Detection of IoT Botnet Attacks

Group 12:
Utkarsh Meshram (181IT250)
Bhagyashri Bhamare (181IT111)
Chinmayi C. Ramakrishna (181IT113)

Introduction

- A ‘bot’ is a computer program which enables the operator to remotely control the infected system where it is installed.
- A network that is compromised with the attack by such bots is called a botnet.
- It is essential to detect such bots in the network to ensure safety of a system.
- The proliferation of IoT devices which can be more easily compromised than desktop computers has led to an increase in the occurrence of IoT based botnet attacks.
- There is a need to differentiate between hour and millisecond long IoT based attacks.

Abstract

- A **network-based** anomaly detection method for the IoT
- Extracts behavior snapshots of the network
- Uses deep autoencoders to detect anomalous network traffic from compromised IoT devices
- More accurate than the traditional machine learning techniques
- Use of neural network for attack classification
- Creates a complete detection and classification pipeline

Objectives

- ❖ **Heterogeneity Tolerance:** Accommodates growing diversity of IoT devices.
- ❖ **Real world:** Detects abnormal behaviour rather than classification.
- ❖ **Efficiency:** Semi online training of autoencoders is used to improve storage efficiency.
- ❖ Use auto encoders as a complete means of botnet detection.
- ❖ Use real traffic to perform analysis

Novelty

- Autoencoder algorithm is used for all the devices.
- Choice of activation function and optimization algorithm.
- Employs a feature selection mechanism
- A deep feedforward neural network with softmax algorithm for attack type classification.
- Use of LIME to explain the neural network predictions.

Methodology

- **Preparing the data:**
  - Splitting the datasets
  - Feature Scaling
  - Feature selection
- **Anomaly detection:**
  - Deep auto encoding
- **Attack classification:**
  - Deep neural network
- **Evaluation Metrics**
- **Local Interpretable Model-Agnostic Explanations**

Methodology

❖ **Splitting the datasets:**

- Splitting the datasets: train, optimise and test
- The Mirai and BASHLITE datasets will be sampled to the size of normal data.
- Data set split in the ratio 80:20
- For Mirai attack type classification, from each of 5 classes 100000 points will be taken to total of 500000 and then split 80:20 for train and test.

Normal	**Mirai**	**BASHLITE**
555 932	3 668 402	1 032 056

Methodology

- **Feature Scaling:**
  - Standard formula for scaling is used

$$\tilde{X}_i = \frac{X_i - \mu_i}{\sigma_i}$$

$\tilde{x}$: dataset feature i after scaling

X: dataset feature before scaling,

μ: mean of the training set feature

σ is the standard deviation of the training set feature

Methodology

- **Feature Selection:**
  - Fisher's score is used as a metric to measure the importance of a feature

$$F = \frac{\sum_{j=1}^{k} p_j(\mu_j - \mu)^2}{\sum_{j=1}^{k} p_j \sigma_j^2}$$

  - A large Fisher's score means the features produce high inter class variability and small intra class variability.

Methodology

❖ **Deep Autoencoding**

- An autoencoder takes an input and aims to reconstructs the original input.
- Optimization is done through the loss function.

$$\mathcal{L}(X, X') = \frac{1}{n}\sum_{i=1}^{n}(X_i - X'_i)^2$$

- Threshold for error optimization.

$$\tau = \overline{MSE(X_{opt})} + N * STD\left(MSE(X_{opt})\right)$$

Methodology

❖ **Deep Autoencoding**

- Autoencoder will use 5 hidden layers of sizes 0.75, 0.5, 0.25, 0.5, 0.75 of the input feature vector size.
- Hyperbolic tangent is used as activation function.

$$g(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

- A nonlinear function is used to retain the power of nonlinear models.

Methodology

❖ **Attack Classification**

- Deep Neural Network
  - ➔ Two hidden layers each with 8 neurons
  - ➔ Hyperbolic tangent activation function for hidden neurons
  - ➔ Softmax function applied to the last layer

$$softmax(x_j) = \frac{e^{x_j}}{\sum_{i=1}^{N} e^{x_i}}$$

  - ➔ Categorical cross-entropy as a loss function

$$P(y = 1 \mid \boldsymbol{x}) = \max\left\{0, \min\left\{1, \boldsymbol{w}^\top \boldsymbol{h} + b\right\}\right\}$$

Methodology

❖ **Evaluation Metrics**

- For anomaly detection a two class confusion matrix is used.

	Predicted normal	**Predicted attack**
Actual normal	True Negative (TN)	False Positive(FP)
Actual attack	False Negative (FN)	True Positive(TP)

Methodology

❖ **Local Interpretable Model-Agnostic Explanations**

- Explanations in a human interpretable form.
- Provides features that resulted in the prediction by the model.
- It permutes existing data.
- Feature weights from a simple model make explanations for the complex model's local behaviour.

