Kitsune Network Attack Classification Using Machine Learning

Kyungbin Lee, Nahid Ebrahimi Majd

Department of Computer Science and Information System

California State University San Marcos, United States

lee305@csusm.edu, nmajd@csusm.edu

*Abstract*— The rise of network attack has emerged as a pressing concern for the technology industry. Either IP-based commercial surveillance system or a network full of IoT devices are required to capture network attack. Most of the prior network attack detection techniques either are signature-based, which are inefficient to identify new type of network attack, or utilize a dynamic analysis, which are complicated and computationally expensive. This paper proposes a feature selection-based framework along with different machine learning algorithms that can effectively detect ransomware based on features extracted from the files. We performed filter methods of feature selection and then applied Decision Tree (DT), Random Forest (RF), Naïve Bayes (NB), Logistic Regression (LR), Support Vector Machine (SVM), k-Nearest Neighbor (KNN), Extreme Gradient Boost (XGB), Gradient Boosting (GB) and Extra Trees (ET) on a kitsune network attack dataset that contains the features and label from files. The experimental results demonstrate that DT, RF, XGB, SVM amd ET classifiers with ANOVA filter method of feature selection outperform other methods in terms of accuracy, precision, and recall.

Keywords— Ransomware Classification; Network security; Feature Selection; Machine Learning;

# Introduction

The number of network attacks on computer system has been increasing over years. In technology industry, network attack on businesses have become increasing concern. It is important to build the system to detect various type of network attack. A successful network attack can have serious implications, including the loss of crucial data, financial losses, reputational damage, and legal liability.

## Dataset

We used a dataset extracted from ‘Kitsune Network Attack.’ The dataset is a collection of nine network attack datasets captured from a either an IP-based commercial surveillance system or a network full of IoT devices. Each dataset contains millions of network packets and differed network attack within it. We extracted sampling dataset from original dataset, there are 9 network attacks, and each attack includes 6500 benign and 6500 malicious rows. For training set, we used total 117000 rows.

## Data preprocessing

There were no missing values or duplicates in the dataset, therefore, minimal preprocessing was required. Since attributes Name or md5 did not contain important information that can be used to predict the class, we dropped these fields. All other attributes were numerical; therefore, no encoding was required.

**1. Dataset split:** We split the dataset into training data and test data in the ratio of 70:30. The resulting training dataset had 81900 samples and the test dataset had 31900 samples. We do not create a validation dataset since we use 5-fold cross validation to generalize the models.

**2. Standardization:** Normalization technique was used to convert each of the variables into a similar scale by centering each variable at zero with a standard deviation of 1.

After the data preprocessing step, we applied feature selection techniques to train the dataset and fed the transformed data to various ML algorithms.

# The Feature Selection Techniques

Feature selection aims to identify a subset of input variables that can effectively represent the input data, minimize the impact of noise or irrelevant variables, and achieve accurate prediction outcomes. I used Filter feature selection method.

The filter method is based on statistical tests or correlation measures that assess the degree of association between each feature and the target variable. In ANOVA (Analysis of Variance) F-value based feature selection, we calculate the F-value for each feature by comparing the variance of the target variable explained by the feature and the variance of the target variable that is not explained by the feature. For comparison to other results, we used all features in Kitsune Network Attack dataset. In Table 1, we list number, name and score of all features from highest score to lowest score.

Table 1: All features F-values applying ANOVA F-test feature selection.

