
UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

# Malware detection using metadata and entropy of PE files and machine learning

Diploma Thesis

**Siavrakas Michail**

**Supervisor:** Katsaros Dimitrios

Month 2022


UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

# Malware detection using metadata and entropy of PE files and machine learning

Diploma Thesis

**Siavrakas Michail**

**Supervisor:** Katsaros Dimitrios

Month 2022


ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

# Ανίχνευση κακόβουλου λογισμικού με χρήση μεταδεδομένων και εντροπίας φορητών εκτελέσιμων και μηχανικής μάθησης

Διπλωματική Εργασία

**Σιαβρακάς Μιχαήλ**

**Επιβλέπων/πουσα:** Κατσαρός Δημήτριος

Μήνας 2022

Approved by the Examination Committee:

Supervisor **Katsaros Dimitrios**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Rafailidis Dimitrios**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Thanos Georgios**
Laboratory Teaching Staff Member, Department of Electrical and Computer Engineering, University of Thessaly

# Acknowledgements

Thanks to my family for their efforts and their continuous support.

# DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Siavrakas Michail

Diploma Thesis

**Malware detection using metadata and entropy of PE files and machine learning**

**Siavrakas Michail**

# Abstract

The struggle against malware is a continuous one, as those files get more sophisticated over time in order to trick analysts and software, an evolution that seems to never stop. For the purpose of malware detection, machine learning could provide a powerful answer. By this perspective, a multimodal approach is proposed on this paper, trained and tested on data obtained from Kaggle and the malware security partner of Meraz'18 - Annual Techno Cultural festival of IIT Bhilai. Neural networks were developed for the two modes and then an extracted feature vector, was used as an input for different machine learning algorithms. The best results taken were an accuracy of 96.4% and an f-1 score of 93.7%.

**Keywords:**

malware, detection, machine, learning, multimodal

Διπλωματική Εργασία

**Ανίχνευση κακόβουλου λογισμικού με χρήση μεταδεδομένων και εντροπίας φορητών εκτελέσιμων και μηχανικής μάθησης**

**Σιαβρακάς Μιχαήλ**

# Περίληψη

Η προσπάθεια καταπολέμησης του κακόβουλου λογισμικού είναι διαρκής, καθώς τέτοιου τύπου αρχεία γίνονται πιο εκλεπτυσμένα με την πάροδο του χρόνου ώστε να ξεγελάσουν αναλυτές και λογισμικά, μια εξέλιξη που φαίνεται να μην έχει τελειωμό. Μία δυναμική απάντηση στο θέμα της ανίχνευσης κακόβουλου λογισμικού, θα μπορούσε να δοθεί απο τή μηχανική μάθηση. Υπό αυτό το πρίσμα, σε αυτή την εργασία παρουσιάζεται μια πολυτροπική προσέγγιση, εκπαιδευμένη και δοκιμασμένη σε δεδομένα από το Kaggle και το συνεργάτη του Meraz'18 - Annual Techno Cultural festival of IIT Bhilai. Για τις δύο εισόδους φτιάχτηκαν νευρωνικά δίκτυα και ύστερα εξήχθη από αυτά ένα διάνυσμα, το οποίο χρησιμοποιήθηκε ως είσοδος για διάφορους αλγορίθμους μηχανικής μάθησης. Τα καλύτερα αποτελέσματα ήταν ακρίβεια 96.4% επί του συνόλου και f-1 απόδοση 93.7%.

**Λέξεις-κλειδιά:**

κακόβουλο, λογισμικό, μηχανική, μάθηση, ανίχνευση

# Table of contents

# Abbreviations

| | |
|---|---|
| CNN | Convolutional Neural Network |
| SVD | Singular Value Decomposition |
| PE | Portable Executable |
| MLP | Multi Layer Perceptron |
| LR | Logistic Regression |
| SVM | Support Vector Machine |
| KNN | K Nearest Neighbours |
| GBC | Gradient Boosting Classifier |
| XGBC | Extreme Gradient Boosting Classifier |
| RFC | Random Forest Classifier |
| DTC | Decision Tree Classifier |
| ETC | Extra Trees Classifier |

# Chapter 1

# Introduction

## 1.1 Context

Malware can be described as any piece of software designed to harm its host. Malware files can be clustered in types such as worms, viruses, Trojan horses, adware, spyware and ransomware. In recent years, a great variety of malicious files has been detected. By analyzing these files, it was determined that they were based on already known malware with some modifications. New variants which are able to elude detection from antivirus systems, demonstrate the continuous evolution of malicious software. One can see the extent of the malware problem in [1].

