Ira A. Fulton Schools of Engineering
Arizona State University

CSE 548: Advanced Computer Network Security

Machine Learning-Based Anomaly Detection Solutions

Purpose

This project offers hands-on experience in using machine learning to spot abnormal patterns in network traffic, indicating potential threats like intrusions or malware. It introduces learners to proactive security measures, highlighting the effectiveness of anomaly detection. Through this project, learners explore machine learning and neural networks, understand data preprocessing, and gain practical skills in building effective anomaly detection systems for cyber security.

Objectives

Learners will be able to:

- Use Python to perform data pre-processing for ML.
- Implement Python programs to achieve desired ML functions/features.
- Describe the basic concept of neural networks for machine learning
- Use feedforward neural networks (FNN) as an example to illustrate the procedure for establishing an anomaly detection solution
- Use FNN to build a basic anomaly detection model for a given network traffic data set
- Analyze and evaluate ML outcomes to address anomaly detection questions.

Technology Requirements

- Hardware
  - Intel or AMD based computer with 8GB or more memory.
  - NOTE: 6GB is technically sufficient, but performance will be sluggish.

- Software
  - VirtualBox 5.0 or Newer
  - Windows 10, Mac OS X, or Linux 64-bit as the base operating system.
  - Access to Apporto through a web browser
  - Linux: Ubuntu 18.04 LTS (Bionic Beaver)

Description

In this project, you will use NSL-KDD dataset and Feed-forward Neural Network (FNN) for network anomaly detection and attack analysis. You will create customized training and testing datasets for several different data analysis scenarios based on the supplied NSL-KDD dataset. Based on the training and testing results of each created scenario, you will need to address a few questions regarding ML-based anomaly detection accuracy and efficiency.

The assessment of the lab is based on the completeness and depth of addressing the provided questions. Additionally, learners need to demonstrate how they fulfilled the ML tasks and produce sound ML results to support their arguments to address supplied questions. Learners need to submit a sequence of screenshots and corresponding illustrations to demonstrate how they fulfilled the required ML setup, training, and validation.

Learners need to submit updated Python programming codes and lab reports to provide in-depth illustrations based on their testing.

In summary, learners will:

- Use Python to perform data pre-processing for ML.
- Implement Python programs to achieve desired ML functions/features.
- Analyze and evaluate ML outcomes to address anomaly detection questions.

Directions

You may work on your project locally or through Apporto. Follow the access guidelines provided and then review the Lab Preparation steps to complete your work. Headings or titles labeled with “Local only” are specific to the local setup, otherwise you may complete your work through Apporto as well. Please make sure your final deliverable follows the requirements.

Project Files

The project files include the VM image file, the assigned project, and associated background labs. These can be found in your course on the Project Overview page.

Local Set-up

Setup your local environment based on the directions provided below:

1. Download VirtualBox 5.0 or newer. **Note**: This step is not required if you have installed the VirtualBox in the previous projects.
2. Download the Virtual Machine Image File mentioned in the Project Files section
3. Extract the .vdi file from the downloaded zip file. Note that the image sizes are large and sometimes you may not be able to decompress to get your image file. In such case follow the instructions below:
    a. Go to your terminal and do: `unzip file_name -d /path/to/directory`
4. After extracting the image files, follow the instructions in the background lab **CS-SYS-00101 (VM in VirtualBox)** available in the **Associated Background Labs** zip file to set up the Virtual Machine.
5. Server credentials:
    a. Username: ubuntu
    b. Password: 123456

Accessing Apporto

For this project, you will work through Apporto's modular cyberlabs available through the course. Review all the project directions before starting so you know how to submit your work correctly:

1. In the course, click "Apporto Virtual Lab – App Store" (be patient while the lab loads)
2. In the App Store, select the "**CSE 548 Modular CyberLab**" lab. Please be patient as the lab initializes, which may take a few moments.
3. Once the lab environment is ready, you will be on a Linux machine and you need to click on the "**GNS3**" app from the Activities Menu.
4. In the pop-up window, choose "Open a project from disk". Navigate to the path Home → Labs → projects → CSE548 Project 4 and then select the "**CSE548 Project 4.gns3**" file.
5. You will see two (2) components listed: a Linux Server(Project4) and a NAT Network(NAT1) with an IP range of 10.0.2.0/24.
6. Right-click on the Linux server and select "**Start"**, to start the server. You will know it's running when the red block next to the server changes to green.

