Report

Angelos Kelekoglou

Email : aggeloskel@outlook.com

Academic email: akelekog@csd.auth .gr

Github: https://github.com/akelekog

Project Ttile: Network Anomaly Detection With Assistance of Deep learning.

Project Specifications , notes :

The project is implemented on jupyter notebook and uses the KDDCUP'99 dataset.

The project is submitted pre-compiled but can also be exececuted .

For that the directory of the python notebook and the file containing the dataset should be the same. Specifically drag and drop the Kaggle file on the jupyter notebook home interface .

The dataset is in the form of txt files.

Results interpretation:

The results are simply given in the format of accuracy on the test dataset for each imple mented model . There are on only two classes (attack , normal) `Class distributio n: 1: 67343, 0: 58630`

Where 0 is normal and 1 is attack.

Implementation:

Network security is becoming more important as a result of the massive growth in computer network traffic and the abundance of applications. All computer systems are prone to security flaws, automating the detection of anomalies is an efficient way of approaching the problem. This project is an indication of the ability of neural networks to simplify anomaly detection.

The project has been implement on jupyter noteboook. In total there have been created four models. This data is KDDCUP'99 data set(see https://www.kaggle.com/datasets/anushonkar/network-anamoly-detection for the listing of the features), which is widely used as one of the few publicly available data sets for network-based anomaly detection systems. In particular the models created consist of:

```
1)A deep learning model with
    activation: 'relu',
    batch_size: 32,
    dropout_rate: 0.2,
    epochs: 20,
    layers': [32, 64, 32] and an accuracy on the test set of around
85%


2) A deep learning model with
    activation: 'relu',
    batch_size: 64,
    dropout_rate: 0,
    epochs: 50,
    layers': [32,64,128,64,32] and an accuracy on the test set of ar
ound 86%

3) A naïve bayes model with an accuracy on the test set around 85%
```

4) An LSTM model with 64 neurons and an accuracy on the test set around 85%

The problem is approached starting by reading the dataset and assigning the features dropping all the ones that mostly have zero values. We then use a label encoder for the categorical features and proceed to created the training set. The problem is approached with a binary classification with a class distribution of :

```
Class distribution: 1: 67343, 0: 58630
```

Where 0 is normal and 1 is attack.

The data is scaled and we have a training set of 125973 samples with 38 features.

Afterwards we use PCA to reduce the dimensions of the data while keeping as much information as possible.In particular, we see that the first 20 components capture about 92% of the variance of the original data so we select the first 200 components and reapply PCA with this number of selected components resulting in x_train_reduced and x_test_reduced.

The next step is to implement the create_model function according to which the first and second model will be implemented. The following function takes as arguments the lists layers,activation,dropout_rate. The declaration of the model is made and then through a for iteration the appropriate layers are added. If index i is 0 then the initial layer is added with input shape = x_train_reduced.shape[1] i.e. the number of columns of x_train after the PCA effect and its conversion to x_train_reduced as well as the number of neurons nodes from the first element of the respective sublist of the layers list , the Activation layer is also added. Otherwise Dropout layer, Dense layer, Activation layer, Dropout layer and finally the output layer with activation function softmax are added in order. Then the model is compiled using opitmizer adam (with default learning rate = 0.001) , for loss function categorical crossentropy and accuracy metrics. Then the KerasClassifier wrapper is used so that HalvingGridSearchCV can be utilized for 5 fold crossvalidation and search for the best parameters(layers,dropout_rate , activations,batch_size,epochs) , HalvingGridSearchCV was chosen instead of GridSearcCV due to faster execution.Halving grid search is only for the first model.

Note:

The attack ATTACK activity is : normal or DOS or PROBE or R2L or U2R
but since the problem is approached with binary classification the values become normal and attack.

Assets used:

https://www.kaggle.com/datasets/anushonkar/network-anamoly-detection