Considering that computer systems can be found everywhere, the problem of malware attacks is of vital importance. In addition, humanity is entering the digital era, an introduction accelerated by real world events, leaving many individuals unprepared. For example, people have been falling victims of different cyber scams over the years. On top of that, we saw at the beginning of the pandemic, that businesses struggled to keep their data intact from third parties, as they did not cope with the cyber security demands. We can see that this new era brings forth problems with consequences that should be resolved before they become even greater.

Since malware evolves that rapidly, current research has turned to machine learning, a field of computer science that evolves at the same pace. The detection step is considered the most important in the malware defence process, and machine learning algorithms have been already used as it can be seen in [2].

## 1.2 Motivation & Contribution

In order to protect personal and sensitive corporate data, there have been great efforts in the field of malware detection. While traditional techniques such as static and dynamic analysis have served us well, they have disadvantages that can be clearly exploited by the latest versions of malware. That said, machine learning steps in, in order to provide modern solutions.

In general, malware detection is considered a multi-modal problem, since the more information one has, the more precise he can be on whether a particular file is malicious or not. In this thesis, a multi-modal approach is examined. A feature vector is extracted from two neural networks trained on the modalities of Portable Executable (PE) files and code entropy statistics, in order to be used as input on different machine learning algorithms. The feature vector is constructed by the method of intermediate fusion for the purpose of combining the intermediate features generated by the models.

This thesis aspires to contribute in the examination of malware detection on multiple modalities and the examination of different machine learning methods in ensemble learning.

## 1.3 Outline

In the rest of this thesis, a brief summary of the related work will be presented in chapter 2, the data preparation and the methods used will be presented in chapter 3, the data sets used for training and testing, the performance measures and the evaluation will be presented in chapter 4 and finally the conclusions and the future work will be presented in chapter 5.

# Chapter 2

# Related Work

Some complete frameworks of machine learning models with a multi-modal approach have already been developed, with the purpose of malware detection/classification. The framework Orthrus is proposed in [3], as a bimodal deep learning architecture with the purpose of classification of malware. The hexadecimal representation of the malware binary, along with the assembly instructions are fed as inputs to a Convolutional Neural Network and then two n-gram like features are generated by a process called intermediate fusion. In intermediate fusion we merge features from intermediate layers of the neural networks for each mode. This way, the manual feature generation was avoided as it was automatically done by the network, and by combining the most dominant features we have avoided any drawbacks related with classification on only one of the modalities.

The HYDRA framework [4] is an end to end framework for malware detection. By feeding the list of the API function calls, the assembly instructions and the hexadecimal representation of the malware binary to a feed forward network, a convolutional and a DeepConv network respectively, the three outputs combined give us a feature vector. This feature vector will then be used to classify an executable to its appropriate class.

A similar approach was followed by the Chimera framework [5] but for Android malware detection. In this work, syscall sequences, dex images, intents & permission were fed to two convolutional networks and a feed-forward network respectively, and then the extracted feature vectors were merged to form a bigger one, which would act as an input to another feed-forward neural network.

In [6] we see another intermediate fusion solution for malware classification, with PE metadata along with PE import data as inputs to a feed-forward neural network and opcodes

as input to a convolutional neural network. Then the outputs of the two are combined to form a feature vector.

[7] propose the most effective feature combination from different modalities, to create a feature vector that will help the classifier of their choice. The accuracy given by this method points out the importance of the feature vector given to the classifier.

[8] approached the problems of malware detection and classification with a hybrid method. They tried to create a feature vector with characteristics obtained from both static and dynamic API call sequences. The implementation was done based on the contribution of each feature on an attempt to increase variance between the samples. The results of the hybrid approach showed an improvement of over 1.5% for the detection problem and a tremendous improvement on the classification problem as opposed to a static or a dynamic approach.