7. Again, right-click on the server and select "**console**" to access the Linux desktop.
8. Server credentials:
    a. Username: ubuntu
    b. Password: 123456
9. You are all set! You can now start working on your project.

***Note**: Learners should refrain from using the virtual machine (VM) for personal purposes. This VM is provided solely for academic and course-related activities. Any personal activities may interfere with its intended purpose and impact the learning environment for yourself and your peers.*

Project Preparation:

Assigned Project Lab and Associated Background Labs

- Unzip the zip file for the labs using: `unzip file_name -d /path/to/directory`

Background Labs

- CS-ML-00001 (Machine Learning Environment Setup);
- CS-ML-00101 (NSL-KDD dataset);
- CS-ML-00199 (Python Machine Learning basic Concepts);
- CS-ML-00200 (Python-based data preprocessing);
- CS-ML-00201 (Feed-Forward Neural Network (FNN))

Task 1 System Setup

In this Task, you need to check if the Machine Learning (ML) environment is set up properly. Following the following instructions on suggested background labs to check the ML running environment.

Suggestions:

1. Review and exercise the lab CS-ML-00001 (Machine Learning Environment Setup)
2. Review the lab CS-ML-00101 (Understand NSL-KDD dataset),
3. Review the lab CS-ML-00199 (Python Machine Learning Basic Concepts)
4. Review and practice CS-ML-00200 (Python-based data preprocessing)
5. Review and practice CS-ML-201 (FNN).

Note that the ML service (anaconda) may have already been set up on your server. Follow the background lab CS-ML-00001 to verify if these servers are set up properly to conduct your required

tasks.

To conduct the following tasks, you need to download the data source and Python programs. You can download lab resource files by using the following command. (Check if wget is installed using the command ‘wget –version’. If wget is not installed, install it using ‘sudo apt install wget’)

- **Note**: Please download the code from the repo below. DO NOT use the code that’s already stored in your VM.

```
$ wget https://github.com/SaburH/CSE548/raw/main/lab-cs-ml-00301.zip
$ unzip lab-cs-ml-00301.zip
$ cd lab-cs-ml-00301
```

Source code files are located in the folder ‘lab-cs-ml-00301’. Their content and features are summarized as follows:

- There is a folder NSL-KDD that contains the data that can be processed and analyzed in this lab. Please refer to Lab CS-ML-00101 (understanding NSL-KDD dataset) for more information about the dataset.
- The file “Spyder ReadMe.pdf" is a brief illustration of how to use the Python programming GUI app Spyder.
- The file fnn_sample.py is the Python code that provides data pre-processing, data training, and the presentation of data analysis results. You will use this file as the starting point to develop new Python codes to implement the required features described in the rest of the tasks.
- The file distinctLabelExtractor.py loads an NSL-KDD file and extracts attack names and types.
- The file DataExtractor.py creates customizable training and testing datasets by extracting a subset of attack classes from the training and testing datasets.
- The file categoryMapper.py creates labeled category data for string-based features.
- The file data_preprocesor.py creates a Python library that can ease the data pre-processing procedure of Python-based ML data processing solutions.

Please refer to lab CS-ML-00201 for more information about how to use fnn_sample.py, and refer to lab CS-ML-00200 for more information about how to use the following Python programs: *distinctLabelExtractor.py, DataExtractor.py, categoryMapper.py, and data_preprocesor.py.*

Task 2 Create Data Modules for Anomaly Detection

In this project, we consider NSL-KDD dataset D = (A, N) that contains attack sub-dataset A and normal sub-dataset N. In this project, we choose D = KDDT rain + .txt in the NSL-KDD dataset for training and D = KDDT est+.txt in the NSL-KDD dataset for testing. Now, you picked 4 attack types, e.g., A1 −A4 ⊂ A along with normal data N, you start with a subset A0 ⊂ A for training the model along with all the normal data N. The four classes of attacks include: A1: Denial of Service (DoS), A2: Probe, A3:User to Root(U2R), and A4: Remote to Local (R2L). Refer to lab CS-ML-00101 for more details of these four classes of attacks their their included sub-classes.

In this task, you need to use DataExtractor.py to create a customized training dataset. Refer to lab CS-ML-00200 for more details on using this Python program to create customized training and testing datasets.

In Table CS-ML-00301.1, three attack scenarios will be chosen. You need to create three sets of data including training and testing for scenarios SA, SB, and SC.

