Adversarial Authorship with GatorCAAT: Probing via an EC

Rahul Alapati
Department of Computer Science and Software Engineering
Auburn University
rza0037@auburn.edu

Sutanu Bhattacharya
Department of Computer Science and Software Engineering
Auburn University
szb0134@auburn.edu

***Abstract*—Over the last few years, we have seen an increase in the development of Authorship Attribution systems, that have the potential to track users based on their writing style. Most of the machine learning models trained for Authorship Attribution, are vulnerable to adversarial attacks i.e. malicious inputs modified to yield erroneous model outputs, while appearing unmodified to human observers. However, all the adversarial attacks require knowledge of either the model internals or its training data. In this paper, we discuss a method for providing the required knowledge for adversarial attacks: Probing a machine learner via an EC. With probing, a user will be able to determine the behavior of the machine learner and also the set of feature vectors required for a successful attack.**

Keywords—Adversarial Authorship, GatorCAAT, Probing, Evolutionary Computation, CASIS-25, SEC Sports Writers

I. INTRODUCTION

The field of authorship attribution has been used to great effect by historians and literary detectives to identify the authors of the Federalist Papers, Civil War letters, and Shakespeare's plays. The use of machine learning techniques in authorship attribution has contributed to literary and historical breakthroughs. However, these machine learning models are prone to adversarial attacks, motivated by the desire to protect anonymity and privacy in a variety of scenarios, which have a devastating effect on the classification accuracy of the Authorship Attribution systems [1].

The goals of the adversarial attacks are to create a forgery of another author by imitating his writing style, and to mask the original author. By imitating the writing style of an author, the adversary will be able to fool the authorship attribution system into believing that the writing sample was written by a targeted author, who didn't write it. In masking, the adversary takes a writing sample and modifies it in order to conceal the true authorship [2].

However, all these adversarial attacks require prior knowledge of the underlying model's behavior or its training data. In order to obtain such knowledge, we probe the machine learner using a genetic algorithm (GA). The genetic algorithm is used to evolve a population of feature vectors that can be used to probe the machine learner to determine the type of feature vectors required for a successful attack and also to determine the behavior of the machine learner.

The organization of the paper is as follows: In section II, methodology is elaborated. In section III, the experiments and results are discussed in detail. Section IV ends the paper with discussion on the breakdown of work among the two authors.

II. METHODOLOGY

A. Model Generation

Initially, before we start probing, we need to save the best models, trained as a part of Assignment-3 [3]. In Assignment-3, we generate three real coded feature masks each for CASIS-25 and SEC Sports Writers [4-13] datasets using three real coded genetic and evolutionary feature selection (GEFeS) algorithms namely Steady State GA, Elitist GA and Elitist Estimation of Distribution Algorithm (EDA). Now, using these real coded feature masks, we train our Linear Support Vector Machine (LSVM) algorithms and save them using pickle [14].

B. Probing via an EC

To acquire the knowledge corresponding to the underlying model's behavior and its training data, we have used the following Genetic Algorithms:

1) Steady State Genetic Algorithm

The following is the Steady State GA, which is used to evolve the population of feature vectors required for probing:

Step 1. Initialize an initial population of 25 real coded feature vectors. Each feature vector consists of 95 values between a specified range.

Step 2. Evaluate the fitness of the 25 feature vectors with the squared difference of the decision functions as its evaluation metric. The squared difference is calculated between the decision function of the trained LSVM model and the desired decision function.

The decision function of the trained LSVM model is obtained using the decision_function( ) method of the Scikit Learn python library [15]. The desired decision function can be created by making the selected authors as 1 and others as 0.

For example, desired decision function for exactly one author (Author 0) can be created as follows:

1	0	…	0

Desired decision function for exactly two authors (Author 0 and 24) can be created as follows:

1	0	…	1

Desired decision function for exactly three authors (Author 0, 12 and 24) can be created as follows:

1	0	…	1	…	1

Step 3. For 4975 evaluations do the following:

a. Select two parents using Tournament Selection.
b. Create a child using Uniform Crossover of the two parents.
c. Mutate the child with a mutation rate of 0.05. In this step, we generate a random number and apply Gaussian mutation with mean 0 and standard deviation 1 on the child, if the random number is less than or equal to 0.05.
d. Evaluate the fitness of the mutated child as described in Step 2.
e. Replace the worst feature vector i.e. the feature vector with maximum squared difference in the population with the child.

At the end of 5000 evaluations, we evolved a population consisting of 25 feature vectors which can be used to probe the machine learner.

2) Elitist Genetic Algorithm

The following is the Elitist GA, which is used to evolve the population of feature vectors required for probing:

Step 1. Initialize an initial population of 25 real coded feature vectors. Each feature vector consists of 95 values between a specified range.

Step 2. Evaluate the fitness of the 25 feature vectors with the squared difference of the decision functions as its evaluation metric. The squared difference is calculated between the decision function of the trained LSVM model and the desired decision function as described in the Step 2 of SSGA above.

Step 3. For 4975/24 iterations do the following:

a. Repeat until 24 children are created
    a. Select two parents using Tournament Selection.
    b. Create two children using Uniform Crossover.
    c. Mutate the children with a mutation rate of 0.05. In this step, we generate a random number and apply Gaussian mutation with mean 0 and standard deviation 1 on the child, if the random number if less than or equal to 0.05.
    d. Evaluate the fitness of the mutated child as described in Step 2.
b. Keep the feature vector with minimum squared difference and replace the rest 24 feature vectors with the newly created children.