[9] state that obfuscated malware detection rate can increase with a multi-modal approach. Thus, they created MalInsight an early fusion machine learning model which takes as an input a feature vector composed of PE sections, API and DLL calls and the file's networks behaviour.

[10] show that a model that the multi-modal approach state the problem of robustness, by comparing the classification results between old and new malware. They also state that the difference observed between the new and old malware data sets becomes less significant if the two are mixed together. Their hybrid method outcompetes the results obtained from each mode separately.

[11] is occupied with the classification of malware in different families by using the modality of PE metadata, making an extended report to this type of data.

[12] used an early fusion method. They tried to transfer the problem from the research area to the field, in order to see how well the model performs on real time and compare it with academic results.

The results of the above-mentioned papers make it difficult to understand if early, intermediate, or late fusion is better suited for the malware classification or detection problem, though in this thesis an intermediate fusion approach is followed since it allowed more flexibility.

# Chapter 3

# Models & Approach

## 3.1 Data Preprocessing

The data set used was found in a Kaggle competition. It contains PE metadata,entropy measures from different sections of Windows PE files, and the hexadecimal representation of the malware binary, stored in tabular form. The Kaggle-data.csv, containing 216353 file samples was split in a training and test set. The training set was comprised of 78309 tuples while the rest were used as the test set. The test set was bigger than the training set in order to see how well the model generalizes and it contains 96724 malware and 41333 benign -previously unseen- samples

The PE metadata are extracted from the header of PE files. These files give instructions to the operating system on how to execute the file.

The entropy statistics are extracted by statistically analyzing each file and each code section of the file. Obfuscation techniques, such as packing, are used by malware authors in order to trick the static analysis of the file's features. By doing that though, they increase the entropy of the file, or that of a particular code section.

The PE metadata and the entropy statistics parts will be used, so we split the dataset in two, accordingly. After that, the features that should be used from the overall of each modality had to be chosen, so 6 of the entropy statistics were dropped. Now, the two modalities are ready to be scaled.

The PE metadata part was standardized first. Then, the mean was subtracted from each feature, and after extracting the U matrix from the Singular Value Decomposition (SVD) factorization, the final form of the data set was formed by the product of the U matrix and

the data we had up to this point. The entropy part was simply standardized and then the mean was subtracted from each feature. This concludes the data pre-processing procedure.

## 3.2 Class Imbalance

A common problem with data sets for machine learning algorithms is the class imbalance. Specifically in problems regarding malware detection, legal restrictions protect legitimate files from being used in public researches of this purpose. On the other hand, malware files are widespread by sites of interest, as mentioned in [2] and [4].

This problem is encountered on this data set too, and since multiple algorithms were used in order to determine if the sample is malicious or not, the class_weight parameter was adjusted for some of them in order to deal with the problem.

## 3.3 Modalities & MLPs

Each modality is provided as an input to a Multi Layer Perceptron neural network. The architecture of the PE metadata neural network is the following:

- Dense layer of 32 units and a tanh activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 64 units and a tanh activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 64 units and a tanh activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 32 units and a tanh activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 1 unit with a sigmoid activation function.

The optimizer picked was the Stochastic Gradient Descent and its parameters were a learning rate of 0.01 and decay of 1e-6. The loss function which was selected was the binary

cross-entropy, as we had to do with a binary classification problem. The model was trained for 50 epochs with a batch size equal to 32.

The architecture for the entropy statistics neural network is the following:

- Dense layer of 25 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 50 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 100 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 200 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 100 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 50 units and a relu activation function,
- Dropout layer of 0.3 probability,
- Dense layer of 1 unit with a sigmoid activation function.

The optimizer picked was the Adam optimizer and its parameters were a learning rate of 0.001 and decay of 1e-6. The loss function which was selected was the mean absolute error. The model was trained for 50 epochs with a batch size equal to 40.

