University of Southampton

Faculty of Engineering and Physical Sciences

Electronics and Computer Science

Filter Based Genetic Algorithm for ML-based Intrusion Detection Systems

By

Orkun Ratip

08.09.2022

Supervisor: Dr. BooJoong Kang

Second examiner: Dr. Markus Brede

A dissertation submitted in partial fulfilment of the degree

of MSc Cyber Security

Abstract

IoT devices are increasingly being used in various industries. The number of IoT devices skyrocketed over the past years. From medicine to security, IoT devices are used to help with automating and monitoring processes. The network these devices communicate and interconnect is called the IoT network. IoT devices use lots of sensitive information such as location, age, height, video footage etc. This raises a security concern because information gathering methods such as network intrusions can be used to access sensitive data if necessary security precautions are not taken. Moreover, denial of service attacks can cause large economic damage to many industries. An effective defence mechanism is intrusion detection systems (IDS). Machine learning (ML) based intrusion detection systems have the ability to learn from network patterns to recognize anomalous patterns. However, ML-based IDS is computationally costly. The amount of data is too large to effectively process with low cost hardware. A solution to this is using feature selection algorithms to reduce the dimensionality of data. Genetic algorithm is a widely used and effective algorithm for optimization. This project aims to improve the genetic algorithm and implement a lightweight IDS to enhance IoT network security. Instead of using a ML algorithm, a filter based method that explores data attributes will be used to improve the fitness function of the genetic algorithm. The results show that the improved genetic algorithm has noticeably lower computational cost as well as accuracy scores around 95%-98% depending on the sample size.

Statement of Originality

Table of Contents

Chapter 1: Introduction

1.1 Background

The Internet of things (IoT) is increasingly used in countless industries. From medicine to security, IoT devices play a big role in the functioning of critical systems [1]. These devices are interconnected by a network named IoT network. IoT devices communicate through the IoT network by collecting and exchanging various data from users. Although these devices are very practical and help with everyday chores, the amount of sensitive data they collect and transfer pose a large security threat [5]. The IoT network is extremely large because of the vast number of IoT devices [4]. The amount of data generated by this network can be classified as big data [6]. Therefore, manually monitoring the IoT network is infeasible.

Cyber criminals commonly use network intrusion attacks to gather information (Reconnaissance) or render systems useless (Denial of Service) [7]. IoT network is a high reward target for attackers and it will progressively be a victim of intrusion attacks.

An intrusion detection system (IDS) is a practical and effective way of preventing network intrusions. Latest technology ML-based IDS provides automated learning and classification processes based on network data. However, data pre-processing and model training time are still impediments when working with big data [8]. Therefore, a lightweight and effective IDS can be a preferable option for this task, especially if computational resources are limited.

1.2 Aims and Objectives

The aim of this project is to improve Genetic Algorithm to implement a lightweight IDS. A ML-based IDS will be implemented by combining SVM algorithm with genetic algorithm. Moreover, the genetic algorithm's slow convergence time will be improved by changing its fitness function (Filter based GA). Additionally, necessary pre-processing techniques to boost the performance measures such as accuracy and precision will be done. Lastly, the computational cost and performance measures of GA and filter based GA (FGA) will be compared to see the performance improvements.

1.3 Outline

There are 7 chapters in this report including introduction, project plan, literature review, methodology, evaluation and conclusion.

Chapter 1: Introduction

In this section, some background information on the internet of things, intrusion detection systems and IoT network will be given. Moreover, the aims and objectives of this project will be proposed.

Chapter 2: Project Plan

The project schedule will be shown as a Gantt chart. Also, a personal reflection on the project will be given.

Chapter 3: Background

In this chapter, detailed background information on intrusion detection systems, feature selection algorithms and genetic algorithms will be provided.

Chapter 4: Literature Review

In this chapter, previous research on intrusion detection systems and genetic algorithms will be explored.

Chapter 5: Methodology

A detailed explanation of the genetic algorithm's design and implementation will be given. The changes that were made to the fitness function as well as hyper-parameters of the algorithm will be discussed.

Chapter 6: Evaluation

In this part, the experimental setup including dataset and pre-processing is shown. Moreover, the performance of GA and FGA will be discussed by using quantitative results.

