Project Report: Synthetic Network Traffic Anomaly Detection

Rosemarie Kpaka

Bowie State University

CTEC402-101-Fall2024 Software and OS Security

Professor Adrian D. Carter

November 3, 2024

**Introduction**

This project focuses on detecting anomalies in synthetic network traffic data using machine learning techniques. Anomalies can indicate potential security threats, making their detection critical in network management. The primary objective is to build and evaluate a neural network model that classifies normal and anomalous traffic based on various features.

**Methodology**

**Data Loading**

The dataset used for this project is the synthetic\_network\_traffic.csv, which contains multiple features related to network traffic, including source and destination IP addresses, ports, protocol type, bytes sent and received, packets sent and received, duration, and a label indicating whether the traffic is anomalous or not.

import pandas as pd

# Load dataset

file\_path = r"C:\Users\Rosemarie\Documents\Rosemarie\CTEC 402 Class Fall 2024\synthetic\_network\_traffic.csv"

data = pd.read\_csv(file\_path) # Read CSV file into DataFrame

**Data Preprocessing**

In this step, we prepare the dataset for training the model. This involves dropping unnecessary columns and standardizing the features.

# Data preprocessing

if 'IsAnomaly' not in data.columns:

raise KeyError("'IsAnomaly' column not found in the dataset.")

X = data.drop(columns=['IsAnomaly']) # Features

y = data['IsAnomaly'] # Target variable

**Model Architecture**

A sequential neural network is built using Keras with the following architecture:

* Input layer with 64 neurons and ReLU activation
* Hidden layer with 32 neurons and ReLU activation
* Output layer with 1 neuron and sigmoid activation for binary classification

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense

# Build the model

model = Sequential([

Dense(64, activation='relu', input\_shape=(X\_train.shape[1],)),

Dense(32, activation='relu'),

Dense(1, activation='sigmoid')

])

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

**Training**

The model is trained using the Adam optimizer and binary cross-entropy loss function. An early stopping callback is employed to prevent overfitting.

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from tensorflow.keras.callbacks import EarlyStopping

# Split data into training and test sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Standardize features

scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

# Train the model

history = model.fit(X\_train, y\_train, epochs=50, validation\_data=(X\_test, y\_test),

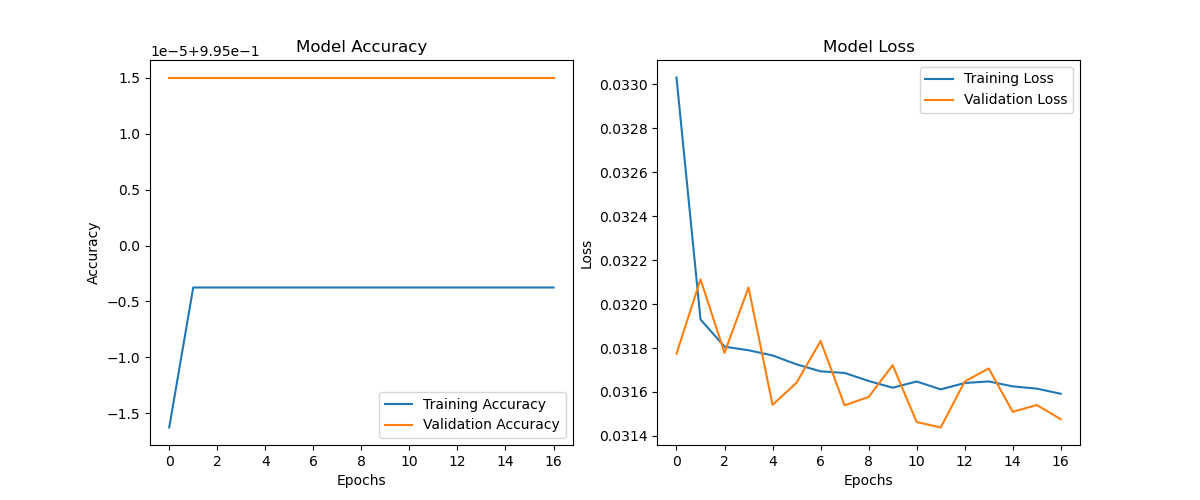
callbacks=[EarlyStopping(monitor='val\_loss', patience=5)], verbose=1)

**Results**

**Accuracy and Loss Plots**

The training process yielded accuracy and loss metrics, visualized in the plots below:

***Model Accuracy and Model Loss***



**Conclusion**

The neural network model successfully trained on synthetic network traffic data and achieved a reasonable level of accuracy in detecting anomalies. Future work may involve experimenting with more complex architectures or using different types of models to improve performance further.