## 3.4 Intermediate Fusion

For the intermediate fusion part the pre-trained neural sub-networks were taken, and the outputs of their hidden layers were concatenated in such way that the variation introduced from the calculations of the feed forward networks could be epxloited. The combinations were checked exhaustively and the best results were given by that combination of these outputs:

- The output of the second hidden layer of the PE metadata side,
- The output of the last hidden layer of the entropy side,
- The output of the feed-forward network of the entropy side.

Note that for the Convolutional Neural Network (CNN), a dense layer of 256 units was used in order to enlarge that feature vector.

## 3.5 Model Ensembling

The feature vector generated will be the input of different machine learning algorithms that will be used for the model ensembling, where a model takes as input the proposed feature vector generated by the two sub-models.

In general, ensembling methods could correct the mistakes of the trained sub-networks and the results are predictions that are less sensitive to the specifics of the training data and choice of training scheme.

For the first ensembling kind we use the 10 different methods presented below:

1. CNN with architecture of:
    - Conv2D layer of 100 with kernel size equal to (3, 3), a gray-scale image input of size 64x4 with a relu activation function,
    - MaxPooling2D layer with pool size equal to (2, 2),
    - Dense layer of 16 units with a relu activation function,
    - Dropout layer of 0.5 probability,
    - Dense layer of 1 unit with a sigmoid function.

    The optimizer picked was the Stochastic Gradient Descent and its parameters were a learning rate of 0.01, a decay of 1e-6. The loss function which was selected was the mean absolute error. The model was trained for 30 epochs with a batch size equal to 32. The class_weight parameter was {0: 1., 1: 3.}.

2. Multi Layer Perceptron (MLP) with architecture of:
    - Dense layer of 115 units with a tanh activation function,

- Dense layer of 100 units with a tanh activation function,
- Dense layer of 150 units with a tanh activation function,
- Batch Normalization layer,
- Dense layer of 75 units with a tanh activation function,
- Dropout layer of 0.2 probability,
- Dense layer of 1 unit with a tanh function.

The optimizer picked was the Stochastic Gradient Descent and its parameters were a learning rate of 0.01 and decay of 1e-6. The loss function which was selected was the mean absolute error. The model was trained for 20 epochs with a batch size equal to 70. The class_weight parameter was {0: 8., 1: 6.}.

3. The CSupport Vector Machine (SVM) with the C parameter equal to 0.1,
4. The K-Nearest Neighbours (KNN) with the number of neighbours to be considered equal to 70.
5. The Random Forest Classifier (RFC) with 100 estimators and max_depth=10,
6. The Decision Tree Classifier (DTC) with max_depth=100 and min_samples_split=2,
7. The L1-regularized Logistic Regression (LR) with 500 iterations, the liblinear as a solver, warm start and the class_weight parameter set as {0: 2., 1: 1.},
8. The Gradient Boosting Classifier (GBC) with 100 estimators, a learning rate of 0.1 and max_depth=2,
9. The Extended Gradient Boosting Classifier (XGBC) with 50 estimators, max_depth=1, objective=logitraw,
10. The Extra Trees Classifier (ETC) with 100 estimators, max_depth=10, min_samples_split=3.

# Chapter 4

# Performance Evaluation

The evaluation of all the methods happened on the metrics of accuracy and f1. The f1 metric was chosen since it is a combination of the precision and recall measurements and gives more information regarding the performance of the occasional method on the true/false positive. In addition the False Positives (FP) and False Negatives (FN) will be presented in order to help us understand how well the model behaves.

The results for the multimodal method and for the two sub-models on the same algorithms except the CNN -with the parameters changed accordingly- are presented below:

| Method | Accuracy(%) | F1(%) | FN | FP |
|---|---|---|---|---|
| CNN | 95.9 | 92.8 | 914 | 4702 |
| MLP | 95.8 | 92.8 | 2059 | 3796 |
| SVM | 95.8 | 92.7 | 1820 | 4032 |
| KNN | 95.5 | 92.2 | 1948 | 4279 |
| RFC | 95.9 | 92.8 | 963 | 4739 |
| DTC | 73.6 | 56.8 | 19026 | 17401 |
| LR | 90.8 | 83.1 | 2661 | 10087 |
| GBC | 95.6 | 92.3 | 916 | 5132 |
| XGBC | 94.7 | 90.4 | 713 | 6617 |
| ETC | 95.5 | 92.1 | 983 | 5235 |