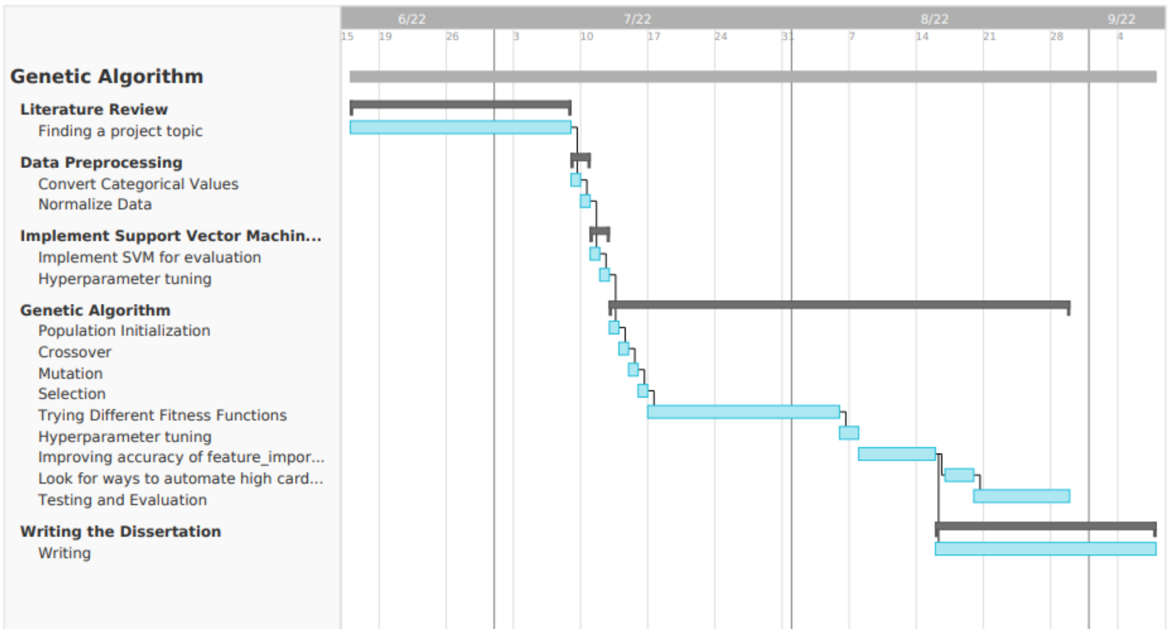
Chapter 7: Conclusion

A brief summary of the project will be given. Also, the limitations of the project and future works will be discussed.

Chapter 2: Project Plan

2.1 Gantt Chart

The Gantt chart was made at the end of the project. At the beginning of the project, literature review took a lot more time than I anticipated. I had to change the project topic many times to find an innovative and challenging one. I used OneNote to communicate with my supervisor for weekly action items. Moreover, we did weekly meetings to discuss the milestones and more action items. This Gantt chart outlines the order and the trivial procedure of the whole project period.



2.2 Personal Reflection

The purpose of this project is to improve security of the IoT environment. I believe this topic is and will be very important in the upcoming future as IoT devices are continuously being used in many industries. It has been very exciting to work with my supervisor who has sophisticated expertise in this field. Initially, finding a project topic was very challenging since the topic was researched by many experts in recent years. However, with good guidance I was able to find an interesting topic. The hardest part of the project was to achieve a high accuracy with filter based feature evaluation. I did extensive research and made several attempts to figure out proper data processing procedures and hyper parameters.

This is my very first try to deeply research feature selection algorithms as well as improve one that is being widely used by many experts. I am thankful to my supervisor that always challenged me to work harder, research more and think widely leading to improving my knowledge and perspective. This project has contributed to not only my technical knowledge and practical skills, but my planning, management skills as well as viewpoint.

Chapter 3: Background

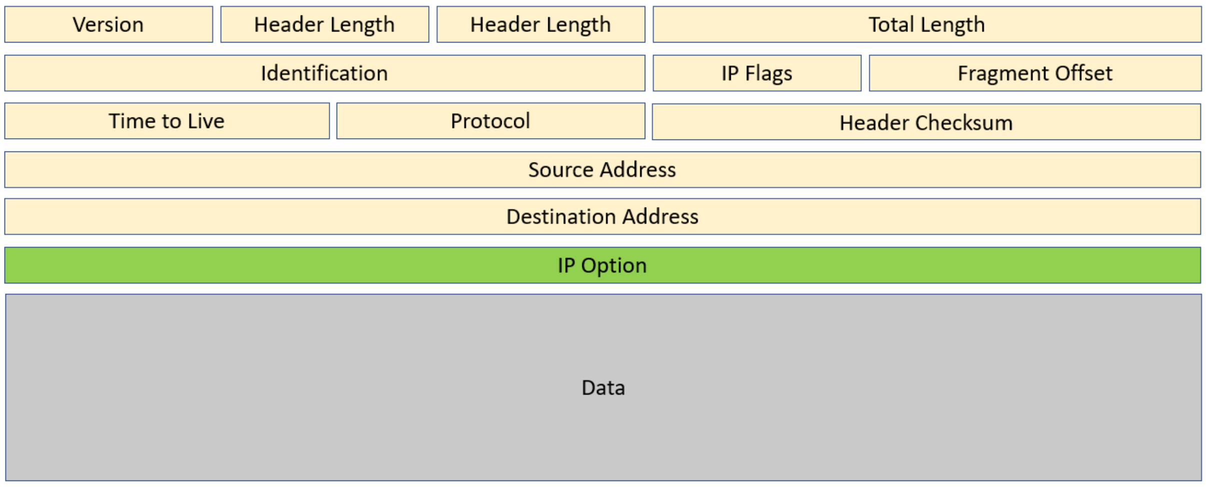
In this section, I will explain intrusion detection systems and feature selection algorithms. Additionally, I will address the motivation behind this project by discussing the limitations in an IoT environment.

3.1 IoT Network

IoT applications are getting more and more popular with time [2]. The number of IoT devices skyrocketed in the past couple of years [4]. These devices are continuously being used in various industries including medicine, smart homes and security [3]. IoT helps people optimise tasks in daily life by analysing, summarising and monitoring data [9]. Some examples of IoT devices popularly used are door locks, video doorbells, fridges as well as heating and cooling systems. IoT devices use lots of sensitive data to display optimal performance [12]. Moreover, most IoT devices are wirelessly connected to the IoT network [11]. These properties bring out security threats for the IoT environment such as denial of service attacks and information theft [10]. Most IoT devices are manufactured without security of concern. Moreover, they often do not use up to date software which increases the chance of exploitation [13]. An IoT environment is a high priority target for attackers, because they can make a significant negative impact on financial aspects of industries [14]. Therefore, securing the IoT network from intrusions is top priority [15].

3.2 Network Data

A network data can be formed by capturing network packets and extracting packet header information. A network packet consists of various headers and each header has their own unique values. Some examples of headers are duration, time to kill, protocol type etc. These network packets must be captured in a realistic environment, whether it is a simulated network or a facility. This will ensure that the intrusion detection system performs optimally in a real life scenario.