Results

2 class (normal, attack)		3 class (2 botnets +1 normal)		5 classes of Mirai attacks	
Feature	**F Sc**	**Feature**	**F Sc**	**Feature**	**F Sc**
MI_dir_L0.1_weight	3.34	MI_dir_L3_weight	1.96	MI_dir_L0.01_var	43.75
H_L0.1_weight	3.34	H_L3_weight	1.96	H_L0.01_var	43.75
MI_dir_L1_weight	3.18	MI_dir_L5_weight	1.93	MI_dir_L0.1_var	41.43
H_L1_weight	3.18	H_L5_weight	1.93	H_L0.1_var	41.43
MI_dir_L3_weight	3.01	MI_dir_L1_weight	1.87	MI_dir_L0.01_mean	30.05
H_L3_weight	3.01	H_L1_weight	1.87	H_L0.01_mean	30.05
MI_dir_L5_weight	2.86	MI_dir_L0.1_weight	1.70	MI_dir_L0.1_mean	27.03
H_L5_weight	2.86	H_L0.1_weight	1.70	H_L0.1_mean	27.03
MI_dir_L0.01_weight	1.65	MI_dir_L0.01_weight	1.43	MI_dir_L1_var	19.62
H_L0.01_weight	1.65	H_L0.01_weight	1.43	H_L1_variance	19.62

Table 1. Feature Selection Scores

Results

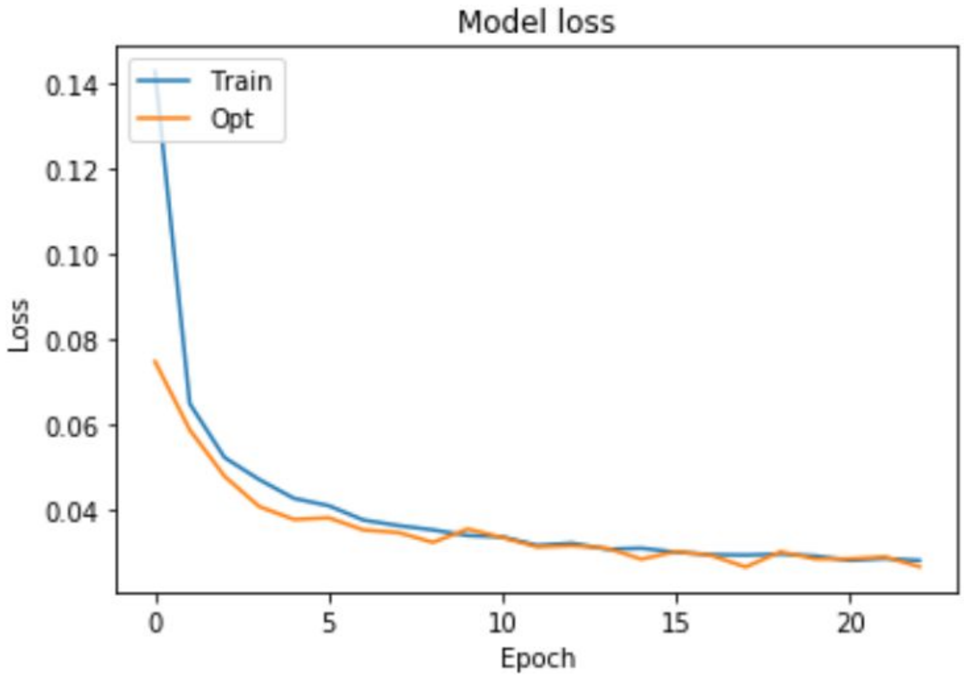


Fig 1. Autoencoder Training Curve

Results

N	Accuracy	False Positives	False Negatives
1	0.9894	3911	23
2	0.9934	2203	25
3	0.9961	1396	28
4	0.9973	978	35
5	0.9979	744	38
6	0.9983	587	41
7	0.9986	467	43
8	0.9988	387	45
9	0.9990	292	51
10	0.992	245	52