Table 4.1: Multimodal method performance

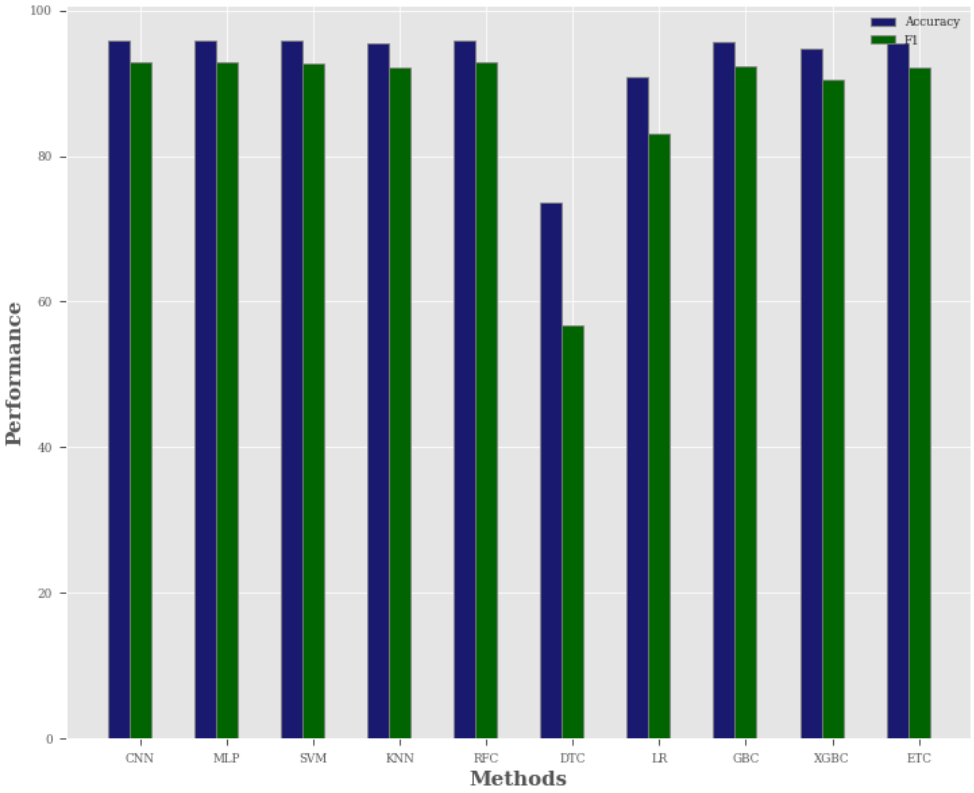


Figure 4.1: Multimodal method performance graph

| Method | Accuracy(%) | F1(%) | FN | FP |
|---|---|---|---|---|
| MLP | 84.4 | 70.2 | 5624 | 15937 |
| SVM | 73.4 | 63.5 | 27212 | 9471 |
| KNN | 80.9 | 70.7 | 16776 | 9589 |
| RFC | 85.2 | 68.0 | 790 | 19606 |
| DTC | 71.9 | 54.5 | 20633 | 18110 |
| LR | 63.7 | 49.7 | 33677 | 16499 |
| GBC | 84.1 | 68.3 | 4183 | 17700 |
| XGBC | 79.2 | 66.6 | 16039 | 12709 |
| ETC | 74.6 | 47.1 | 9355 | 25715 |