(Figure 1: An example of a network packet)

3.2 Intrusion Detection Systems

Intrusion detection systems are a popular way of preventing network intrusions. There are many algorithms that can be used to build an intrusion detection system. ML, deep learning (DL), swarm and evolutionary based algorithms are used in IDSs [16]. IDS can be classified by their analysis strategy. There are two types of analysis strategy: misuse (signature) based and anomaly based [16].

Misuse (signature) Based IDS: In misuse based IDS, intrusion detection is done by analysing a database that contains previous system vulnerabilities and attack patterns. This is achieved by storing logs of all exploited system vulnerabilities and successful attack patterns. A drawback of this method is keeping the database up to date [16]. Additionally, misuse based IDS can not detect zero day attacks since it is limited to known attack and vulnerability specific data.

Anomaly Based IDS: The aim of this approach is to analyse normal and malicious network data patterns. By doing so, a network anomaly that deviates from normal network patterns can be detected successfully. This can be done by building a ML model and training it with normal and malicious network data. Anomaly based IDSs can detect zero day attacks [16]. This is an important advantage because attackers are constantly changing strategy and finding innovative attack methods.

3.3 Feature Selection

Feature selection algorithms can be used as a data pre-processing tool. They can filter redundant features, as well as features with low impact on class prediction. This helps with reducing the data size and computational cost, as well as improving accuracy [17]. Feature selection framework can be based on selection criteria, evaluation criteria and learning method [18]. Selection criteria implies the feature subset to be selected according to certain rules. Evaluation criteria measures the validity of a feature subset according to the performance measures of the developers choice. Finally, the learning algorithm is the method that determines the performance of a given feature subset. There are two main types of feature selection algorithms: wrapper based and filter based.

Wrapper based: Wrapper based feature selection uses ML or DL as a learning algorithm. With this approach a feature subset is given as input to the ML algorithm. The feature subset that gives the highest learning performance (accuracy, precision, recall etc.) is chosen as the optimal solution. One setback for wrapper based feature selection is the search space. Considering a high dimensional data, using ML algorithms will result in a very high computational cost since the performance measurement will repeat for each feature subset [17]. Therefore, in a real life scenario that includes working with big data, this approach may be impractical.

Filter based: Filter based methods do not use ML as a learning algorithm. They consider attributes of the data to determine the importance of features [18]. There are two types of filter based methods. Univariate selection measures importance of features individually whereas multivariate selection measures importance of a batch of features. Filter selection considers traits such as feature correlation, mutual information gain, entropy, impurity etc. It is a faster method compared to wrapper based selection since instead of learning algorithms, it uses statistical measures. Therefore, filter based selection is more suitable for problems involving high dimensional data.

3.4 Genetic Algorithm

Genetic algorithms (GA) are widely used for optimisation [19]. The algorithm is inspired by Darwin's theory of natural selection [20] which emphasises the phrase "survival of the fittest". Genetic algorithm is initialised by a population that contains various feature subsets. These feature subset candidates are called chromosomes. They are coded in the form of a binary sequence where each bit represents if a feature is included or not: 0 for features that are not included, 1 for features that are included. For each iteration of the algorithm a series of operations are made to select the fittest chromosomes. There are four operations in GA: selection, crossover, mutation and fitness evaluation. Figure 2 shows the lifecycle of the genetic algorithm.

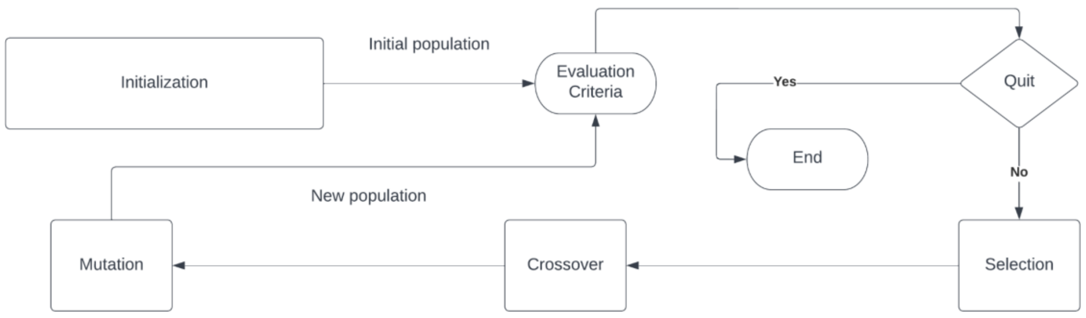


Figure 2: Life cycle of a genetic algorithm

Selection: In this operation, a number of fittest chromosomes are selected according to their performance scores. A ML algorithm is used to measure the performance of each chromosome and the highest scoring chromosomes are selected to form the new generation.

Crossover: This operation imitates reproduction [19]. Every two chromosomes from the fittest chromosomes will be paired to represent parents. Each of them will produce two children that will be added to the new generation. A child will inherit a portion of the sequence of its parents. This will be achieved by assigning a crossover point and dividing the binary sequence of each parent from that point.

Chromosome 1	**1001 1001**
Chromosome 2	**1111 0000**
Offspring 1	**1001 0000**
Offspring 2	**1111 1001**

Figure 3: Single point crossover (Crossover point = 4)

Mutation: The new generation will undergo another transformation that involves flipping a random bit in their binary sequence. This will improve variation in the population.

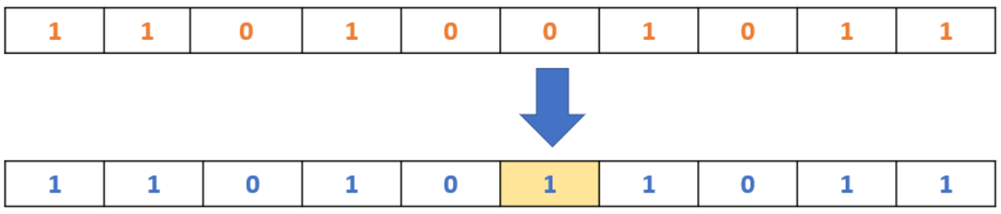


Figure 4: Single point mutation (Mutation point = 6)

As the algorithm proceeds the chromosomes get fitter over generations since the algorithm's selection operation chooses the highest scoring individuals.