At the end of 5000 evaluations, we evolved a population consisting of 25 feature vectors which can be used to probe the machine learner.

3) Elitist Estimation of Distribution Algorithm

The following is the Elitist EDA, which is used to evolve the population of feature vectors required for probing:

Step 1. Initialize an initial population of 25 real coded feature vectors. Each feature vector consists of 95 values between a specified range.

Step 2. Evaluate the fitness of the 25 feature vectors with the squared difference of the decision functions as its evaluation metric. The squared difference is calculated between the decision function of the trained LSVM model and the desired decision function as described in the Step 2 of SSGA above.

Step 3. For 4975/24 iterations do the following:

a. Select twelve parents using Tournament Selection.
b. Create a probability distribution function (PDF).
c. Create 24 children by sampling from the PDF 24 times. Here we a create a child using 12-parent uniform crossover.
d. Mutate the children with a mutation rate of 0.05. In this step, we generate a random number and apply Gaussian mutation with mean 0 and standard deviation 1 on the child, if the random number if less than or equal to 0.05.
e. Evaluate the fitness of the mutated children as described in Step 2.
f. Keep the feature vector with minimum squared difference and replace the rest 24 feature vectors with the newly created children.

At the end of 5000 evaluations, we evolved a population consisting of 25 feature vectors which can be used to probe the machine learner.

In all the methods described above, the objective is to minimize the squared difference. Also, we have recorded the fitness i.e. the squared difference over the 5000 evaluations and plotted the online performance of the three EC probers.

III. Experiments and Results

Here, we probed the trained LSVM model to determine what type of feature vectors will cause exactly one, two and three authors to be associated with a feature vector. The minimum and average squared differences of the three methods over 10 runs are reported in the tables below.

Table 1: Squared Difference on CASIS-25 Dataset

Method	No. of Authors	Avg. Squared Difference	Best Squared Difference
SSGA	1 Author	2.12	2.71
	2 Authors	2.06	2.91
	3 Authors	2.07	3.14
EGA	1 Author	2.84	4.76
	2 Authors	2.75	4.39
	3 Authors	2.55	4.62
EEDA	1 Author	1.92	2.46
	2 Authors	1.85	2.41
	3 Authors	1.32	2.57

Table 2: Squared Difference on SEC Sports Writers Dataset

Method	No. of Authors	Avg. Squared Difference	Best Squared Difference
SSGA	1 Author	0.99	2.17
	2 Authors	1.79	2.34
	3 Authors	1.53	2.04
EGA	1 Author	0.71	1.46
	2 Authors	0.84	1.38
	3 Authors	1.01	1.61
EEDA	1 Author	0.84	1.17
	2 Authors	0.74	1.06
	3 Authors	0.95	1.37

The following are the online performance plots for the three probers in the best run:

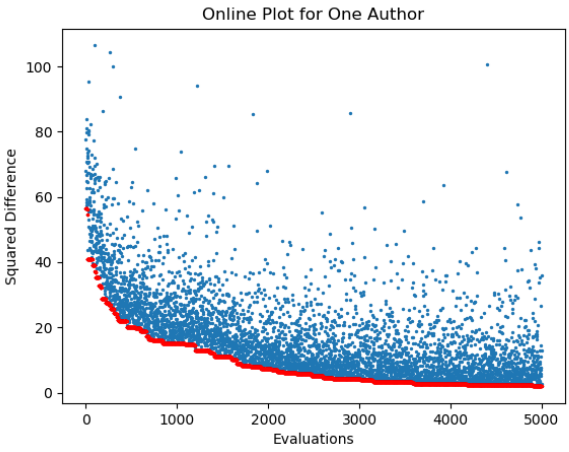

Figure 1: Online Performance of SSGA prober for 1 Author over CASIS-25 Dataset

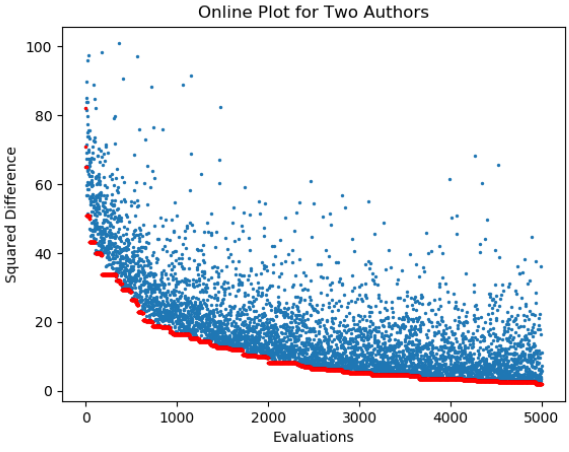

Figure 2: Online Performance of SSGA prober for 2 Authors over CASIS-25 Dataset

Figure 3: Online Performance of SSGA prober for 3 Authors over CASIS-25 Dataset

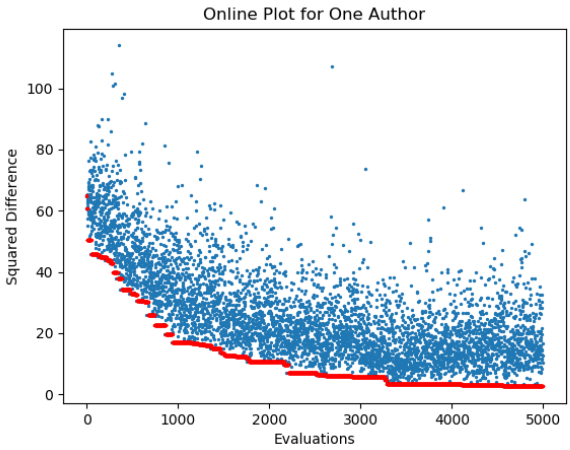

Figure 4: Online Performance of EGA prober for 1 Author over CASIS-25 Dataset

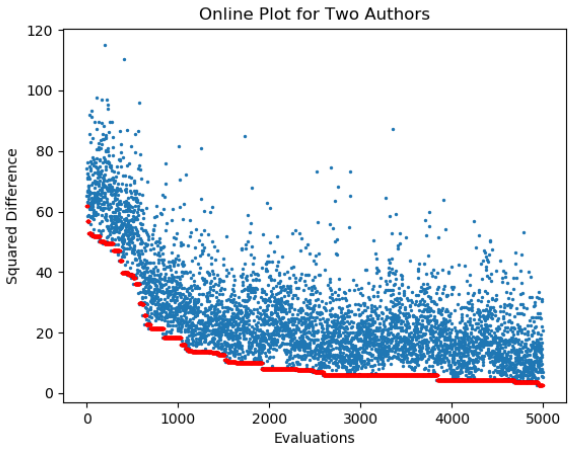

Figure 5: Online Performance of EGA prober for 2 Authors over CASIS-25 Dataset

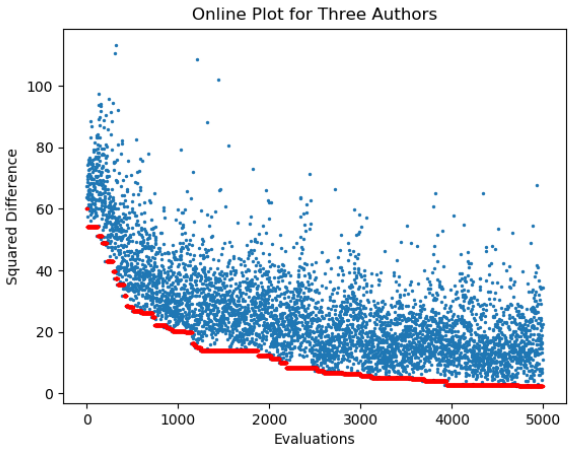

Figure 6: Online Performance of EGA prober for 3 Authors over CASIS-25 Dataset

Figure 9: Online Performance of EEDA prober for 3 Authors over CASIS-25 Dataset

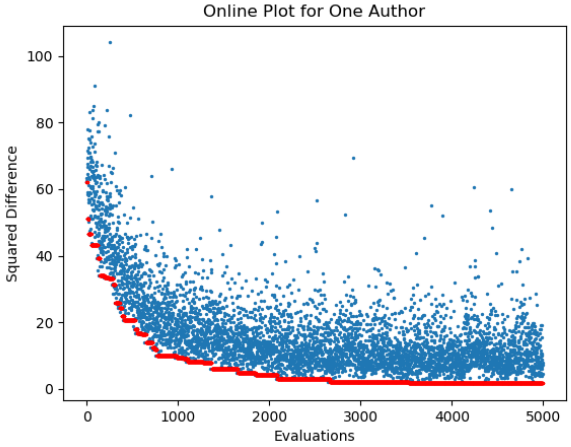

Figure 7: Online Performance of EEDA prober for 1 Author over CASIS-25 Dataset

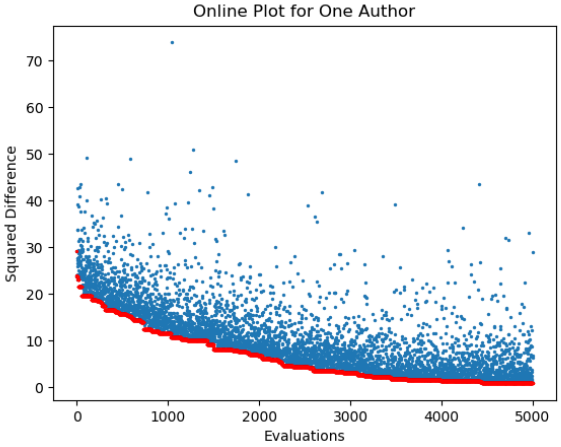

Figure 10: Online Performance of SSGA prober for 1 Author over SEC Sports Writers Dataset

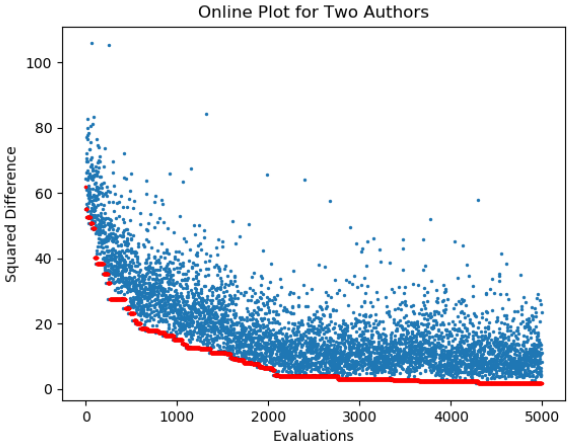

Figure 8: Online Performance of EEDA prober for 2 Authors over CASIS-25 Dataset

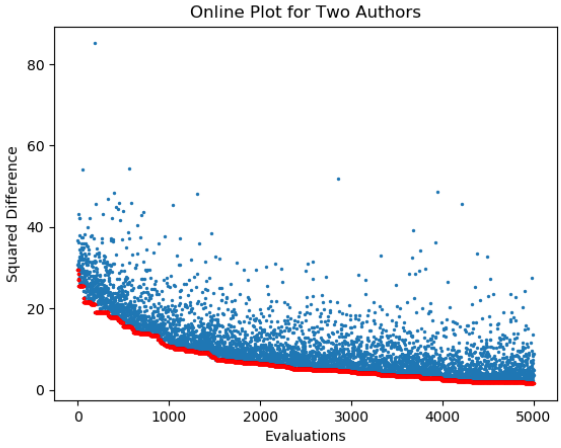

Figure 11: Online Performance of SSGA prober for 2 Authors over SEC Sports Writers Dataset