Table 4.2: Entropy side performance

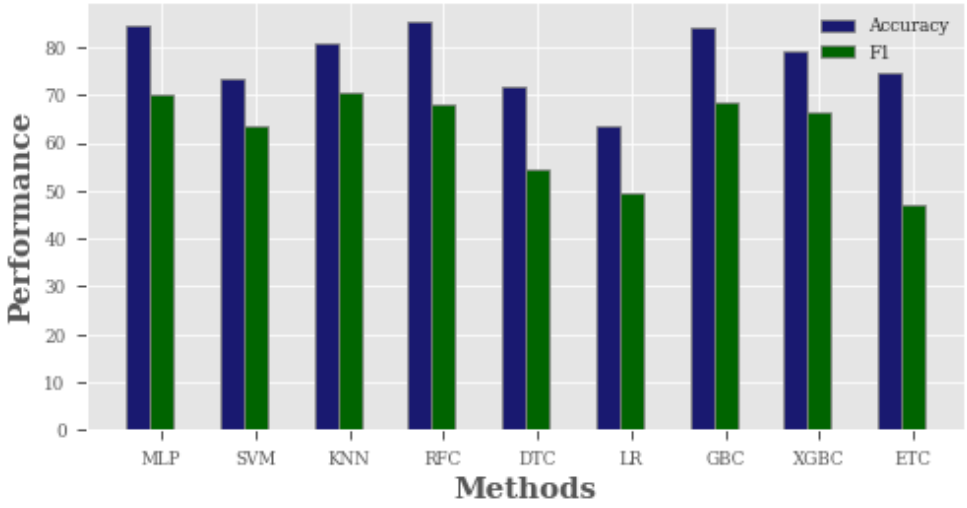


Figure 4.2: Entropy side performance graph

| Method | Accuracy(%) | F1(%) | FN | FP |
|---|---|---|---|---|
| MLP | 93.8 | 88.6 | 497 | 8087 |
| SVM | 96.4 | 93.7 | 1065 | 3930 |
| KNN | 95.5 | 91.9 | 687 | 5575 |
| RFC | 92.5 | 86.3 | 1750 | 8615 |
| DTC | 79.7 | 61.1 | 8734 | 19327 |
| LR | 63.7 | 49.7 | 33677 | 16499 |
| GBC | 94.0 | 89.9 | 3268 | 4948 |
| XGBC | 91.4 | 83.7 | 1112 | 10804 |
| ETC | 88.1 | 75.5 | 441 | 15997 |

Table 4.3: Metadata side performance

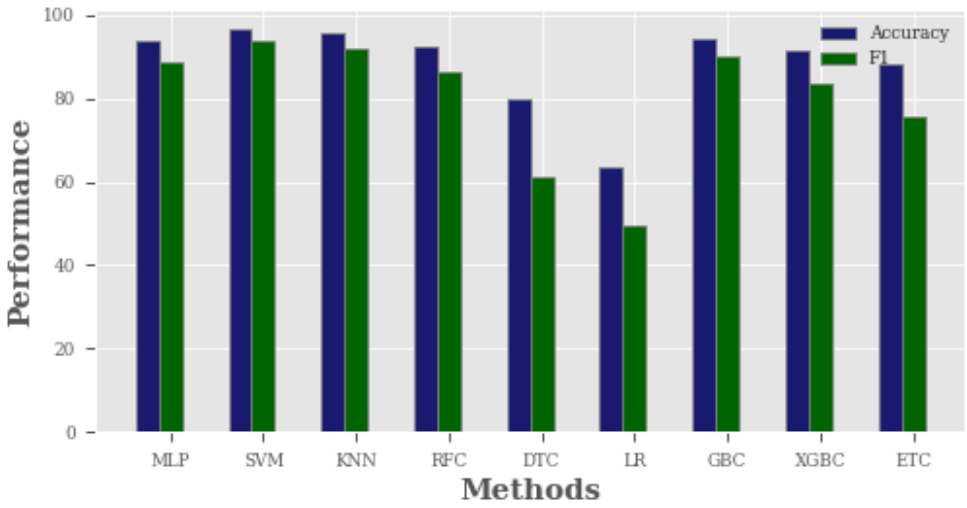


Figure 4.3: Metadata side performance graph

By comparing the results of this thesis with these of the Related Work 2, the impression that convolutional neural networks are stronger for malware detection/classification tasks is given. They provide more flexibility on the feature generation part, but they are computationaly expensive. In general, we can see an improvement to both metrics, correction to the sub-model biases and a stability in the results despite the chosen classifier. The neural networks seem to provide better solutions than the other classifiers. The best result was taken drom the PE metadata side and the results taken from the entropy side were poor, so it is assumed that the entropy statistics are a difficult feature to handle.