Fitness function: Genetic algorithm is considered as a wrapper based feature selection method. This is due to the fitness function being a ML algorithm. The fitness of each chromosome is determined by training a ML algorithm and measuring the classification performance.

Due to the fitness function, GA has a slow convergence time [19]. This time complexity can increase as the sample size and/or the data dimensions get larger. Fitness function takes most of GA's time. Figure 5 shows clearly how much more time fitness function consumes relative to the whole algorithm.

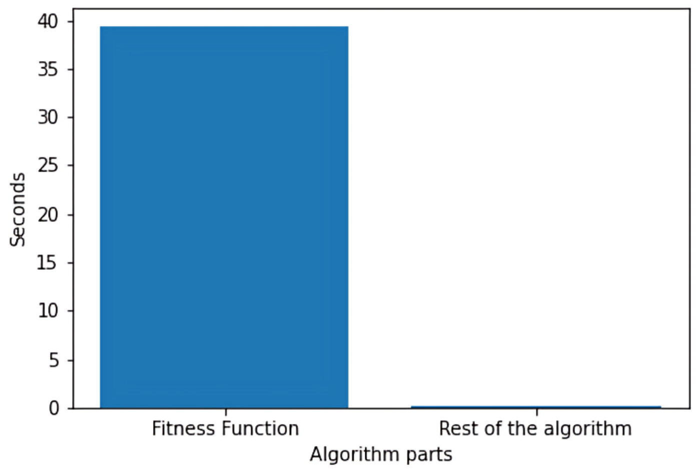


Figure 5: Computational cost of the fitness function with respect to the rest of the algorithm (Sample size = 2000), X-axis : parts of the algorithm, Y-axis: seconds

Chapter 4: Literature Review

Genetic algorithms have been used by many researchers for intrusion detection systems. Hoque proposed an IDS using GA. The algorithm was implemented using KDD’ 99 dataset with 4 distinct attack types and normal network activities. The algorithm achieved a 0.95 detection rate score. Detection rate was calculated as the number of true positives divided by the sum of number of false negatives and true positives.

Gharaee proposed an IDS using GA and SVM. In this approach, a fitness function that consists of true positive rate (TPR), false positive rate (FPR) and the number of features (NumF(S)) selected was used. The fitness of a feature was computed as Wa * TPR + Wb*FPR + Wc*NumF(S), where Wa is the weight value for TPR, Wb is the weight value of FPR and Wc is the weight value of the number of samples. SVM was used to measure FPR and TPR. The algorithm achieved very impressive multi-class classification performance results with accuracy being generally around 97%-99% over multiple datasets.

Moreover, Tapaswi proposed a lightweight genetic algorithm. This version of the algorithm applies an ageing based selection. In ageing based selection, the new population is formed by only the latest generation and the older generation (parents) are discarded. This has proved to achieve a high convergence time. However, the algorithm was not tested in the context of intrusion detection. Rather, it was used for mobility detection.

AlSukk proposed Diverse Genetic Algorithm (DGA) to address the elitism issue of genetic algorithm. Elitism is the guaranteed selection of the fittest members which results in premature convergence. In DGA this issue was solved by setting a threshold as a percentile of the number of iterations. This threshold is set to 10% of the maximum number of iterations. When this threshold is reached, modified selection will start. After the first parent is selected, the second parent is selected from the section that does not represent parent 1. This increases variety in the population. Moreover, DGA introduces another threshold as 30% of the maximum number of iterations. When this threshold is reached, duplicate members from the population are replaced with random members that are formed from the unused features. This method proved to be successful with very high accuracy results. However, experimental results show that DGA is impractical when used with high dimensional and highly redundant data.

Many researchers have worked on GA to solve both the diversity problem and the slow convergence time. However, a research gap still exists when it comes to improving the computational cost in the context of IDS. This project aims to fill the research gap by changing the design of GA in an innovative way. Replacing the fitness function from a ML algorithm to an impurity based method will improve the speed of the algorithm immensely.

Chapter 5: Filter Based Genetic Algorithm

One of the drawbacks of GA is the slow convergence time because GA uses a ML algorithm as a fitness evaluation operation. In this case the ML algorithm executes for each feature subset. The ML algorithm is costly as it is but with the repeated operation the computational cost increases a lot. Especially with higher samples and dimensions. This project aims to improve the computational cost by using a feature importance function instead of a ML algorithm. The feature importance function will execute only one time, further improving the speed of the algorithm.

5.1 Algorithm

Population initialization: The initial population includes 100 chromosomes. Each chromosome is a binary array that consists of randomly generated 0's and 1's. 0's represent features that are not included in the feature subset. Similarly, 1's represent features that are included in the feature subset.

Single Point Crossover: A random crossover point is selected between 1 and len(array) -1. From then onwards, the two parents exchange the binary sequences located at array[crossover_point:].

Mutation: A random mutation point is selected between 0 and len(array). Following that, a bit flip operation is done at array[mutation_point].

Parent Selection: At every iteration the ¼ of the highest scoring chromosomes are chosen as parents. This way at every iteration the population is halved.

Evaluation Criteria: The evaluation criteria can include an accuracy goal. However, in this version of GA, population continuously diminishes and the algorithm stops when the number of chromosomes reaches 1. At that point, the algorithm already achieves very good accuracy results. Therefore, a set accuracy goal is not needed.