Table 2. Results with different N values for threshold

Results

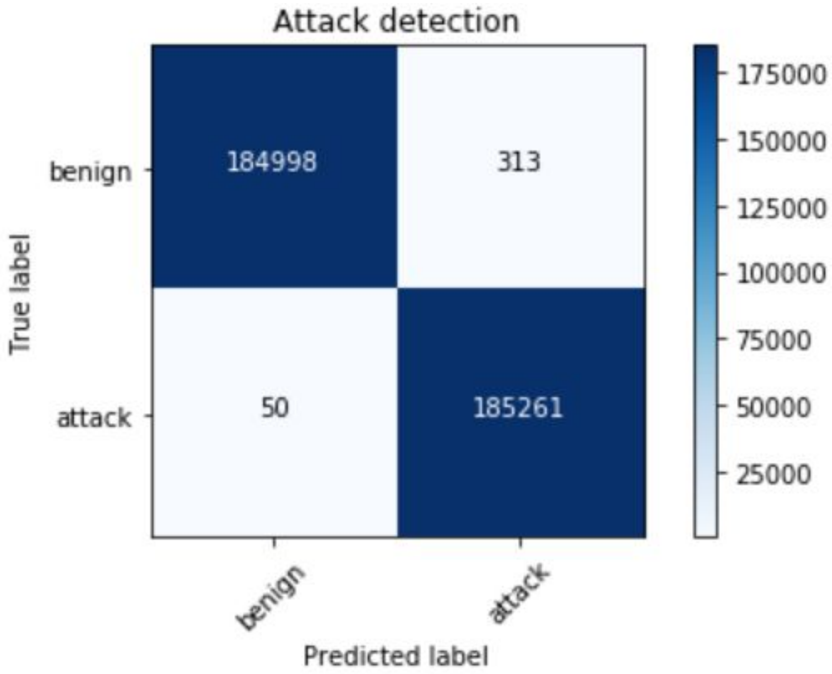


Fig 2. Attack detection on test set.