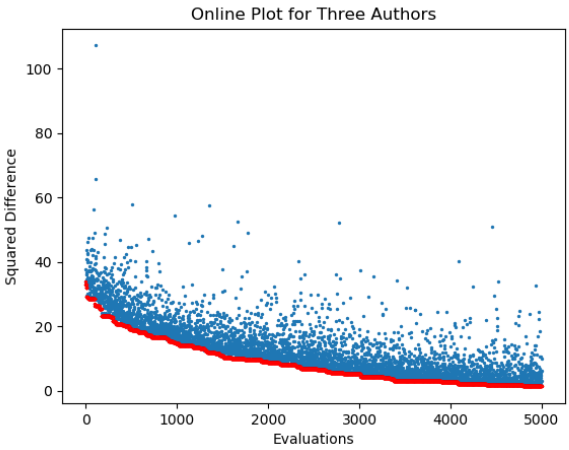

Figure 12: Online Performance of SSGA prober for 3 Authors over SEC Sports Writers Dataset

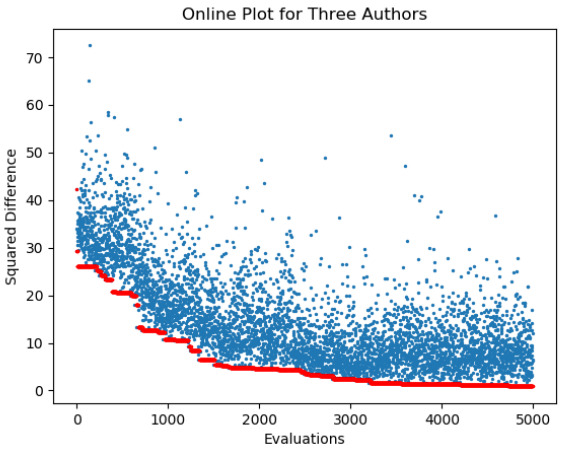

Figure 15: Online Performance of EGA prober for 3 Authors over SEC Sports Writers Dataset

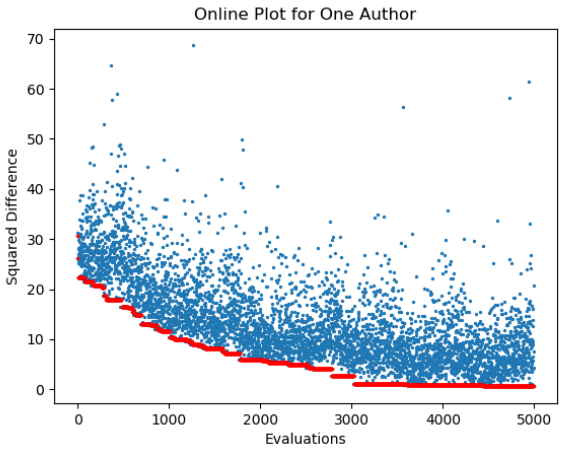

Figure 13: Online Performance of EGA prober for 1 Author over SEC Sports Writers Dataset

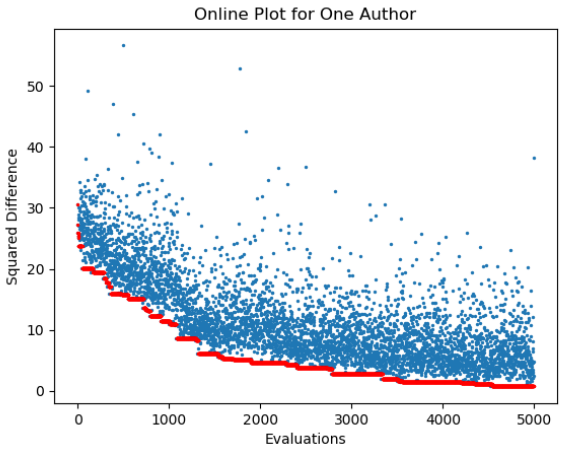

Figure 16: Online Performance of EEDA prober for 1 Author over SEC Sports Writers Dataset

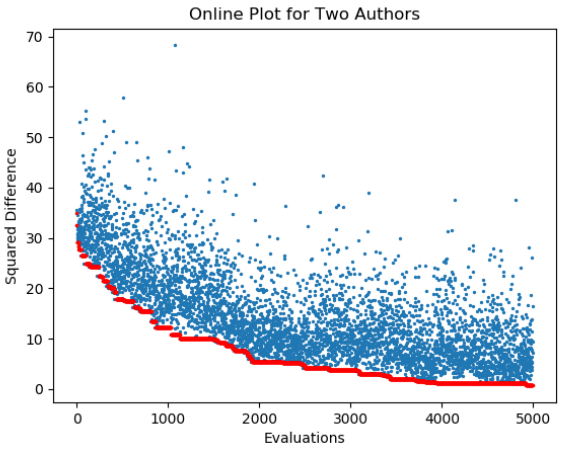

Figure 14: Online Performance of EGA prober for 2 Authors over SEC Sports Writers Dataset

