# Final Report

## INSE 6630: Recent Developments in Information System Security

## DeepFool – A Simple and Accurate Method To Fool Deep Neural Networks

### Submitted to

Prof. Jun Yan

UNIVERSITÉ Concordia UNIVERSITY

GINA CODY
SCHOOL OF ENGINEERING
AND COMPUTER SCIENCE

### Group Members

| | | |
|---|---|---|
| Yunfeng Lu | Ho-Yang Chang | Yunus Emre Aydar |
| 40118242 | 40130521 | 40110411 |

### Submitted by

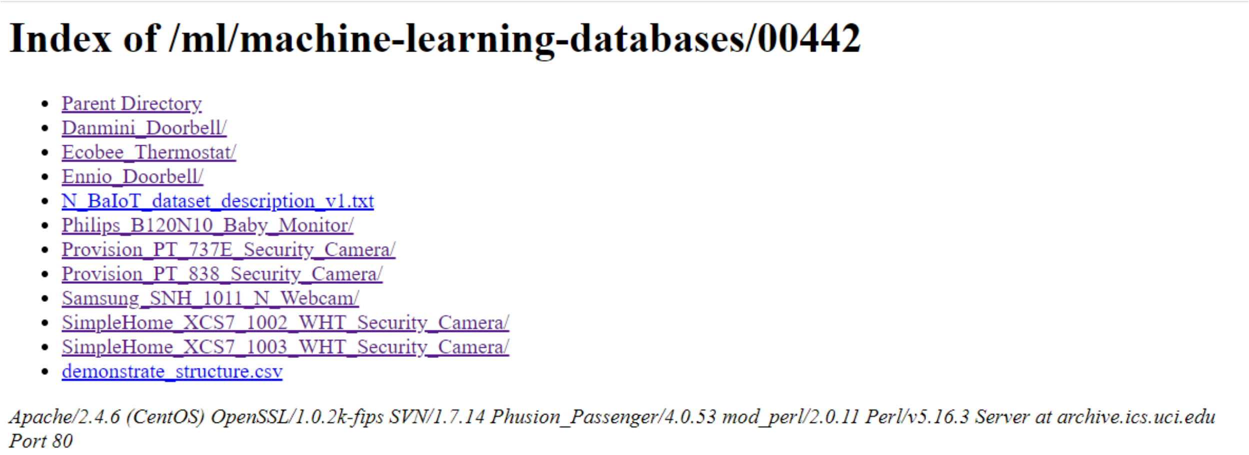
Yunus Emre Aydar

40110411

### December 2020

## Implementation Paper 1

First of all, we chose the paper of A Supervised Intrusion Detection System for Smart Home IoT Devices [3] for implementation project as group. We shared parts during working on the implementation project. My part was focusing on machine learning algorithms such as K-nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, and Naïve Bayes. Also, I wanted focusing on autoencoding deep learning. I perused the papers which are ”Machine Learning DDoS Detection for Consumer Internet of Things Devices” [1] and “N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders”[2] because I wanted to have more knowledge about IoT devices. Then, I decided to work on the dataset of the paper [2] and which are mentioned in the paper through the link [6]. I observed the all datasets and tried finding way to analyze it. According to data set, there were 9 devices with 3 network traffics. 9 devices are Danmini Doorbell, Ecobee Thermostat, Ennio Doorbell, Philips B120N10 Baby Monitor, Provision PT 737E Security Camera, Provision PT 838 Security Camera, Samsung SNH 1011 N Webcam, SimpleHome XCS7 1002 WHT Security Camera, and SimpleHome XCS7 1003 WHT Security Camera. The categories are Benign Traffic, Gafgyt Traffic, and Mirai Traffic. The dataset [6] can be seen below:



**Index of /ml/machine-learning-databases/00442**

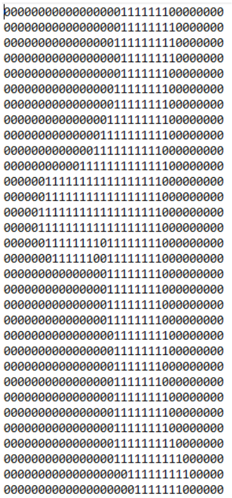
- Parent Directory
- Danmini_Doorbell/
- Ecobee_Thermostat/
- Ennio_Doorbell/
- N_BaIoT_dataset_description_v1.txt
- Philips_B120N10_Baby_Monitor/
- Provision_PT_737E_Security_Camera/
- Provision_PT_838_Security_Camera/
- Samsung_SNH_1011_N_Webcam/
- SimpleHome_XCS7_1002_WHT_Security_Camera/
- SimpleHome_XCS7_1003_WHT_Security_Camera/
- demonstrate_structure.csv

*Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips SVN/1.7.14 Phusion_Passenger/4.0.53 mod_perl/2.0.11 Perl/v5.16.3 Server at archive.ics.uci.edu Port 80*

According to each network traffics are categorized with 115 independent features separately. I tried gather 2 of the networks traffics which are Benign and Gafgyt in one csv file (benignandgafgyt_traffic(DanminiDoorbell)csv). After that I put another feature as outcome for Benign and Gafgyt  in the csv file that I was able to use K-nearest Neighbors, Support Vector Machine, Decision Tree, Random Forest, and Naïve Bayes to operate ML algorithms (INSE6630DecisionTree, INSE6630K-Nearest, INSE6630NaiveBayesApproach, INSE6630RandomForest, and INSE6630SVM) in Python. Some algorithms have not good results like Naïve Bayes and Support Vector Machine. Then, I worked on autoencoding algorithm with other team members for N-BaloT dataset [6]. Our algorithm that we created did not give us satisfying result. Also, we moved to other autoencoder sample like KDD99 dataset [7] to understand different approaches. I implemented KDD99 code and dataset via the website [7]. After working on our autoencoding algorithm, we decided to change the paper for implementation as group. Because, we did not get satisfying outputs that we wanted as general. All implementations can be seen in IoT Paper Implementation file that I attached.