5.2 Fitness Function

GA has 5 main operations: population initialization, crossover, mutation, parent selection and fitness function. Population initialization is generating some number of binary arrays with random 0's and 1's. Crossover is the sequence exchange of two chromosomes from a given point. Similarly, a bit flip operation is done on a chromosome as a form of mutation. In parent selection the fittest chromosomes of the population are selected as parents. All of these operations take negligible amount of time. Even with very low computational resources executing these operations may not be a problem. However, fitness functions take an immense amount of time due to the repeated execution of a ML algorithm. Therefore, improvements can be made on the fitness function to reduce time complexity. In this version of GA, Scikit Learn feature importance function is used to assess the mean decrease in impurity for each feature. If the decrease in impurity is high, it means the feature is important. The function is used only one time to determine each feature's importance. Feature importance function outputs an array of scores. When calculating the fitness of a feature subset, the impurity scores of individual features are added to generate a fitness score.

Feature importance function: Feature importance is closely related to decision trees. It is calculated as the decrease in node impurity weighted by the probability of reaching that node. The node probability can be calculated by the number of samples that reach the node, divided by the total number of samples [23].

Scikit learn calculates feature importance as:

$$ni_j = w_j + c_j - w_{left(j)}c_{left(j)} - w_{right(j)}c_{right(j)}$$

where, ni_j is the importance of node j, w_j is the weighted number of samples reaching node j, cj is the impurity value of node j , left(j) is the child node from left split on node j, right(j) is child node from right split on node j.

From then onwards, the importance of each feature can be calculated as:

$$fi_j = \sum_{j = \text{node j splits on feature}} i^{ni}_j \Big/ \sum_{k \in \text{all nodes}} ni_k$$

where, fi_j is the importance of feature i, ni_j is the importance of node j.

Chapter 6: Evaluation

6.1 Experimental Setup

In this section the experimental setup will be discussed by mentioning the dataset and the classification algorithm used as well as data pre-processing steps.

6.1.1 Dataset

For the IDS, I used the NSL-KDD dataset. NSL-KDD dataset was easy to access, up to date and has been used by many researchers in recent years. The dataset contains 41 features and 1 class label [21]. It includes 4 types of attacks. However, in this project I will implement binary classification. Moreover, the dataset contains a mix of binary, continuous and categorical features. Therefore, data pre-processing is needed.

6.1.2 Data pre-processing

Firstly, the categorical values such as service and protocol_type must be converted to numerical values. This can be done by using Pandas' get_dummies() function. This function converts categorical variables into indicator variables. After this operation, the categorical features will be dropped and be replaced by the indicator variables. Additionally, class labels are categorical and represent attack types or normal network activity. In order to achieve binary classification every attack type was encoded as 0 and normal activity as 1. After data pre-processing, the number of features increased. This increase depends on the sample size, as more samples introduce more categories. Moreover, the data has been normalised using Scikit Learn StandardScaler() function. Impurity based feature importance methods suffer from extreme bias towards high cardinality features [22]. The dataset used in this project contains lots of binary features and a low amount of high cardinality features. Therefore, high cardinality features were dropped for a better performing feature importance function. This process increased the accuracy by 25-28%. Moreover, in the dataset some features only had one value. These features have no effect on classification. Hence they were dropped from the dataset for faster process speed. The effect on accuracy increase was negligible.

6.1.3 Classification Algorithm

I used support vector machines (SVM) to implement the IDS. Because SVM works very well with both filter feature selection methods and wrapper feature selection methods [17]. Additionally, SVM is very capable of solving nonlinear and high dimensional problems compared to other classification algorithms [24,25]. SVM is also very good at generalisation [27,28], which is important for practices that go beyond the training dataset.

6.2 Performance Measures

In this section the performance of FGA will be analysed. Comparisons will be made on performance metrics such as computational cost and accuracy. All of the measures are taken as an average of 100 iterations. For performance measures the Scikit Learn metrics library is used.

6.2.1 Computational Cost

The main purpose of this project was to improve GA to achieve faster convergence time.

Table 1 and figure 6 shows that computational cost is significantly lower for FGA, especially as the sample size increases.

Table 1: Computational cost of GA and FGA in seconds (N: sample size)

	GA	**FGA**
N = 500	3.217588197860002	0.004210971709999995
N = 1000	11.237424663360008	0.004161524989999776
N = 2000	39.49993457397001	0.004260668489999802
N = 4000	145.1944092250899	0.004281935710000937

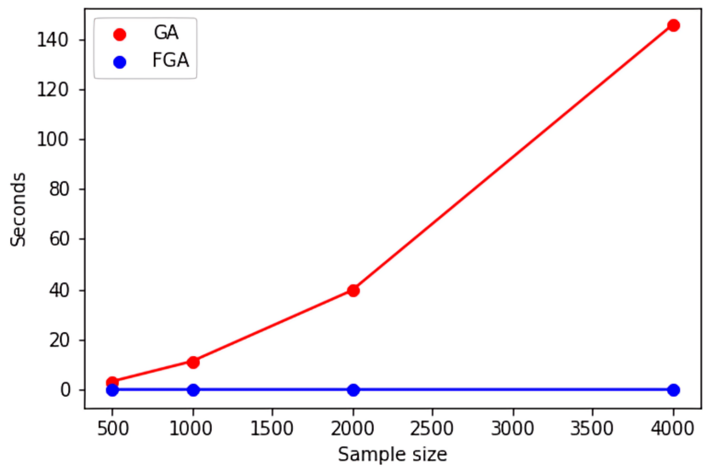


Figure 6: Comparison of computational cost of GA and FGA in seconds (X-axis = sample size, Y-axis = seconds)

Table 2: Computational cost of fitness function for both algorithms in seconds (N = sample size)

	GA	**FGA**
N = 500	3.097476460650009	0.0038459507400008166
N = 1000	11.113220329210003	0.0038041779499986463
N = 2000	39.359323855440095	0.0038969388099997105
N = 4000	145.0220745035302	0.003910511630001139

Table 2 shows that the fitness function has improved greatly. GA uses a ML algorithm to assess fitness of each chromosome. The ML algorithm is time consuming as it is, but it also executes for each chromosome. In FGA, fitness evaluation is done by comparison and the importance of features are only calculated once. Therefore, it drastically reduces the computational cost.