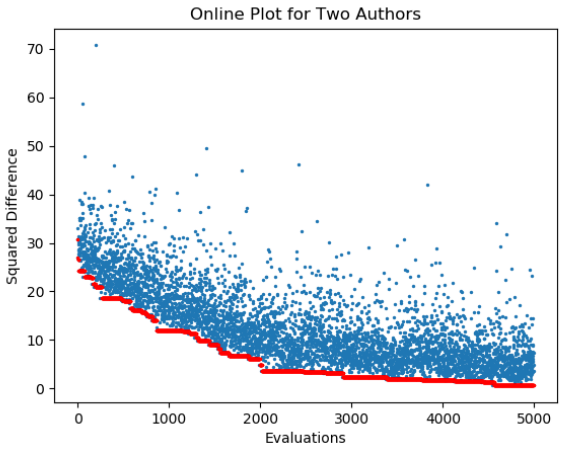

Figure 17: Online Performance of EEDA prober for 2 Authors over SEC Sports Writers Dataset

Figure 18: Online Performance of EEDA prober for 3 Authors over SEC Sports Writers Dataset

In all the figures above the red color indicates the best performing feature vector with minimum squared difference and the blue color indicates the fitness of the feature vector.

In all the three probers shown above, the initial population has been generated using the range of feature values generated after the tf-idf, standardization and normalization are applied (usually the ranges are between -1 to 1).

Alternatively, we generated the initial population using the range of feature values generated after the tf-idf, standardization, normalization and feature mask are applied (usually the ranges are between -3 to 4). The minimum and average squared differences of these three methods over 10 runs are reported in the tables below.

Table 3: Squared Difference on CASIS-25 Dataset

Method	No. of Authors	Avg. Squared Difference	Best Squared Difference
SSGA	1 Author	1.25	2.20
	2 Authors	1.35	2.48
	3 Authors	1.44	3.07
EGA	1 Author	3.19	4.66
	2 Authors	2.88	3.97
	3 Authors	2.95	4.80
EEDA	1 Author	1.53	2.55
	2 Authors	2.42	2.82
	3 Authors	1.80	2.55

Table 4: Squared Difference on SEC Sports Writers Dataset

Method	No. of Authors	Avg. Squared Difference	Best Squared Difference
SSGA	1 Author	0.90	1.66
	2 Authors	1.18	1.85
	3 Authors	0.71	1.83
EGA	1 Author	0.83	1.59
	2 Authors	0.50	1.17
	3 Authors	0.97	1.49
EEDA	1 Author	0.74	1.06
	2 Authors	0.87	1.18
	3 Authors	0.76	1.34

The following are the online performance plots for the three probers in the best run:

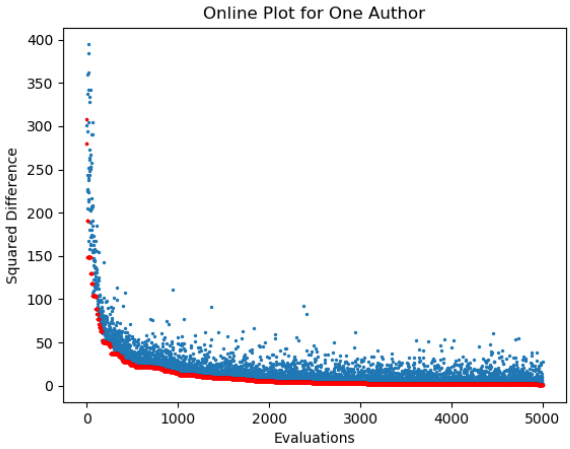

Figure 19: Online Performance of SSGA prober for 1 Author over CASIS-25 Dataset

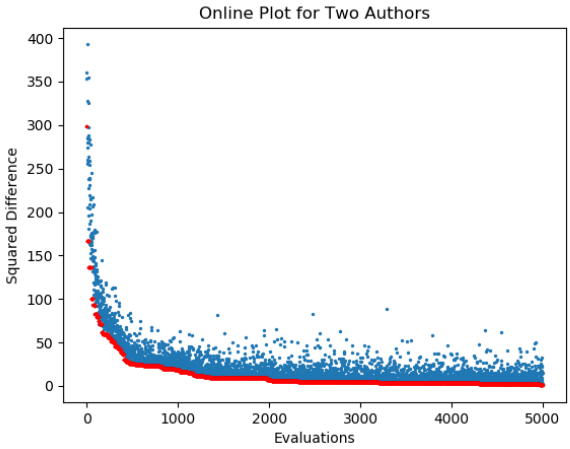

Figure 20: Online Performance of SSGA prober for 2 Authors over CASIS-25 Dataset

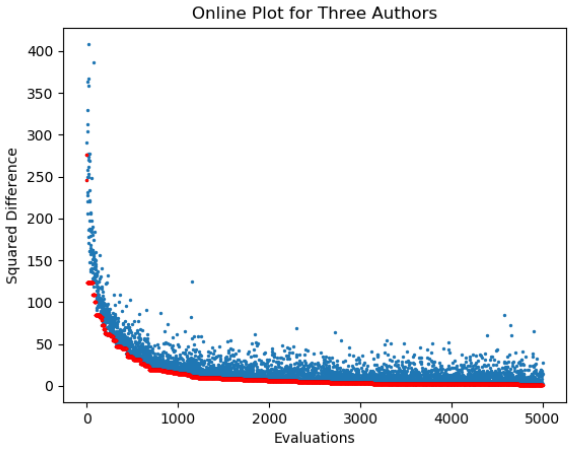

Figure 21: Online Performance of SSGA prober for 3 Authors over CASIS-25 Dataset

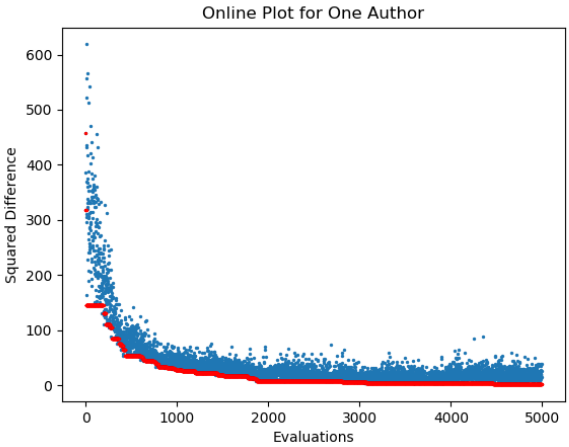

Figure 22: Online Performance of EGA prober for 1 Author over CASIS-25 Dataset

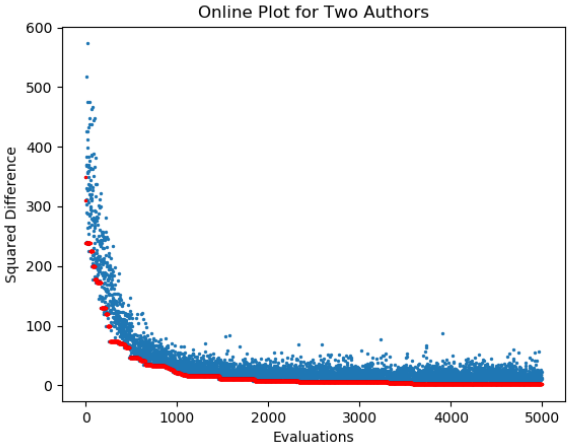

Figure 23: Online Performance of EGA prober for 2 Authors over CASIS-25 Dataset

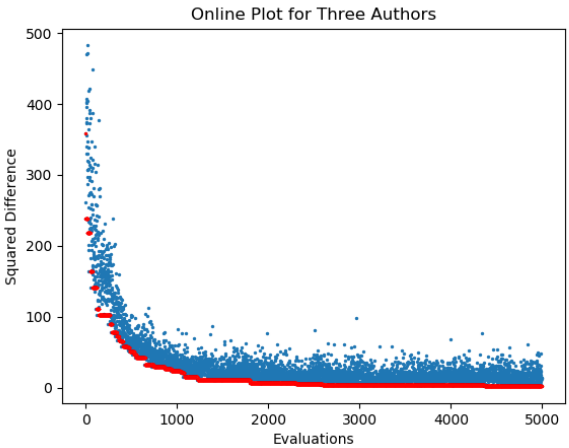

Figure 24: Online Performance of EGA prober for 3 Authors over CASIS-25 Dataset

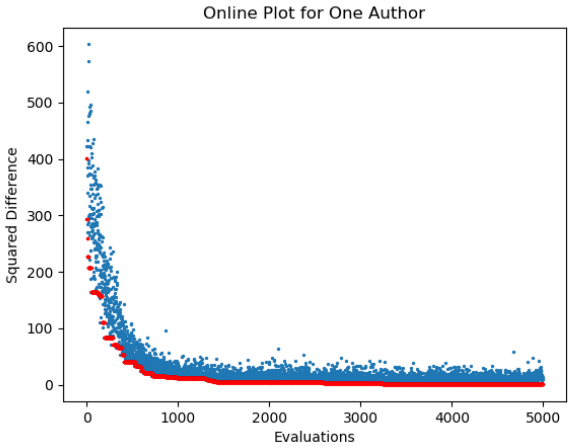

Figure 25: Online Performance of EEDA prober for 1 Author over CASIS-25 Dataset

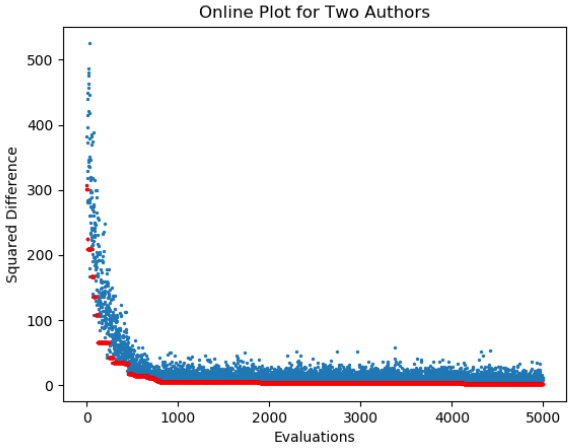