Results

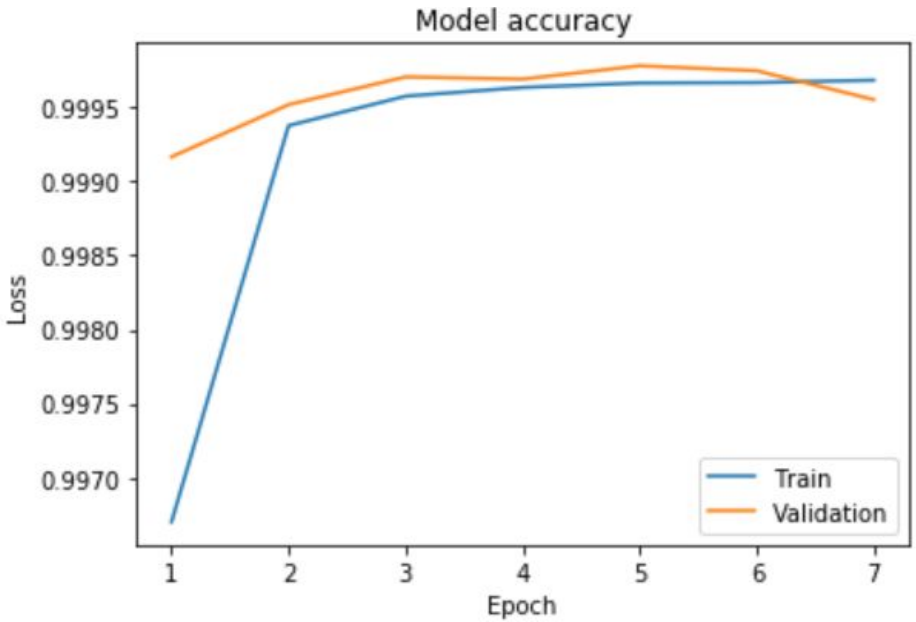


Fig 3. Botnet classification learning curve

Results

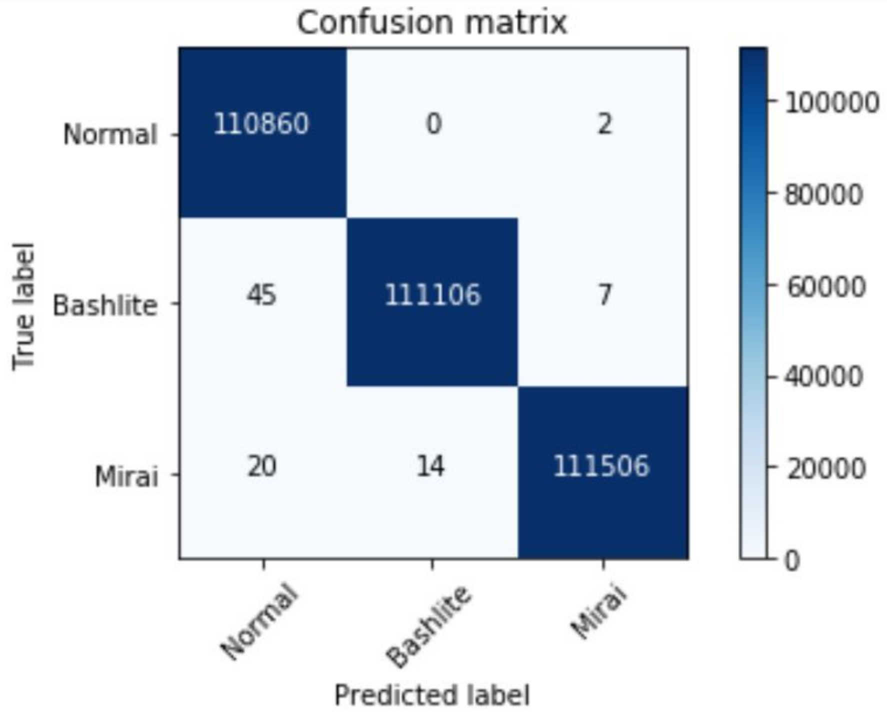


Fig 4. Botnet classification confusion matrix