|  |  |  |
| --- | --- | --- |
| **Feature** | **Name** | **Score** |
| Feature 49 | HH\_L1\_radius | 13502.728850635633 |
| Feature 42 | HH\_L3\_radius | 13496.757285304384 |
| Feature 102 | HpHp\_L0.1\_weight | 13470.305291577235 |
| Feature 35 | HH\_L5\_radius | 13408.138259447853 |
| Feature 40 | HH\_L3\_std | 12821.612847277936 |
| Feature 47 | HH\_L1\_std | 12805.166998930708 |
| Feature 33 | HH\_L5\_std | 12764.10547601086 |
| Feature 56 | HH\_L0.1\_radius | 12556.895291258925 |
| Feature 95 | HpHp\_L1\_weight | 12313.862233927848 |
| Feature 54 | HH\_L0.1\_std | 11892.144859446758 |
| Feature 88 | HpHp\_L3\_weight | 11337.088295503863 |
| Feature 18 | H\_L5\_variance | 11162.768639061092 |
| Feature 63 | HH\_L0.01\_radius | 11161.25372944349 |
| Feature 3 | MI\_dir\_L5\_variance | 11151.170047414867 |
| Feature 21 | H\_L3\_variance | 11094.136017077044 |
| Feature 6 | MI\_dir\_L3\_variance | 11085.113410039732 |
| Feature 9 | MI\_dir\_L1\_variance | 10832.675886539131 |
| Feature 24 | H\_L1\_variance | 10819.500296130498 |
| Feature 15 | MI\_dir\_L0.01\_variance | 10319.710972788054 |
| Feature 109 | HpHp\_L0.01\_weight | 10311.665010052213 |
| Feature 81 | HpHp\_L5\_weight | 10132.299906129792 |
| Feature 12 | MI\_dir\_L0.1\_variance | 9266.85709507781 |
| Feature 61 | HH\_L0.01\_std | 9143.103009680352 |
| Feature 27 | H\_L0.1\_variance | 9011.098488421185 |
| Feature 30 | H\_L0.01\_variance | 8350.46275297705 |
| Feature 62 | HH\_L0.01\_magnitude | 7771.545558509689 |
| Feature 16 | H\_L5\_weight | 7619.452867729353 |
| Feature 1 | MI\_dir\_L5\_weight | 7611.145095141901 |
| Feature 66 | HH\_jit\_L5\_weight | 7592.988308051574 |
| Feature 31 | HH\_L5\_weight | 7592.987812912864 |
| Feature 55 | HH\_L0.1\_magnitude | 7472.791814397792 |
| Feature 19 | H\_L3\_weight | 7466.28452360348 |
| Feature 4 | MI\_dir\_L3\_weight | 7454.602227287389 |
| Feature 69 | HH\_jit\_L3\_weight | 7445.272298193304 |
| Feature 38 | HH\_L3\_weight | 7445.272003243772 |
| Feature 22 | H\_L1\_weight | 7076.0901079186015 |
| Feature 72 | HH\_jit\_L1\_weight | 7062.060299307608 |
| Feature 45 | HH\_L1\_weight | 7062.060204764576 |
| Feature 7 | MI\_dir\_L1\_weight | 7051.696151012256 |
| Feature 48 | HH\_L1\_magnitude | 6841.029646791543 |
| Feature 41 | HH\_L3\_magnitude | 6721.887708476153 |
| Feature 34 | HH\_L5\_magnitude | 6669.258483079881 |
| Feature 105 | HpHp\_L0.1\_magnitude | 5791.949044062374 |
| Feature 98 | HpHp\_L1\_magnitude | 5781.835830541465 |
| Feature 112 | HpHp\_L0.01\_magnitude | 5781.404866798354 |
| Feature 91 | HpHp\_L3\_magnitude | 5767.165117353031 |
| Feature 84 | HpHp\_L5\_magnitude | 5750.934171768116 |
| Feature 25 | H\_L0.1\_weight | 5476.94450183509 |
| Feature 75 | HH\_jit\_L0.1\_weight | 5461.018641222112 |
| Feature 52 | HH\_L0.1\_weight | 5461.018636574525 |
| Feature 10 | MI\_dir\_L0.1\_weight | 5402.724532832011 |
| Feature 65 | HH\_L0.01\_pcc | 4261.2873207294815 |
| Feature 64 | HH\_L0.01\_covariance | 3999.1523925974157 |
| Feature 28 | H\_L0.01\_weight | 3980.9902850541257 |
| Feature 59 | HH\_L0.01\_weight | 3899.152650586466 |
| Feature 78 | HH\_jit\_L0.01\_weight | 3899.1526504348453 |
| Feature 13 | MI\_dir\_L0.01\_weight | 3495.0502566928935 |
