Hands-on Lab Description

CS-ML-00301 – Use Feed-Forward Neural Network (FNN) for Network Traffic Anomaly Detection

Category:

```
CS-ML: Machine Learning
```

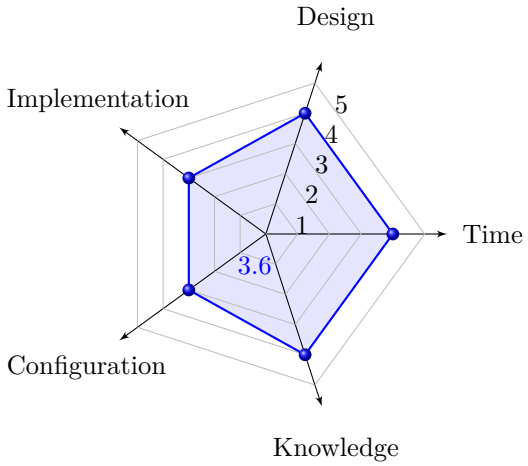
Objectives:

```
1 Learn how to setup FNN training model for network anomaly detection (using NSL-KDD
      dataset)
2 Understanding the prediction of FeedForward Neural Network model in detecting network
       attacks
```

Estimated Lab Duration:

```
1 Expert: 120 minutes
2 Novice: 600 minutes
```

Difficulty Diagram:



Difficulty Table.

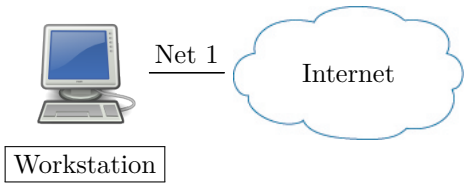
Measurements	Values (0-5)
Time	4
Design	4
Implementation	3
Configuration	3
Knowledge	4
Score (Average)	3.6

Required OS:

```
Linux: Ubuntu 18.04 LTS
```

Lab Running Environment:

```
ThoTh Lab: https://thothlab.org
```




```
1 Server: Linux (Ubuntu 18.04 LTS)
2 Network Setup: connected to the Internet
```

Lab Preparations:

```
Python and Anaconda software packages installed on the Linux VM. Reference Lab:
    CS-ML-00001
NSL-KDD dataset. Reference Lab: CS-ML-00101
Python Machine Learning basic concepts: CS-ML-00199
Python-based data pre-processing: CS-ML-00200
FNN ML model. Reference Lab: CS-ML-00201
```

Lab Overview

In this lab, 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 address the provided questions. Additionally, students need to demonstrate how they fulfilled the ML tasks and produce sound ML results to support their arguments to address supplied questions. Students need to submit a sequence of screenshots and corresponding illustrations to demonstrate how they fulfilled the required ML setup, training, and validation. Students need to submit updated Python programming codes and lab report to provide in-depth illustration based on their testing.

In summary, students will do:

- 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.

Task 1 System Setup

In this Task, you need to check if the Machine Learning (ML) environment is setup 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 setup on your server. Follow the background lab CS-ML-00001 to verify if these servers are setup 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 following command. (Check if wget is installed using 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://gitlab.thothlab.org/thoth-group/ThoThLabResource/raw/master/
       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 in follows:

- There is a folder *NSL-KDD* contains the data that can be processed and analyzed in this lab. Please refer to the Lab CS-ML-00101 (understanding NSL-KDD dataset) for more information about the dataset.
- The file "Spyder ReadMe.pdf" is a brief illustration on how to use python programming GUI app Spyder.
- The file *fnn_sample.py* is the python codes that provide 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 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 $D = KDDTrain + .txt$ in the NSL-KDD dataset for training and $D = KDDTest + .txt$ in the NSL-KDD dataset for testing. Now, you picked 4 attack types, e.g., $A_1 - A_4 \subset A$ along with normal data N, you start with a subset $A' \subset A$ for training the model along with all the normal data N. The four classes of attacks include: A_1: Denial of Service (DoS), A_2: Probe, A_3:User to Root(U2R), and A_4: 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 datasets. Refer to lab CS-ML-00200 for more details of 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 S_A, S_B and S_C.

Table CS-ML-00301.1
Customized Attack Scenarios Setup

Datasets	Scenario A (S_A)	Scenario B (S_B)	Scenario C (S_C)
Training Dataset	A_1, A_3, N	A_1, A_2, N	A_1, A_2, N
Testing Dataset	A_2, A_4, N	A_1, N	A_1, A_2, A_3, 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 portion. 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: S_A, S_B, and S_C. 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 presents your reasoning and analysis results.

1. Which scenario produce 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 subset of attack types. Observe if the model predicts better when it was 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 S_A and S_C) to be as normal or attack? (Hint: With the model set to perform binary classification, the predicted values of the model can be:

   - $0 \rightarrow$ Normal,
   - $1 \rightarrow$ 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 A_2 and A_4 in S_A or A_3 in S_C).

3. What is the difference between attacks in untrained subset and attacks from the trained dataset? (Hint: For example in S_A, how are A_1 and A_3 different from A_2 and A_4. Do A_1 and A_2 belong to 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).

Deliverable

Students need to document lab running procedures by taking a sequence of screenshots and explain the ML training results. Students also need to submit a report to explain what they can accomplish based on the description presented in the *Lab Assessment* section. On how to submit screenshots, please refer to the *Guidance for Students* Section B.4 (Submit Lab Reports).

Related Information and Resource

```
NSL-KDD dataset:
https://www.unb.ca/cic/datasets/nsl.html

KDD CUP ARCHIVES
https://www.kdd.org/kdd-cup

Data preprocessing:
https://www.shanelynn.ie/select-pandas-dataframe-rows-and-columns-using-iloc-loc
```

```
-and-ix/
https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine
-learning-3fc273365621
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train
_test_split.html
https://scikit-learn.org/stable/modules/preprocessing.html

Build ANN:
https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/
https://machinelearningmastery.com/rectified-linear-activation-function-for-deep
-learning-neural-networks/
https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/
https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep
-learning-neural-networks/
```

\nobreak

Lab Assessment (100 points)

In this lab, the students is required to use basic FNN model to create an anomaly detection model for network intrusion detection.

1. (35 points) Using the lab screenshot feature to document ML running results only. Provide sufficient but concise explanations on the generated results for each given training/testing scenarios.
2. (20 points) Submit the updated *fnn_simple.py* and provide sufficient comments to illustrate your updated codes.
3. (45 points) Submit the report to provide in-depth analysis to address the questions given in Task 3.