Results

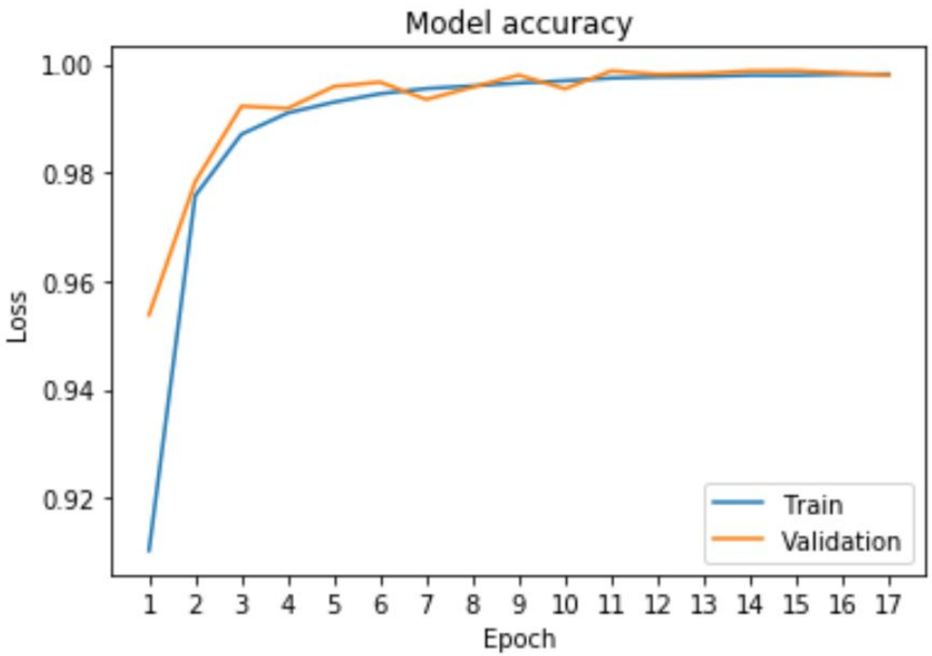


Fig 5. Mirai attack classification learning curve

Results

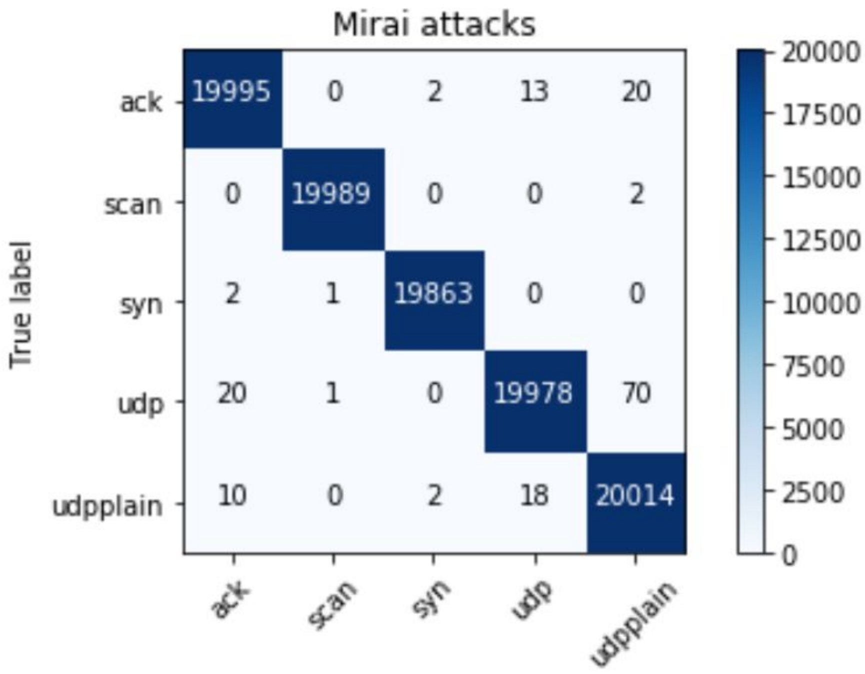


Fig 6. Mirai attack classification confusion matrix