| Feature 46 | HH\_L1\_mean | 3334.397898904083 |
| Feature 53 | HH\_L0.1\_mean | 3320.6371519922045 |
| Feature 39 | HH\_L3\_mean | 3317.5809050902308 |
| Feature 32 | HH\_L5\_mean | 3302.3747789492327 |
| Feature 2 | MI\_dir\_L5\_mean | 3288.072588268256 |
| Feature 5 | MI\_dir\_L3\_mean | 3287.8656483502914 |
| Feature 17 | H\_L5\_mean | 3285.2973526982532 |
| Feature 20 | H\_L3\_mean | 3284.7885189523417 |
| Feature 8 | MI\_dir\_L1\_mean | 3252.752098545946 |
| Feature 23 | H\_L1\_mean | 3247.9914923171978 |
| Feature 60 | HH\_L0.01\_mean | 3092.163064714781 |
| Feature 11 | MI\_dir\_L0.1\_mean | 2916.7594399667314 |
| Feature 26 | H\_L0.1\_mean | 2885.732463114174 |
| Feature 14 | MI\_dir\_L0.01\_mean | 2527.768750618433 |
| Feature 103 | HpHp\_L0.1\_mean | 2455.7492513753887 |
| Feature 96 | HpHp\_L1\_mean | 2450.083785077427 |
| Feature 110 | HpHp\_L0.01\_mean | 2449.0480716707534 |
| Feature 89 | HpHp\_L3\_mean | 2440.3815577541986 |
| Feature 82 | HpHp\_L5\_mean | 2430.052123996081 |
| Feature 29 | H\_L0.01\_mean | 2307.6924039766745 |
| Feature 58 | HH\_L0.1\_pcc | 2045.4795036420855 |
| Feature 111 | HpHp\_L0.01\_std | 1852.8246566087314 |
| Feature 104 | HpHp\_L0.1\_std | 1825.847559215551 |
| Feature 97 | HpHp\_L1\_std | 1816.5828416027907 |
| Feature 90 | HpHp\_L3\_std | 1721.6772588292383 |
| Feature 83 | HpHp\_L5\_std | 1630.1601353206142 |
| Feature 113 | HpHp\_L0.01\_radius | 1225.4641063540075 |
| Feature 106 | HpHp\_L0.1\_radius | 1200.9552148002322 |
| Feature 99 | HpHp\_L1\_radius | 1187.2420195096793 |
| Feature 57 | HH\_L0.1\_covariance | 1110.4046077103637 |
| Feature 92 | HpHp\_L3\_radius | 1079.0828307450943 |
| Feature 85 | HpHp\_L5\_radius | 985.2017485880364 |
| Feature 79 | HH\_jit\_L0.01\_mean | 742.5786700375953 |
| Feature 115 | HpHp\_L0.01\_pcc | 741.8886011941507 |
| Feature 80 | HH\_jit\_L0.01\_variance | 685.296481452748 |
| Feature 76 | HH\_jit\_L0.1\_mean | 623.3389978824373 |
| Feature 77 | HH\_jit\_L0.1\_variance | 583.0854436766899 |
| Feature 73 | HH\_jit\_L1\_mean | 328.64154476606967 |
| Feature 70 | HH\_jit\_L3\_mean | 291.9953168254289 |
| Feature 67 | HH\_jit\_L5\_mean | 291.5417461021846 |
| Feature 74 | HH\_jit\_L1\_variance | 256.4486039593032 |
| Feature 100 | HpHp\_L1\_covariance | 245.19005542463913 |
| Feature 114 | HpHp\_L0.01\_covariance | 119.61052212300294 |
| Feature 107 | HpHp\_L0.1\_covariance | 77.0383178085574 |
| Feature 101 | HpHp\_L1\_pcc | 63.5024161517666 |
| Feature 94 | HpHp\_L3\_pcc | 34.03497108908622 |
| Feature 43 | HH\_L3\_covariance | 33.401557826311745 |
| Feature 100 | HpHp\_L1\_covariance | 32.009675738252156 |
| Feature 71 | HH\_jit\_L3\_variance | 27.772789813618466 |
| Feature 36 | HH\_L5\_covariance | 23.15405988977655 |
| Feature 50 | HH\_L1\_covariance | 21.929526732817404 |
| Feature 87 | HpHp\_L5\_pcc | 20.762446281266012 |
| Feature 68 | HH\_jit\_L5\_variance | 15.011759475441615 |
| eature 93 | HpHp\_L3\_covariance | 14.079435774625537 |
| Feature 86 | HpHp\_L5\_covariance | 5.765141390952015 |
| Feature 44 | HH\_L3\_pcc | 2.392538121018022 |
| Feature 37 | HH\_L5\_pcc | 1.846865117635773 |
| Feature 51 | HH\_L1\_pcc | 0.041463482383639214 |