## Implementation Paper 2

We chose DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks [4] as our implementation paper that the paper [4] became our final implementation project as group. I wanted to focus on Handwritten Recognition to approach DeepFool. I worked on Recognition of Handwritten Digits for SVM and KNN algorithms like I used them in previous project to bring stimulating idea while implementation of the paper [4]. I made use of Machine Learning in Action [5] book to understand how I can use those algorithms for recognition and DeepFool. Also, I got help from my group members about parts that I could not understand in some algorithms during implementation. For the dataset [8], to make sure the performance and effectiveness, I only preserved two numbers '1' and '9' as the recognition objects. Since I only defined the classification label of '9' with -1 and '1' with +1 to demonstrate how SVM handwriting recognition works. Thus, the K-nearest algorithm has been tested by sharing the same dataset. Then, the results of both algorithms were at the same level and made the comparison meaningful. The raw data was written in the text file. And for each text file contains a transformed 32pi*32pi image. I derived the dataset from the Machine Learning Repository of UCI [8] . The figure below is shown as a sample of '1'.



```
00000000000000000111111100000000
00000000000000000111111110000000
00000000000000001111111110000000
00000000000000000111111110000000
00000000000000000111111100000000
00000000000000001111111100000000
00000000000000001111111100000000
00000000000000011111111100000000
00000000000000111111111100000000
00000000000001111111111000000000
00000000000111111111111100000000
00000011111111111111111000000000
00000011111111111111111000000000
00000111111111111111111000000000
00000111111111111111111000000000
00000011111111011111111000000000
00000001111110011111111000000000
00000000000000011111111000000000
00000000000000011111111000000000
00000000000000011111111000000000
00000000000000011111111000000000
00000000000000001111111100000000
00000000000000001111111100000000
00000000000000001111111000000000
00000000000000001111111000000000
00000000000000001111111100000000
00000000000000001111111100000000
00000000000000001111111100000000
00000000000000001111111110000000
00000000000000001111111111000000
00000000000000000011111111000000
00000000000000000001111111000000
```

According to my experiment between KNN and SVM, KNN was more efficient than SVM in terms of computational time, accuracy and training difficulty. SVM needed more time to fit each classification result to determine the best threshold for the algorithm in order to get the best performance. Furthermore, linear SVM also provided a relatively good result and it should not be ignored by the experimenters and more kernels are also needed to be tested to determine the best performance parameters. I tried changing the images of digits, but the output does not reveal like what I have imagined. Overall, I had closer knowledge about how DeepFool can be done. I shared my ideas to my group. After that I helped my group to implement the paper [4] of DeepFool.  In the paper [4], the code of DeepFool is mentioned that we use the link [9] to retrieve dataset and we tried to implement it in Python environment. After that we made use of the paper [10] for dataset and implementation as well. Firstly, we imported a pretrained deep neural network ResNet34. Then, we inputted our image and normalized it. After that, we used ResNet34 model and input image DeepFool into algorithm which returns with perturbation image from original label . Finally, the algorithm in the code was working successfully that a pretrained ResNet34 deep neural network model with misclassified labels. We acknowledged as group that DeepFool algorithm  is efficient and accurate to compute the minimum perturbation. Also, It was useful to change the label . Working demo of the code can be seen in our group presentation. At the end of implementation, we understood as group that DeepFool is very effective method as tool that can actuates the robustness of classifier. All implementations can be seen in Deep Fool Paper Implementation file that I attached. Overall, we achieved satisfying result as group and everyone in the group contributed fairly throughout all project.

## References

[1] R. Doshi, N. Apthorpe and N. Feamster, "Machine Learning DDoS Detection for Consumer Internet of Things Devices," in *2018 IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, USA, 24-24 May 2018.

[2] Y. Meidan et al., "N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders," *IEEE Pervasive Computing,* vol. vol. 17, no. no. 3, pp. pp. 12-22, Jul.-Sep. 2018.

[3] E. Anthi, L. Williams, M. Słowińska, G. Theodorakopoulos and P. Burnap, "A Supervised Intrusion Detection System for Smart Home IoT Devices," *IEEE Internet of Things Journal,* vol. vol. 6, no. no. 5, pp. pp. 9042-9053, Oct. 2019.

[4] S. Moosavi-Dezfooli, A. Fawzi and P. Frossard, "DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016.

[5] P. Harrington, Machine Learning in Action, Shelter Island, NY: Manning Publications Co., April 2012.

[6] Y. Meidan at al., "UCI Machine Learning Repository," 19 March 2018. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT.

[7] Competition, The Third International Knowledge Discovery and Data Mining Tools, "UCI Machine Learning Repository," 1 January 1999. [Online]. Available: http://archive.ics.uci.edu/ml/datasets/KDD+Cup+1999+Data.

[8] E. Alpaydin and C. Kaynak, "UCI Machine Learning Repository," 1 July 1998. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/optical+recognition+of+handwritten+digits.

[9] S. Moosavi-Dezfooli, A. Fawzi and P. Frossard, "LTS4/DeepFool," 24 August 2017. [Online]. Available: https://github.com/LTS4/DeepFool/tree/master/Python.

[10] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, 2016.