Results

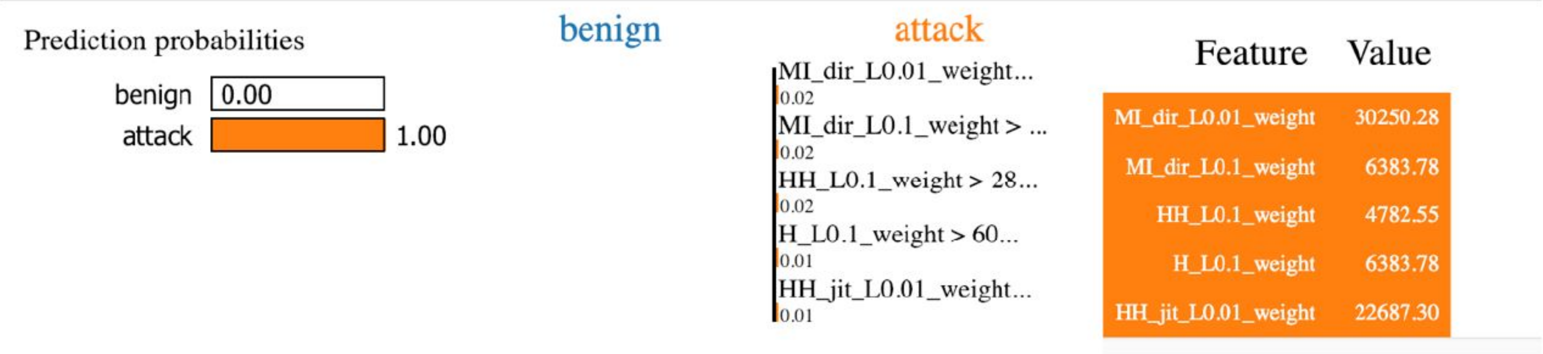


Fig 7. LIME explanation for attack detection

Results

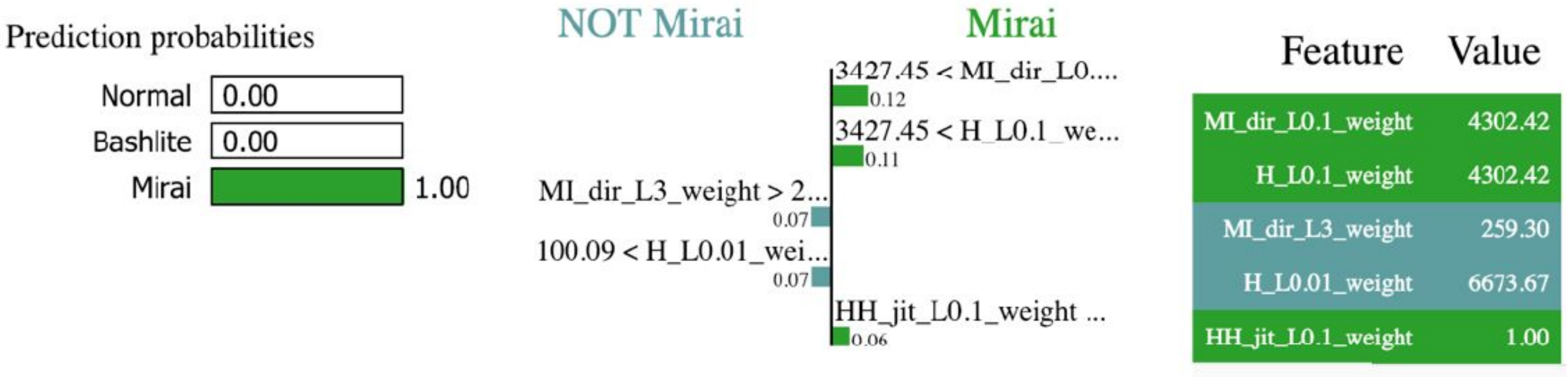
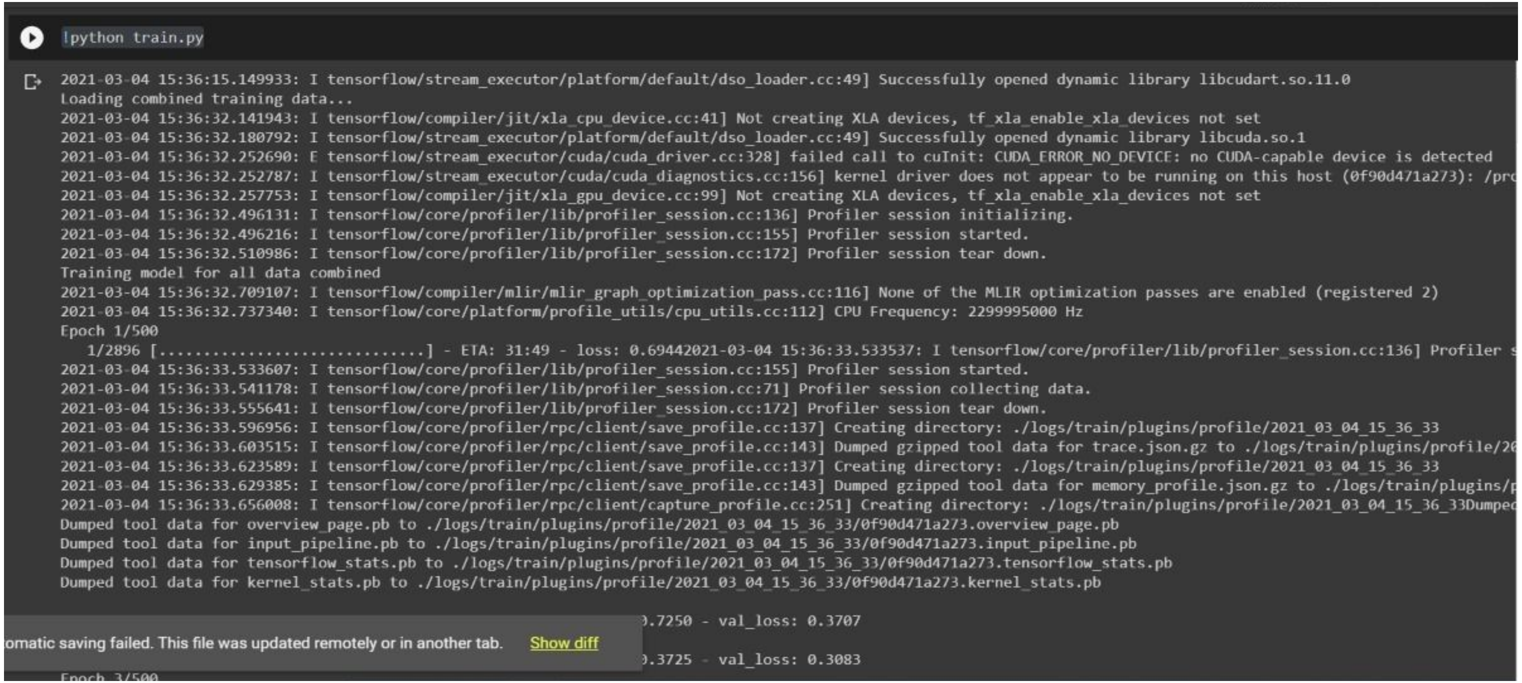