# Machine Learning Models

The following 9 supervised Machine Learning algorithms were used to create models for filter method feature selection. Hyperparameter tuning of each model was done using 5-fold cross validation. The final hyperparameters used are listed in Table 2. We use GridSearch to find the best hyperparameter for each model. From the GridSearch results, we get different hyperparameter from other research paper. We use our best hyperparameter for training models

Table 2: ML models with their Hyperparameters.

|  |  |
| --- | --- |
| **ML algorithm** | **Hyperparameter** |
| LG | C=100, penalty=‘none’, solver=’newton-cg’ |
| DT | criterion=’entropy’, max\_depth=20, min\_sample\_leaf=9 |
| RF | criterion=’entropy’, max\_depth=20, n\_estimators=90 |
| GB | learning\_rate=0.1, n\_estimators=90 |
| SVM | C=1000, kernel=’rbf’, gamma=3.0 |
| KNN | n\_neighbors=6 |
| XGB | learning\_rate=0.1, n\_estimators=90 |
| NB | priors=’none’, var\_smoothing=0.1 |
| ET | criterion=’entropy’,max\_depth=90, n\_estimators=30 |

# Results and Discussion

After training and testing our models, we investigated the results for each individual model as well as in comparison to other research paper we found. Table 3 illustrates the results we achieved from our models. We compare 4 models (DT, RF, SVM and KNN) to other research paper results. Other research paper uses same size of sampling dataset. Other research excludes Feature 33 for training models whereas we trained our models with all features. Table 4 illustrates the comparison to other results. Our DT, RF and KNN models achieved lower accuracy than other results whereas our SVM model achieved higher accuracy than other results. We trained other ML algorithms such as GB, XGB, LG, NB and ET. Our XGB and GB get high accuracy compared to LG and NB. We searched what feature selection method and how many features improve our ML models. We trained each model from 15 features to 110 features based on ANOVA F-value feature selection and chi-squared feature selection method. In Table 5, We get higher accuracy for each model. Table 5 shows that each ML models get higher accuracy when they train and test with the best number of features from the best feature selection method.

Table 3: Results of my proposed models

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature Selection** | **Model No** | **Model** | **No of Feature** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **FP** | **FN** | **TP** | **TN** |
| Filter | 1 | LG | All | 0.796 | 0.89 | 0.68 | 0.77 | 1514 | 5640 | 11933 | 16013 |
| 2 | DT | All | 0.959 | 0.95 | 0.97 | 0.96 | 881 | 535 | 17038 | 16646 |
| 3 | RF | All | 0.945 | 0.95 | 0.94 | 0.95 | 796 | 1118 | 16455 | 16731 |
| 4 | GB | All | 0.937 | 0.96 | 0.92 | 0.94 | 742 | 1468 | 16105 | 16785 |
| 5 | SVM | All | 0.948 | 0.94 | 0.95 | 0.95 | 1093 | 847 | 16726 | 16434 |
| 6 | KNN | All | 0.901 | 0.91 | 0.89 | 0.90 | 1612 | 1849 | 15724 | 15915 |
| 7 | XGB | All | 0.968 | 0.96 | 0.98 | 0.97 | 749 | 363 | 17210 | 16778 |
| 8 | NB | All | 0.639 | 0.79 | 0.38 | 0.51 | 1731 | 10935 | 6638 | 15796 |
| 9 | ET | All | 0.963 | 0.96 | 0.97 | 0.96 | 684 | 609 | 16964 | 16843 |
| 10 | LG | 60 | 0.777 | 0.87 | 0.966 | 0.75 | 1791 | 6014 | 11559 | 15736 |
|  | 11 | DT | 60 | 0.959 | 0.95 | 0.97 | 0.96 | 868 | 581 | 16659 | 16992 |
|  | 12 | RF | 60 | 0.967 | 0.96 | 0.98 | 0.97 | 734 | 434 | 17139 | 16793 |
|  | 13 | GB | 60 | 0.937 | 0.96 | 0.91 | 0.94 | 650 | 1549 | 16024 | 16877 |
|  | 14 | SVM | 60 | 0.940 | 0.931 | 0.95 | 0.94 | 1241 | 853 | 16720 | 16286 |
|  | 15 | KNN | 60 | 0.908 | 0.91 | 0.91 | 0.91 | 1592 | 1629 | 15944 | 15935 |
|  | 16 | XGB | 60 | 0.965 | 0.96 | 0.97 | 0.96 | 657 | 576 | 16997 | 16870 |
|  | 17 | NB | 60 | 0.674 | 0.97 | 0.36 | 0.52 | 193 | 11310 | 6263 | 17334 |
|  | 18 | ET | 60 | 0.963 | 0.96 | 0.96 | 0.96 | 674 | 641 | 16932 | 16853 |

