLP***
Mean Accuracy	0.7876	0.7209	0.7650
Standard Deviation of Accuracy	0.0154	0.0294	0.0
Mean Feature Vector Length	76	72	78

The mean and standard deviation of each feature is shown here in Fig 1. The mean feature values are shown by the blue line and the orange dotted line shows the standard deviation in importance of each feature. A mean value indicates, how important a feature is for the classification (it varies between 0 and 1). A zero mean indicates that a feature is useless in the classification and maximum mean indicates that the feature cannot be neglected in any way. The standard deviations of the features is also between 0 and 1. The standard deviation indicates, how consistent a features' importance for a classification task is. A standard deviation of zero means that features' importance always is same.

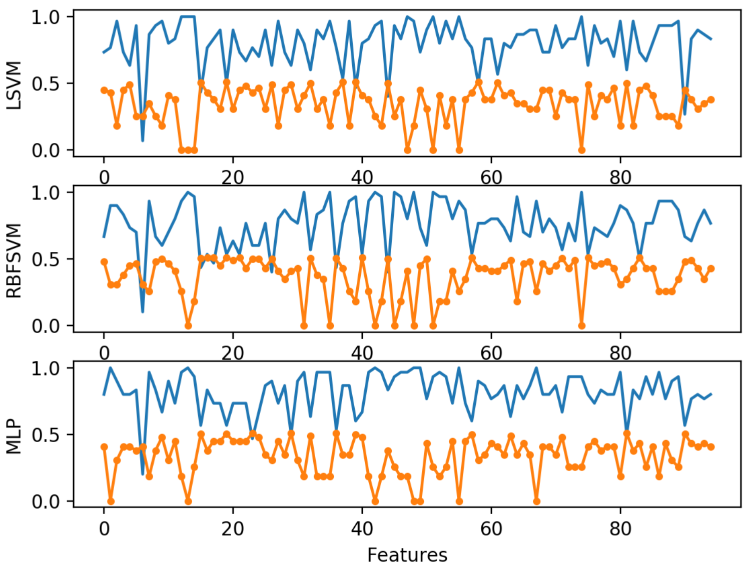

Fig. 1. Mean and standard deviations of the features (blue line shows mean curve and orange line shows standard deviations)

The features with the standard deviation of zero are the most consistent features. So overall, the features with maximum mean and minimum standard deviation are the most important for the analysis of text samples and the classifiers.

TABLE III
10-FOLD CROSS VALIDATION ACCURACY OF THE EACH CLASSIFICATION MODEL AND ALL SHOWS THEIR COMBINED PERFORMANCE.

	Classifiers						
	LSVM	***RBFSVM***	***MLP***	***DT***	***RF***	***NB***	***ALL***
Accuracy	0.85	0.87	0.86	0.79	0.92	0.79	0.85

Table III shows the improved accuracies of the robust models and the result of our voting procedure. Comparing with Table I we can see that there is significant improvement in the accuracy of the models compared to our previous project. Overall, we see an 85 percent cross-validation accuracy for our new AAS.

V. Conclusion

In this work, we put ourselves in a security and an attacker mindset, to make a secure AAS for the CASIS-25 dataset. We used guidance from the results of our previous GEFeS feature mask designing method, to decide on a final mask. Then we built a perturbed version of our original dataset, and trained 5 classification models on it. We saw a significant improvement in the accuracy of our model, as we trained with the new data. We use an ensemble procedure of voting to decide based on opinion of all the models. We are expecting a significant performance from our AAS, when it classify attacker's input data.

References

[1] "On the feasibility of internet-scale author identification. - Google Search."
[2] A. Narayanan, H. Paskov, N. Z. Gong, J. Bethencourt, E. Stefanov, E. C. R. Shin, and D. Song, "On the Feasibility of Internet-Scale Author Identification," in *2012 IEEE Symposium on Security and Privacy*, pp. 300–314, May 2012. ISSN: 2375-1207, 1081-6011, 1081-6011.
[3] C. Faust, G. Dozier, J. Xu, and M. C. King, "Adversarial authorship, interactive evolutionary hill-climbing, and author CAAT-III," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–8, Nov. 2017.
[4] T. Neal, K. Sundararajan, A. Fatima, Y. Yan, Y. Xiang, and D. Woodard, "Surveying Stylometry Techniques and Applications," *ACM Comput. Surv.*, vol. 50, pp. 86:1–86:36, Nov. 2017.
[5] L. D. Davis, *Handbook Of Genetic Algorithms*. New York: Van Nostrand Reinhold, 1st edition ed., Jan. 1991.
[6] M. Mitchell, "L.D. Davis, Handbook of Genetic Algorithms," p. 6.
[7] "Neural Networks and Learning Machines, 3rd Edition."