Fig 8. LIME explanation for botnet classification

Results

Fig 9. LIME explanation for Mirai attack type UDP

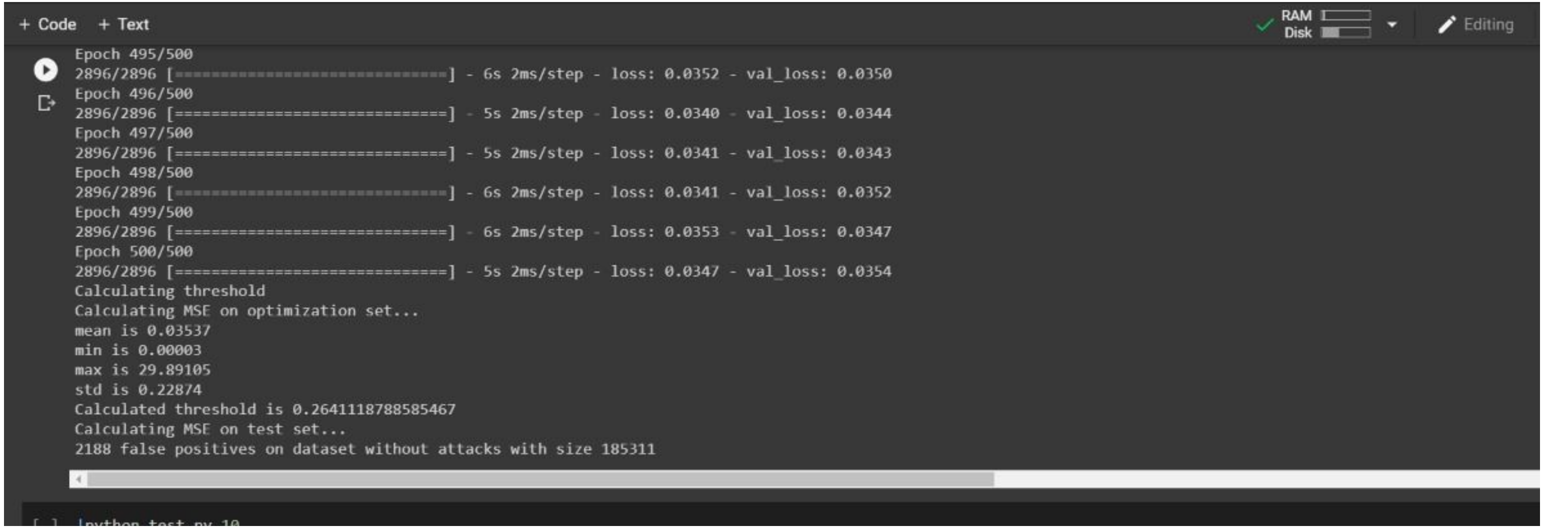
Screenshots


```
!python train.py

2021-03-04 15:36:15.149933: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.11.0
Loading combined training data...
2021-03-04 15:36:32.141943: I tensorflow/compiler/jit/xla_cpu_device.cc:41] Not creating XLA devices, tf_xla_enable_xla_devices not set
2021-03-04 15:36:32.180792: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcuda.so.1
2021-03-04 15:36:32.252690: E tensorflow/stream_executor/cuda/cuda_driver.cc:328] failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected
2021-03-04 15:36:32.252787: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (0f90d471a273): /pro
2021-03-04 15:36:32.257753: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_devices not set
2021-03-04 15:36:32.496131: I tensorflow/core/profiler/lib/profiler_session.cc:136] Profiler session initializing.
2021-03-04 15:36:32.496216: I tensorflow/core/profiler/lib/profiler_session.cc:155] Profiler session started.
2021-03-04 15:36:32.510986: I tensorflow/core/profiler/lib/profiler_session.cc:172] Profiler session tear down.
Training model for all data combined
2021-03-04 15:36:32.709107: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are enabled (registered 2)
2021-03-04 15:36:32.737340: I tensorflow/core/platform/profile_utils/cpu_utils.cc:112] CPU Frequency: 2299995000 Hz
Epoch 1/500
   1/2896 [..............................] - ETA: 31:49 - loss: 0.69442021-03-04 15:36:33.533537: I tensorflow/core/profiler/lib/profiler_session.cc:136] Profiler s
2021-03-04 15:36:33.533607: I tensorflow/core/profiler/lib/profiler_session.cc:155] Profiler session started.
2021-03-04 15:36:33.541178: I tensorflow/core/profiler/lib/profiler_session.cc:71] Profiler session collecting data.
2021-03-04 15:36:33.555641: I tensorflow/core/profiler/lib/profiler_session.cc:172] Profiler session tear down.
2021-03-04 15:36:33.596956: I tensorflow/core/profiler/rpc/client/save_profile.cc:137] Creating directory: ./logs/train/plugins/profile/2021_03_04_15_36_33
2021-03-04 15:36:33.603515: I tensorflow/core/profiler/rpc/client/save_profile.cc:143] Dumped gzipped tool data for trace.json.gz to ./logs/train/plugins/profile/20
2021-03-04 15:36:33.623589: I tensorflow/core/profiler/rpc/client/save_profile.cc:137] Creating directory: ./logs/train/plugins/profile/2021_03_04_15_36_33
2021-03-04 15:36:33.629385: I tensorflow/core/profiler/rpc/client/save_profile.cc:143] Dumped gzipped tool data for memory_profile.json.gz to ./logs/train/plugins/p
2021-03-04 15:36:33.656008: I tensorflow/core/profiler/rpc/client/capture_profile.cc:251] Creating directory: ./logs/train/plugins/profile/2021_03_04_15_36_33Dumped
Dumped tool data for overview_page.pb to ./logs/train/plugins/profile/2021_03_04_15_36_33/0f90d471a273.overview_page.pb
Dumped tool data for input_pipeline.pb to ./logs/train/plugins/profile/2021_03_04_15_36_33/0f90d471a273.input_pipeline.pb
Dumped tool data for tensorflow_stats.pb to ./logs/train/plugins/profile/2021_03_04_15_36_33/0f90d471a273.tensorflow_stats.pb
Dumped tool data for kernel_stats.pb to ./logs/train/plugins/profile/2021_03_04_15_36_33/0f90d471a273.kernel_stats.pb

.7250 - val_loss: 0.3707
omatic saving failed. This file was updated remotely or in another tab.   Show diff
.3725 - val_loss: 0.3083
```

Screenshots




```
Epoch 495/500
2896/2896 [==============================] - 6s 2ms/step - loss: 0.0352 - val_loss: 0.0350
Epoch 496/500
2896/2896 [==============================] - 5s 2ms/step - loss: 0.0340 - val_loss: 0.0344
Epoch 497/500
2896/2896 [==============================] - 5s 2ms/step - loss: 0.0341 - val_loss: 0.0343
Epoch 498/500
2896/2896 [==============================] - 6s 2ms/step - loss: 0.0341 - val_loss: 0.0352
Epoch 499/500
2896/2896 [==============================] - 6s 2ms/step - loss: 0.0353 - val_loss: 0.0347
Epoch 500/500
2896/2896 [==============================] - 5s 2ms/step - loss: 0.0347 - val_loss: 0.0354
Calculating threshold
Calculating MSE on optimization set...
mean is 0.03537
min is 0.00003
max is 29.89105
std is 0.22874
Calculated threshold is 0.2641118788585467
Calculating MSE on test set...
2188 false positives on dataset without attacks with size 185311
```