Table 4: Comparison our proposed models to other research results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | **Key Approach** | **ML Model** | **Our Research** | **Other Research** |
| Kitsune Network Attack Dataset | Binary Classification | DT | 0.959 | 0.999 |
| RF | 0.945 | 0.999 |
| SVM | 0.948 | 0.713 |
| KNN | 0.901 | 0.999 |

Table 5: Improved result according to No. of feature selection and method

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature Selection** | **Model No** | **Model** | **Method and**  **No of Feature** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **FP** | **FN** | **TP** | **TN** |
| Filter | 1 | LG | ANOVA, all | 0.796 | 0.89 | 0.68 | 0.77 | 1514 | 5640 | 11933 | 16013 |
| 2 | DT | ANOVA, 90 | 0.960 | 0.95 | 0.97 | 0.96 | 832 | 560 | 17013 | 16695 |
| 3 | RF | ANOVA, 90 | 0.969 | 0.96 | 0.98 | 0.97 | 722 | 389 | 17184 | 16805 |
| 4 | GB | chi-squared, 100 | 0.938 | 0.96 | 0.92 | 0.94 | 697 | 1479 | 16094 | 16830 |
| 5 | SVM | ANOVA, all | 0.948 | 0.94 | 0.95 | 0.95 | 1093 | 847 | 16726 | 16434 |
| 6 | KNN | ANOVA, 80 | 0.909 | 0.91 | 0.91 | 0.91 | 1532 | 1662 | 15911 | 15995 |
| 7 | XGB | ANOVA, 100 | 0.969 | 0.96 | 0.98 | 0.97 | 755 | 342 | 17231 | 16772 |
| 8 | NB | ANOVA, 60 | 0.640 | 0.97 | 0.36 | 0.52 | 193 | 11310 | 6263 | 17334 |
| 9 | ET | ANOVA, 100 | 0.964 | 0.96 | 0.97 | 0.96 | 666 | 600 | 16973 | 16861 |

# Conclusion

With the increasing threat of various types of network attacks, it is important to develop a system that can effectively detect known and new forms of network attack. In this paper, we designed models and performed extensive experiments by filtering method feature selection and ML algorithms along with hyperparameter tuning to achieve best results. The experimental results demonstrates that DT, RF, XGB SVM and ET with ANOVA perform better than other ML models. It is the best choice to use DT, RF, XGB SVM and ET for network attack detection.

# References

[1] Kim, YeaSul, YeEun Kim, & Hwankuk Kim. "A Comparison Experiment of Binary Classification for Detecting the GTP Encapsulated IoT DDoS Traffics in 5G Network." *Journal of Internet Technology* [Online], 23.5 (2022): 1049-1060. Web. 12 Jul. 2023

[2] Y.Mirsky, T. Doitshman, Y.Elovici, and A. Shabtai, “Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection,” arXiv.org, May 27, 2018. http://arxiv.org/abs/1802.09089 13 Jul. 2