# Chapter 5

# Conclusion & Future Work

To conclude, this thesis had the purpose of malware detection of Windows PE files based on two modalities, that of the PE metadata and that of the entropy statistics. Two MLPs were developed for each one and after their training, the best suited for the purpose layer outputs were merged to provide a feature vector. This feature vector was fed to different machine learning algorithms. The two modalities were tested on almost all of the methods too. The best results were given from the first one with an accuracy of 96.4% and an f-1 measure of 93.7%, while the proposed method gave an accuracy of 95.9% and an f-1 measure of 92.8%. We can see that the two modalities separately are not very good in correctly classifying the benign samples, something that is being corrected by the multi-modal approach followed. In addition high percentages on both metrics are achieved despite the classifier, something perceived as a result of the feature vector that was created. In general, the method seems to correct the biases of the sub-models and achieving results close to the best for the majority of the classifiers.

For future work, more modalities will be added,since the hexadecimal representation of the malware can be visualized as a picture and can be used as input to a CNN, with performance improvement being the goal. Research on adversarial examples and how to trick machine learning algorithms for malware detection/classification will be done, as new or obfuscated malware appear. Unpacked malware should be examined since unpacking is a heavy procedure and packers are designed from malware authors to trick the models. Models should be retrained to check their robustness since malware authors can adapt to their abilities. Different methods for feature generation or extraction will be examined, possibly with self-organizing maps since it can help machine learning models perform better.

# Bibliography


[1] Ajit Kumar, Bong Jun Choi, K. S. Kuppusamy, and G. Aghila. Malware attacks: Dimensions, impact, and defenses. pages 157–179. Springer Science and Business Media Deutschland GmbH, 2022.

[2] Daniel Gibert, Carles Mateu, and Jordi Planes. The rise of machine learning for detection and classification of malware: Research developments, trends and challenges, 3 2020.

[3] IEEE Computational Intelligence Society, International Neural Network Society, Institute of Electrical, Electronics Engineers, and IEEE World Congress on Computational Intelligence (2020 : Online). *2020 International Joint Conference on Neural Networks (IJCNN) : 2020 conference proceedings.*

[4] Daniel Gibert, Carles Mateu, and Jordi Planes. Hydra: A multimodal deep learning framework for malware classification. *Computers and Security*, 95, 8 2020.

[5] Angelo S Oliveira and Renato J Sassi. Chimera: An android malware detection method based on multimodal deep learning and hybrid analysis a preprint, 2020.

[6] Bojan Kolosnjaji, Ghadir Eraisha, George Webster, Apostolis Zarras, and Claudia Eckert. Empowering convolutional networks for malware classification and analysis. volume 2017-May, pages 3838–3845. Institute of Electrical and Electronics Engineers Inc., 6 2017.

[7] Mansour Ahmadi, Dmitry Ulyanov, Stanislav Semenov, Mikhail Trofimov, and Giorgio Giacinto. Novel feature extraction, selection and fusion for effective malware family classification. pages 183–194. Association for Computing Machinery, Inc, 3 2016.

[8] Weijie Han, Jingfeng Xue, Yong Wang, Lu Huang, Zixiao Kong, and Limin Mao. Maldae: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics. *Computers and Security*, 83:208–233, 6 2019.

[9] Weijie Han, Jingfeng Xue, Yong Wang, Zhenyan Liu, and Zixiao Kong. Malinsight: A systematic profiling based malware detection framework. *Journal of Network and Computer Applications*, 125:236–250, 1 2019.

[10] Rafiqul Islam, Ronghua Tian, Lynn M. Batten, and Steve Versteeg. Classification of malware based on integrated static and dynamic features, 3 2013.

[11] Pham Van Hung. An approach to fast malware classification with machine learning technique.

[12] Matilda Rhode, Lewis Tuson, Pete Burnap, and Kevin Jones. Lab to soc: Robust features for dynamic malware detection. pages 13–16. Institute of Electrical and Electronics Engineers Inc., 6 2019.