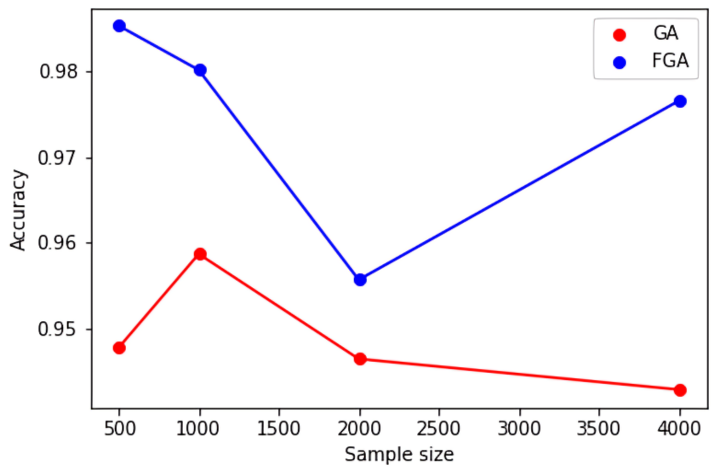
6.2.2 Accuracy

Accuracy is the measure of all the correctly identified cases. Accuracy comparison for GA and FGA can be seen at table 3.

Table 3: Accuracy comparison for GA and FGA (N = sample size)

	GA	**FGA**
N = 500	0.9478	0.9853
N = 1000	0.9587	0.98015
N = 2000	0.946475	0.9557
N = 4000	0.9429000000000001	0.9765375

Moreover, the accuracy plot for each algorithm can be seen at figure 7.



(Figure 7: Accuracy comparison for FGA and GA (X-axis = sample size, Y-axis = accuracy)

The accuracy results from FGA is very remarkable. These high accuracy results are due to good data pre-processing practices according to Scikit Learn feature importance function.

More performance measures are also explored for FGA.

6.2.3 Standard Deviation of Accuracy

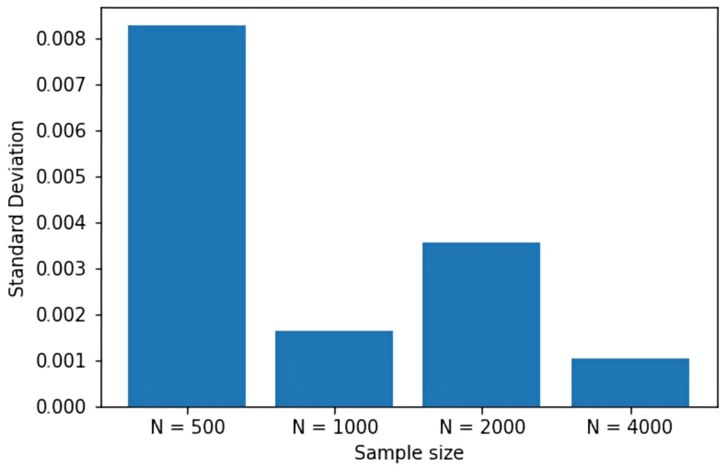


Figure 8: Standard deviation of accuracy for FGA (X-axis = sample size, Y-axis = standard deviation

The standard deviations are very low, meaning that for each iteration the accuracy values are clustered around the mean. The data is less spread out. This in turn proves that the algorithm is more consistent in its accuracy performance.

6.2.4 Precision

Precision is the ratio of Fp / (Fp + Tp), where Fp is false positive and Tp is true positive. Essentially, precision measures the capability of the classifier not to label a positive sample as negative. Precision can take values between 0 and 1: 0 being the worst and 1 being the best.

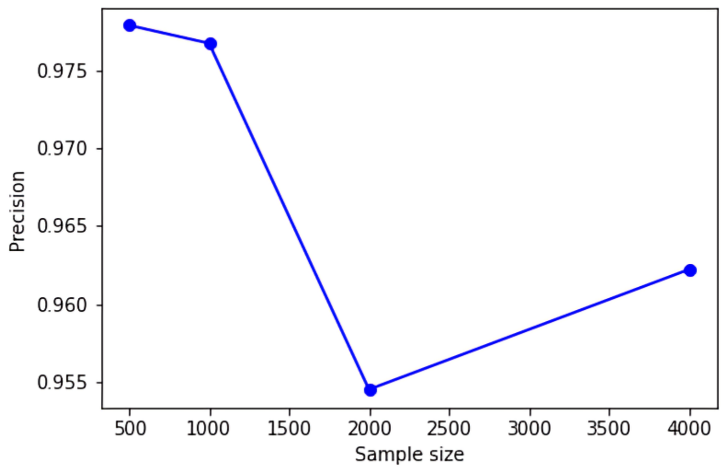


Figure 9: Precision metrics of FGA (X-axis = sample size, Y-axis = precision

Figure 9 shows that FGA is highly capable of correctly classifying a positive sample.

6.2.5 Recall

Recall is the ratio of Tp / (Tp + Fn), where Tp is true positive and Fn is false negative. Recall measures the classifier's ability to identify actual positive samples. Similar to precision, recall can take values between 0 and 1: 0 being the worst and 1 being the best.