Screenshots

```
[ ]  !python test.py 10

2021-03-04 18:54:03.638122: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcudart.so.11.0
tcmalloc: large alloc 3374931968 bytes == 0x561ef2a5c000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 1212366848 bytes == 0x561fbbcf2000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 1363894272 bytes == 0x562004926000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 1539252224 bytes == 0x562004926000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 1743896576 bytes == 0x561e3810c000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 1939357696 bytes == 0x561e3810c000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 2224668672 bytes == 0x562004926000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 2516475904 bytes == 0x561e3810c000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
tcmalloc: large alloc 5986140160 bytes == 0x5620a0366000 @  0x7f980f49b1e7 0x7f980d01b46e 0x7f980d06bc7b 0x7f980d06bd18 0x7f980d113010 0x7f980d11373c 0x7f980d11385d
Testing
Loading model
2021-03-04 18:57:47.750878: I tensorflow/compiler/jit/xla_cpu_device.cc:41] Not creating XLA devices, tf_xla_enable_xla_devices not set
2021-03-04 18:57:47.890973: I tensorflow/stream_executor/platform/default/dso_loader.cc:49] Successfully opened dynamic library libcuda.so.1
2021-03-04 18:57:47.978257: E tensorflow/stream_executor/cuda/cuda_driver.cc:328] failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected
2021-03-04 18:57:47.983652: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:156] kernel driver does not appear to be running on this host (0f90d471a273): /proc/
2021-03-04 18:57:48.016396: I tensorflow/compiler/jit/xla_gpu_device.cc:99] Not creating XLA devices, tf_xla_enable_xla_devices not set
Calculated threshold is 0.2641118788585467
2021-03-04 18:57:49.645103: I tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:116] None of the MLIR optimization passes are enabled (registered 2)
2021-03-04 18:57:49.673362: I tensorflow/core/platform/profile_utils/cpu_utils.cc:112] CPU Frequency: 2299995000 Hz
Accuracy
0.8543826324395206
Recall
0.7206641807555947
Precision
0.9837571453827568
[[183106   2205]
```

omatic saving failed. This file was updated remotely or in another tab. Show diff

Screenshots

```
[ ] 0.9837571453827568
[[183106   2205]
 [ 51764 133547]]
explaining with LIME
Explaining for record nr 91960
[('73.59 < MI_dir_L0.01_mean <= 91.75', -0.06922442151196985), ('H_L0.1_mean <= 72.31', -0.05681679758681528), ('73.59 < H_L0.01_mean <= 91.75', -0.05470405659892685)
Actual class
305947    0
Name: malicious, dtype: int64
Explaining for record nr 269261
[('H_L0.1_mean <= 72.31', -0.04496481820668882), ('MI_dir_L0.01_mean <= 73.59', -0.04305258765039592), ('H_L1_mean <= 66.04', -0.035567844275570235), ('H_L0.01_mean <
Actual class
2438167    1
Name: malicious, dtype: int64
Explaining for record nr 186865
[('H_L0.01_weight > 100.18', 0.1057330345551586), ('MI_dir_L0.01_weight > 100.18', 0.08021745998033558), ('72.31 < MI_dir_L0.1_mean <= 86.55', -0.04866644307204839),
Actual class
1322769    1
Name: malicious, dtype: int64
Explaining for record nr 333469
[('H_L0.01_weight <= 28.27', -0.04571651243046312), ('H_L0.01_variance <= 354.13', -0.04551832643607435), ('MI_dir_L0.1_mean <= 72.31', -0.04389183662962892), ('H_L1_
Actual class
1574966    1
Name: malicious, dtype: int64
Explaining for record nr 320699
[('MI_dir_L0.01_mean > 149.58', 0.11489944655419028), ('MI_dir_L0.1_mean > 151.60', 0.11185859966148959), ('H_L0.01_mean > 149.58', 0.11180743961963402), ('H_L0.1_mea
Actual class
3097876    1
Name: malicious, dtype: int64
---------------------------------
```

Conclusion

- The attack detection with deep autoencoder showed outstanding results with accuracy of 0.9991 and false positive rate of 0.0015.
- The anomaly detection with autoencoders can be applied to the data coming from multiple different IoT devices.
- The LIME technique was applied to an attack data point that gave an intuitive interpretation of the features and predicted output.
- Both botnet type classification and Mirai attack type classification showed very good results on the whole 115 feature set with accuracies over 0.99.