Comparison of Machine Learning Algorithms and Their Ensembles for Botnet Detection

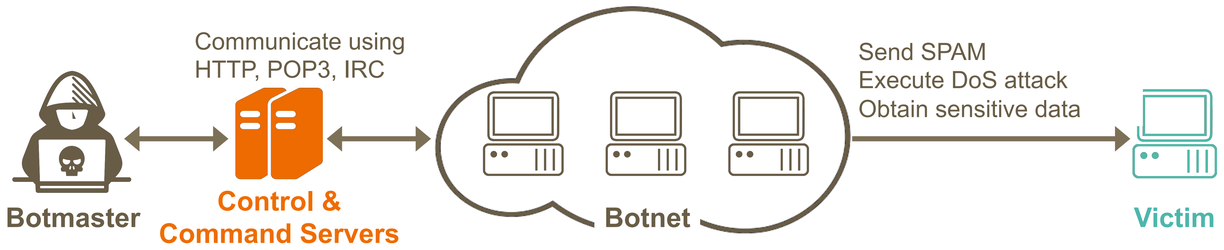
Songhui Ryu
Department of Computer and Information Technology, Purdue University, West Lafayette, IN 47907, USA
ryu26@purdue.edu

WHPC WOMEN IN HIGH PERFORMANCE COMPUTING PURDUE UNIVERSITY

@purduewhpc

INTRODUCTION

Botnet is a network of compromised devices controlled by a **Botmaster** for malicious tasks. **Botnet Detection,** like Intrusion Detection System, can be done by monitoring observable botnet behaviors in network traffics. Machine learning techniques, nowadays, have been popularly used to detect abnormal traffics.



1. Botmaster generates traffic while communication
2. DNS traffic to C&C server often remain unchanged
3. System/library calls are made by Botnet

Figure 1. A centralized Botnet architecture and its three different behaviors

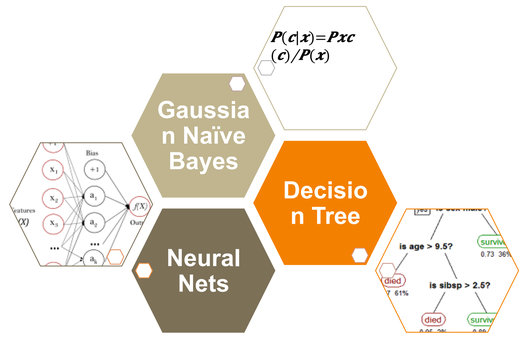
Botnet detection systems may use one or more machine learning algorithms, **but would making use of them together really make a difference?**

METHODOLOGY: ALGORITHMS & METRICS

Machine Learning Algorithms & Ensemble Methods

The most common machine learning (ML) algorithms for classification are *Gaussian Naïve Bayes (GNB), Neural Networks (NN) and Decision Tree (DT)*. These all are supervised learning which need labeled data.

Ensemble methods, which combine predictions of each ML classifier with or without weights, can be distinguished as below.



- **Voting:** As the simplest way, all different classifiers are trained separately with whole training data and its posterior probabilities are averaged.
- **Bagging:** It samples multiple random sub-datasets from the original dataset and feed them to each classifier to use Voting at the end.
- **Boosting:** To strengthen a classifier, Boosting incrementally builds an ensemble by training each model with the same dataset but weighted by error of the last prediction.

Accuracy Measurements

To compare accuracy of the algorithms, the following metrics are considered.

- **F1 score:** F1 score considers both the precision p and the recall r of the test. F1 score can be between 0 and 1 where 1 means its best value and 0 its worst.

$F1=2*p*r/p+r$ where $p=TP/TP+FP$, $r=TP/TP+FN$

- **Matthews correlation coefficient (MCC):** With true and false positives and negatives, MCC can be between -1 and +1 where +1 means a perfect prediction, 0 no better than random prediction and -1 and inverse prediction.

$MCC=TP*TN-FP*FN/\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}$

DATASET

The **CTU-13 dataset** is a public dataset featuring Botnet traffic mixed with normal and background traffic captured at the CTU university, Czech Republic in 2011. Among 13 different network traffic captures, #9, #10, and #11 were used.

Dataset	#9	#10	#11
Duration(hrs)	5.18	4.75	0.26
# Packets	115,415,321	90,389,782	6,337,202
Tot. size of Packets	94GB	73GB	5.2GB
# NetFlows	2,753,885	1,309,792	107,252
Tot. size of Netflows	1.5GB	980MB	74MB
# Botnet Flows	184,979 (6.68%)	106,352 (8.11%)	8,163 (7.60%)
# Normal Flows	43,340 (1.57%)	15,847 (1.2%)	2,718 (2.53%)
# Background Flows	2,525,565 (91.7%)	1,187,592 (90.67%)	96,369 (89.85%)
Bot (#Bots)	Neris(10)	Rbot(10)	Rbot(3)

Table 1. Data description for each dataset (Left)

Feature
Date
Flow start time
Duration
Protocol
Src IP addr:Port
Dst IP addr:Port
Flags
Types of services
of Packets
of Bytes
of Flows
Label

Table 2. Features in each dataset (Right)

TEST & RESULTS

Data Preparation

Among 12 features, *Date and time, IP address and port number* and *the number of flows* were excluded because Date was all the same, and there were very specific IP addresses that were set to Bots. Each test was done by splitting the dataset randomly into training and test set in ratio of 8:2. The following is the averaged values of 5 runs for each algorithm on a single machine with 64GB of memory.

Results

Accuracy scores of each algorithms and ensemble method

Although F1 score generally shows higher scores than MCC, MCC is more reliable because the true negatives are not considered in F1 score.

Individual algorithms	• ***DT >= NN >GNB in accuracy*** • NN > DT > GNB in time consumption
Voting	• Better than GNB and NN, but worse than DT • This is because combining all three algorithms works by averaging the predictions
Boosting	• Adaboosting does not help either GNB or DT. • Boosting works combining multiple 'weak' classifiers. DT is already strong enough. • It takes much more time than sole algorithm and lowers the scores for GNB. • Taking Adaboosting to make the detection better is not worth considering.
Bagging	• Bagging each algorithm seems very similar to using a single classifier. • Bagging DT does not provide considerably better results for each algorithm alone. • Bagging works by sampling training data to literately train the model. • But data itself is very sparse.

Table 2. Interpretation of the results

CONCLUSION

Findings

- Decision tree without any ensemble method would the most preferable.
- Taking ensemble methods in a hope of enhancing the accuracy of machine learning algorithms for Botnet detection is not likely to be a help to the actual detection.
- When a real-time detection system is considered, taking GNB or DT without any ensemble methods could be a good option.

Future works

- **Data dependency:** The accuracy results are much lower when it comes to the huge dataset. Possible reason can be overfitting or the very nature of the different bots (Neris & Rbot). Also, Boosting GNB shows near zero of MCC which means no better than random prediction. The reason for this is also to be studied.
- **Other algorithms / dataset can be evaluated:** SVM, k-NN or Random forest algorithms can be tested to see if they are appropriate for botnet detection. Also, more network attribute can be considered by making full use of NetFlow features.
- **Scalability:** Spark MLlib also provides classification algorithms. But ensemble methods are not developed yet. Finding out if it is feasible to implement the ensemble methods with Spark MLlib can be considered.