Table CS-ML-00301.1

Customized Attack Scenarios Setup

Datasets	Scenario A (SA)	Scenario B (SB)	Scenario C (SC)
Training Dataset	A1, A3, N	A1, A2, N	A1, A2, N
Testing Dataset	A2, A4, N	A1, N	A1, A2, A3, N

Task 3 Anomaly detection analysis

In this task, you will need to modify the provided fnn_sample.py file to take new inputs training, and testing datasets generated from Task 2. Note that the supplied fnn_sample.py takes one training dataset and splits it into training and testing portions. However, in this task, you need to modify it to take separate training and testing datasets generated from Task 2. You will need to run three different training and validation according to three scenarios: SA, SB, and SC. You should make the training and testing parameter setup for FNN the same, i.e., the batch size, number of epochs, neural network setup including the number of first-layer nodes, the number of hidden layer nodes, applied loss functions, threshold, etc. Based on the newly created training datasets and testing datasets, you should address the following questions. Please present your reasoning and analysis results.

1. Which scenario produces the most accurate testing results? Observe your model's prediction capability with respect to the change in the attack classes on which it was trained. Then,

observe the accuracy of the prediction when trained on these subsets of attack types. Observe if the model predicts better when it is trained on a specific subset of attack classes.

2. What is the average accuracy that it detects the new class of attacks (in the testing dataset in SA and SC) to be as normal or attack? (Hint: With the model set to perform binary classification, the predicted values of the model can be:
   - 0 → Normal,
   - 1 → Attack.

   This prediction is associated with the accuracy of the prediction. Note down the accuracy of the prediction, the prediction is normal or attack, for the above unknown attacks A2 and A4 in SA or A3 in SC).

3. What is the difference between attacks in an untrained subset and attacks from the trained dataset? (Hint: For example in SA, how are A1 and A3 different from A2 and A4? Do A1 and A2 belong to the same attack category or have similar differences with respect to the normal data? Present your observations can be made by looking at the data for these attacks).

4. Does the prediction accuracy relate to the attacks being similar? If so, what is the similarity? (Hint: If the prediction accuracy was found better, look at the data and the attack types for any similarities that would have made the prediction better).

Formatting Specifications

Use the Lab Report template to submit your sequence of screenshots and lab running procedures. Use the file name: [Your Name_CSE 548_Name of Project or Assignment]

Create a zip file with your updated code file with comments that illustrate your code and submit it along with your project report.

Submission Directions for Project Deliverables

For this project, you will document lab running procedures by taking a sequence of screenshots and explain ML training results. You will also need to submit a report to explain what they can accomplish based on the description presented in the Evaluation section below.

You are given an unlimited number of attempts to submit your best work. The number of attempts is given to anticipate any submission errors you may have in regards to properly submitting your best work within the deadline (e.g., accidentally submitting the wrong paper). It is not meant for you to receive multiple rounds of feedback and then one (1) final submission. Only your most recent submission will be assessed.

You must submit your Machine Learning-Based Anomaly Detection Solutions Project deliverable(s) in this submission space. Learners may not email or use other means to submit any assignment or project for review, including feedback, and grading.

The Machine Learning-Based Anomaly Detection Solutions Project includes one (2) deliverables:

- **Project Report:** Use the Lab Report template to submit your sequence of screenshots and lab running procedures. Use the file name: [Your Name_CSE 548_Name of Project or Assignment]
- **fnn_simple.py:** Create a zip file with your updated code file with comments that illustrate your code and submit it along with your project report.

Report Submission

1. Navigate to your course and click on the "**Submission: Machine Learning-Based Anomaly Detection Solutions**" submission space
2. Click "**Start Assignment**" in the upper right corner
3. Click "**Upload File**" and add your completed project report template.
    a. To add another file, select, "**+ Add Another File**" and add additional files
4. If needed: to resubmit the assignment
    a. Click "**New Attempt**" and add your file submission again
    b. When finished, click "**Submit Assignment**"

Evaluation

In this lab, learners are required to use a basic FNN model to create an anomaly detection model for network intrusion detection.

1. Using the lab screenshot feature to document ML running results only. Provide sufficient but concise explanations of the generated results for each given training/testing scenario.
2. Submit the updated fnn_simple.py and provide sufficient comments to illustrate your updated codes.
3. Submit the report to provide an in-depth analysis to address the questions given in Task 3.