Figure 26: Online Performance of EEDA prober for 2 Authors over CASIS-25 Dataset

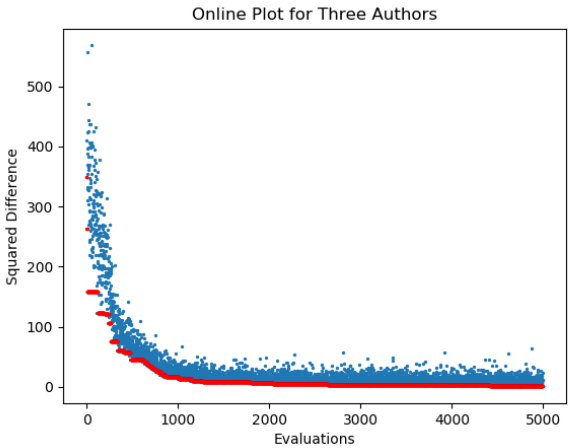

Figure 27: Online Performance of EEDA prober for 3 Authors over CASIS-25 Dataset

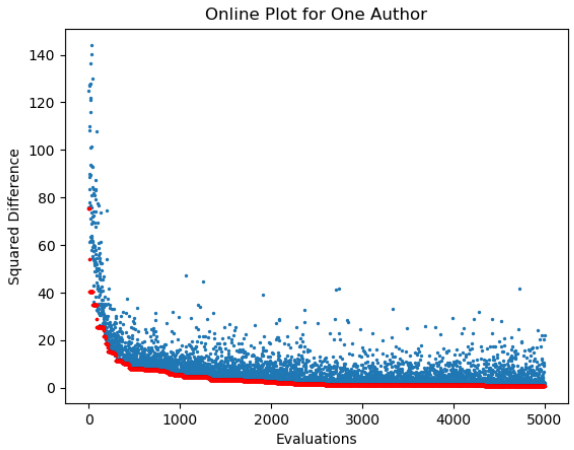

Figure 28: Online Performance of SSGA prober for 1 Author over SEC Sports Writers Dataset

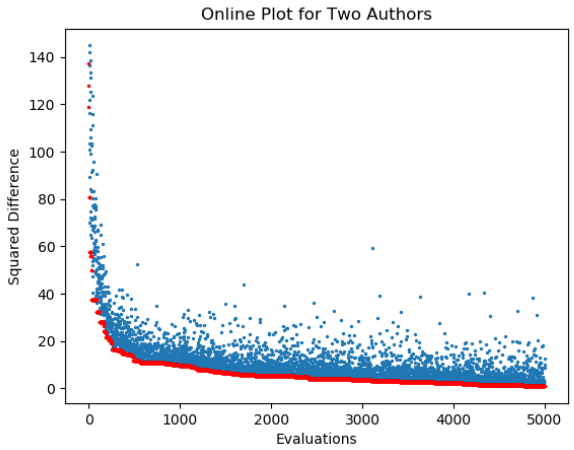

Figure 29: Online Performance of SSGA prober for 2 Authors over SEC Sports Writers Dataset

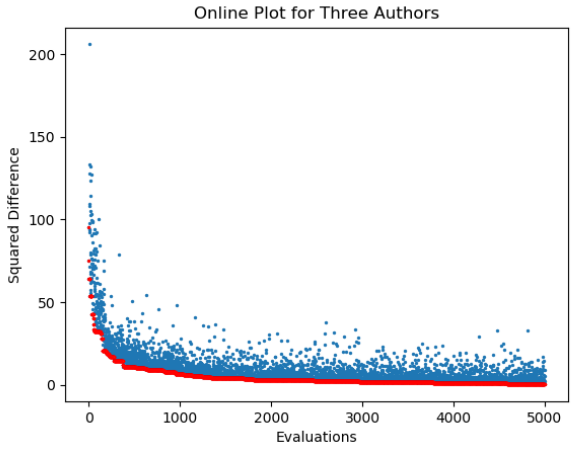

Figure 30: Online Performance of SSGA prober for 3 Authors over SEC Sports Writers Dataset

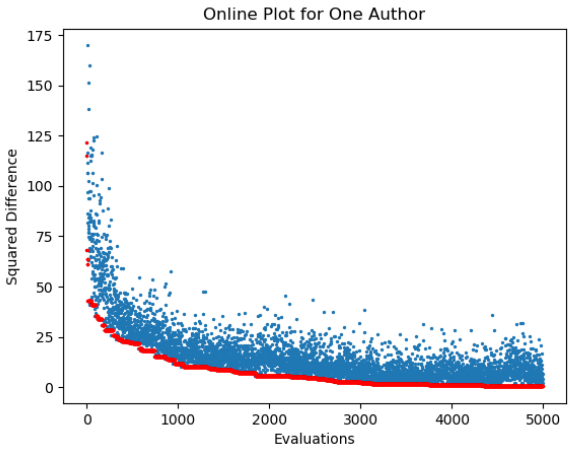

Figure 31: Online Performance of EGA prober for 1 Author over SEC Sports Writers Dataset

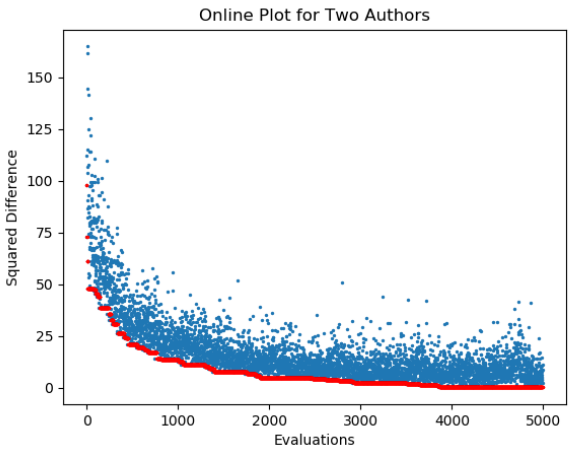

Figure 32: Online Performance of EGA prober for 2 Authors over SEC Sports Writers Dataset