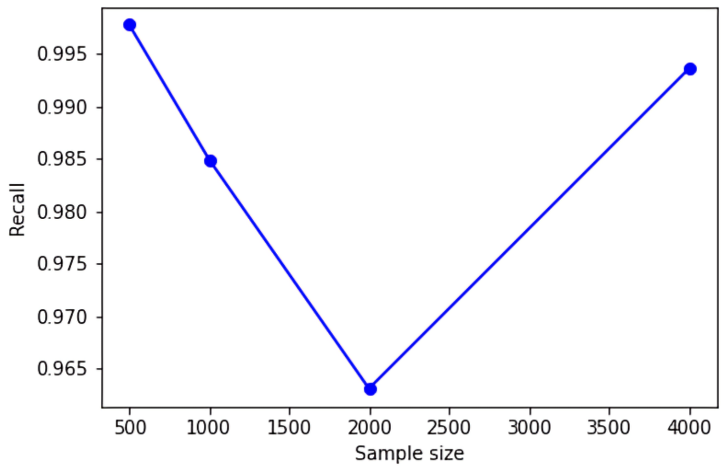


Figure 10: Recall metrics of FGA (X-axis = sample size, Y-axis = recall)

Figure 10 shows that FGA is highly capable of identifying actual positive samples.

6.2.6 F1 Score

F1 score can be interpreted as the harmonic mean of precision and recall. F1 score is very important because it measures performance by combining two otherwise metrics. In most real life classification problems, imbalanced class distributions exist. Accuracy, only measures the correctly identified cases. Whereas F1 score gives a better measure of the incorrectly classified cases than accuracy. The mathematical formula for F1 score is as follows: F1 = 2 * (precision * recall) / (precision + recall). F1 score can take values between 0 and 1: 0 being the worst and 1 being the best.

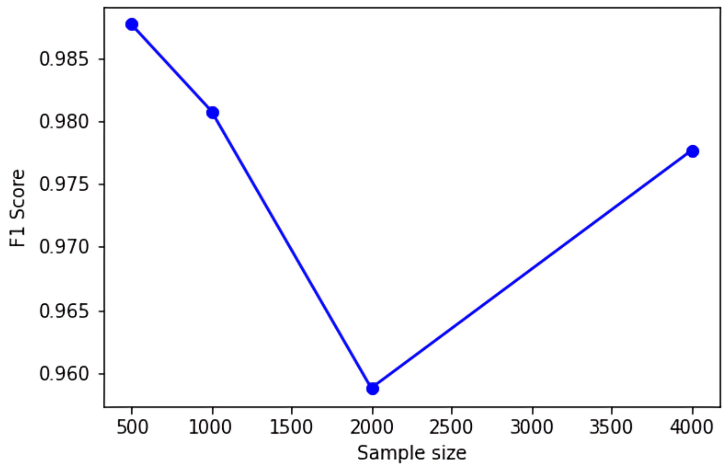


Figure 11: F1-score of FGA (X-axis = sample size, Y-axis = F1-score)

According to figure 11, FGA shows impressive F1 scores for all of the sample sizes. This shows FGA's balanced ability to both capture positive cases and be accurate with the cases it does capture.

FGA has surpassed GA significantly on speed. Moreover, it achieved very high performance results with accuracy over 95% in multiple sample sizes. The accuracy scores had very low standard deviation which emphasises that the algorithm is consistent. With F1 scores greater than 96% for every sample size the algorithm is very balanced as its ability of classifying both incorrectly and correctly identified cases is outstanding. Additionally, the algorithm has proved to be highly capable of identifying actual positive samples and not classifying them as negative. This can clearly be seen by looking at the precision and recall scores that stand around 95%-99% over multiple sample sizes.

Chapter 7: Conclusion

There is a pressing need to improve security for the IoT network. This project aims to implement an intrusion detection system to increase the security for the IoT environment by improving genetic algorithm for a lightweight IDS. In this paper, a brief introduction of feature selection algorithms, genetic algorithms and intrusion detection systems as well as previous research on the topic have been given. Then, a detailed explanation of the filter based genetic algorithm's implementation is given. In this version of the algorithm, the fitness function was changed from a machine learning algorithm to an impurity based feature importance algorithm. Later, the experimental setup is explained with dataset and pre-processing as the main focus. Lastly, the performance of FGA is determined with quantitative measures and plots. With F1 and accuracy scores over 95% for multiple sample sizes, FGA proved to be very effective at classification. Moreover, FGA highly surpasses GA in terms of speed.

7.2 Limitations & Future Work

The biggest limitation of this project was the amount of data that could be used. The NSL-KDD dataset contains approximately 1 million samples. However, in this project the maximum number of samples used were 4000 because of limited computational resources. Therefore, testing with more samples can be done for the algorithm. Moreover, data pre-processing for Scikit Learn feature importance function was done manually. An algorithm can be implemented to shape the dataset into a form which the feature importance function that can perform optimally.

References

[1] S. V. Zanjal and G. R. Talmale, “Medicine reminder and monitoring system for secure health using IOT,” Procedia Computer Science, vol. 78, pp. 471–476, 2016.

[2] B. Dorsemaine, J.-P. Gaulier, J.-P. Wary, N. Kheir, and P. Urien, “Internet of things: A definition &amp; taxonomy,” 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies, 2015.

[3] A. Witkovski, A. Santin, V. Abreu, and J. Marynowski, “An IDM and key-based authentication method for providing single sign-on in IOT,” 2015 IEEE Global Communications Conference (GLOBECOM), 2015.

[4] P.Brody and V.Pureswaran, “Devicedemocracy–saving the future of the internet of things. ,” IBM Inst. Bus. Value, 2014.

[5] A. Ukil, S. Bandyopadhyay and A. Pal,” IoT-Privacy: To be private or not to be private,” 2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 123-124, 2014.

[6] A. Dehghantanha and K.-K. R. Choo, “Big Data and Internet of Things Security and Forensics: Challenges and Opportunities,” Handbook of Big Data and IOT Security, Cham: Springer International Publishing, 2019.

[7] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, “CNN-based network intrusion detection against denial-of-service attacks,” Electronics, vol. 9, no. 6, p. 916, 2020.

[8] A. Adnan, A. Muhammed, A. A. Abd Ghani, A. Abdullah, and F. Hakim, “An intrusion detection system for the internet of things based on Machine Learning: Review and Challenges,” Symmetry, vol. 13, no. 6, p. 1011, 2021.

[9] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, “Proposed security model and threat taxonomy for the internet of things (IOT),” Recent Trends in Network Security and Applications, pp. 420–429, 2010.

[10] P. Varga, S. Plosz, G. Soos, and C. Hegedus, “Security threats and issues in automation IOT,” 2017 IEEE 13th International Workshop on Factory Communication Systems (WFCS), 2017.

[11] S. A. Al-Qaseemi, H. A. Almulhim, M. F. Almulhim, and S. R. Chaudhry, “IOT architecture challenges and issues: Lack of standardization,” 2016 Future Technologies Conference (FTC), 2016.

[12] C. Thirumalai and H. Kar, “Memory efficient multi key (MEMK) generation scheme for secure transportation of sensitive data over cloud and IOT

devices," 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), 2017.

[13] S. Pallavi and V. A. Narayanan, "An overview of practical attacks on BLE based IOT devices and their security," 2019 5th International Conference on Advanced Computing &amp; Communication Systems (ICACCS), 2019.

[14] M. M. Salim, S. Rathore, and J. H. Park, "Distributed denial of service attacks and its defenses in IOT: A survey," The Journal of Supercomputing, vol. 76, no. 7, pp. 5320–5363, 2019.

[15] M. Conti, A. Dehghantanha, K. Franke, and S. Watson, "Internet of things security and forensics: Challenges and opportunities," Future Generation Computer Systems, vol. 78, pp. 544–546, 2018.

[16] A. Thakkar and R. Lohiya, "A survey on Intrusion Detection System: Feature selection, model, performance measures, application perspective, challenges, and future research directions," Artificial Intelligence Review, vol. 55, no. 1, pp. 453–563, 2021.

[17] A. Jovic, K. Brkic, and N. Bogunovic, "A review of feature selection methods with applications," 2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2015.

[18] J. Novakovic, P. Strbac, and D. Bulatovic, "Toward optimal feature selection using ranking methods and classification algorithms," Yugoslav Journal of Operations Research, vol. 21, no. 1, pp. 119–135, 2011.

[19] N. El Aboudi and L. Benhlima, "Review on wrapper feature selection approaches," 2016 International Conference on Engineering &amp; MIS (ICEMIS), 2016.

[20] C. T. Walbridge, "Genetic algorithms: What computers can learn from Darwin," Technology Review; (USA), vol. 92, 1989.

[21] B. Ingre and A. Yadav, "Performance Analysis of NSL-KDD dataset using ann," 2015 International Conference on Signal Processing and Communication Engineering Systems, 2015.

[22] D. Bethge, L. Chuang, and T. Grosse-Puppendahl, "Analyzing transferability of happiness detection via gaze tracking in multimedia applications," Symposium on Eye Tracking Research and Applications, 2020.

[23] S. Ronaghan, The Mathematics of Decision Trees, Random Forest and Feature Importance in Scikit-learn and Spark, Medium, May 11, 2018. https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest-and-feature-importance-in-scikit-learn-and-spark-f2861df67e3

[24] P. Tao, Z. Sun, and Z. Sun, “An improved intrusion detection algorithm based on Ga and SVM,” IEEE Access, vol. 6, pp. 13624–13631, 2018.

[25] V. K. Chauhan, K. Dahiya, and A. Sharma, “Problem formulations and solvers in Linear SVM: A Review,” Artificial Intelligence Review, vol. 52, no. 2, pp. 803–855, 2018.

[26] T. T. Ademujimi, M. P. Brundage, and V. V. Prabhu, “A review of current machine learning techniques used in manufacturing diagnosis,” Advances in Production Management Systems. The Path to Intelligent, Collaborative and Sustainable Manufacturing, pp. 407–415, 2017.

[27] D. Qader Zeebaree, A. Mohsin Abdulazeez, D. Asaad Zebari, H. Haron, and H. Nuzly Abdull Hamed, “Multi-level fusion in ultrasound for cancer detection based on uniform LBP features,” Computers, Materials &amp; Continua, vol. 66, no. 3, pp. 3363–3382, 2021.

[28] M. Sazzadul Hoque, “An implementation of intrusion detection system using genetic algorithm,” International Journal of Network Security &amp; Its Applications, vol. 4, no. 2, pp. 109–120, 2012.

[29] R. Suraj, S. Tapaswi, S. Yousef, K. K. Pattanaik, and M. Cole, “Mobility prediction in Mobile ad hoc networks using a lightweight genetic algorithm,” Wireless Networks, vol. 22, no. 6, pp. 1797–1806, 2015.

[30] H. Gharaee and H. Hosseinvand, “A new feature selection ids based on genetic algorithm and SVM,” 2016 8th International Symposium on Telecommunications (IST), 2016.

[31] A. AlSukk, R. N. Khushaba, and A. Al-Ani, “Enhancing the diversity of genetic algorithm for improved feature selection,” 2010 IEEE International Conference on Systems, Man and Cybernetics, 2010.