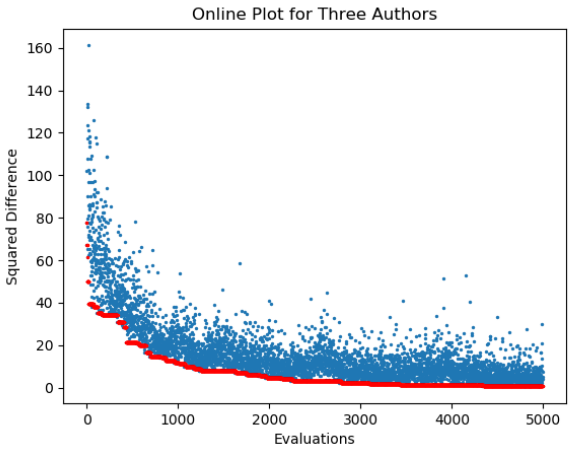

Figure 33: Online Performance of EGA prober for 3 Authors over SEC Sports Writers Dataset

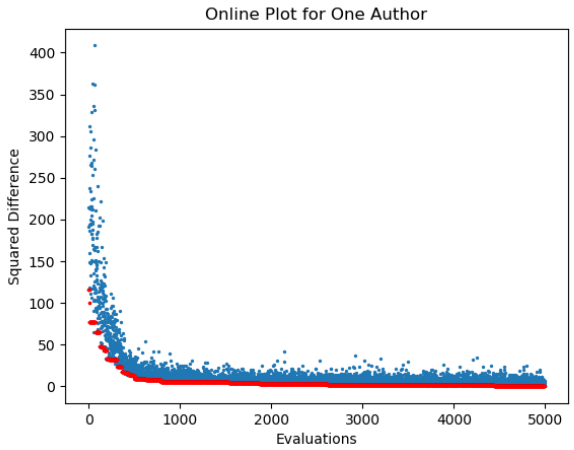

Figure 34: Online Performance of EEDA prober for 1 Author over SEC Sports Writers Dataset

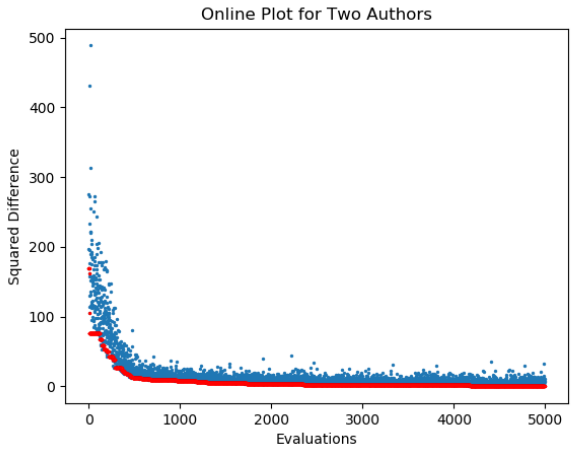

Figure 35: Online Performance of EEDA prober for 2 Authors over SEC Sports Writers Dataset

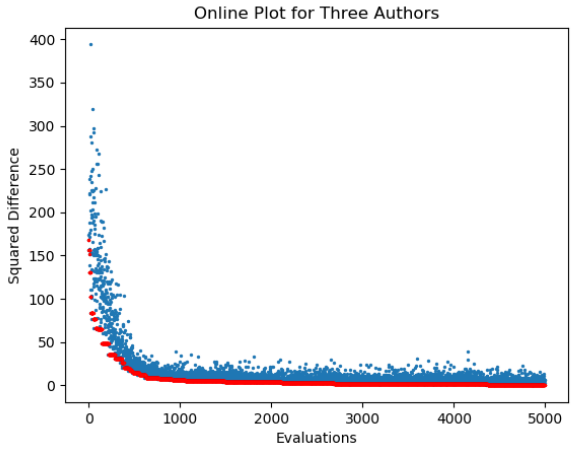

Figure 36: Online Performance of EEDA prober for 3 Authors over SEC Sports Writers Dataset

With the use of a larger range for initializing the population, we obtain a better probing performance only in the case of Steady State GA for 1, 2 and 3 authors and Elitist EDA for 2 authors.

IV. Breakdown of Work

Rahul was also responsible for implementing Elitist GA and Elitist EDA for probing. He also experimented with Steady State GA, Elitist GA and Elitist EDA by changing the range of the initial population.

Sutanu was responsible for implementing the Steady State GA for probing.

References

[1] Brennan, Michael Robert, and Rachel Greenstadt. "Practical attacks against authorship recognition techniques." In IAAI. 2009.

[2] Simko, Lucy, Luke Zettlemoyer, and Tadayoshi Kohno. "Recognizing and Imitating Programmer Style: Adversaries in Program Authorship Attribution." Proceedings on Privacy Enhancing Technologies 2018, no. 1 (2018): 127-144.

[3] R Alapati, S Bhattacharya, S Dsouza. “Authorship Attribution with GatorCAAT: Genetic and Evolutionary Feature Selection (GEFeS)”.

[4] http://www.staugustine.com/.

[5] http://www.ocala.com/.

[6] https://www.alligator.org/.

[7] https://www.orlandosentinel.com/.

[8] http://www.tampabay.com/.

[9] https://www.palmbeachpost.com/.

[10] https://www.local10.com/.

[11] https://www.cbssports.com/.

[12] https://www.tennessean.com/.

[13] https://www.alligatorarmy.com/.

[14] https://docs.python.org/3/library/pickle.html.

[15] http://scikitlearn.org/.