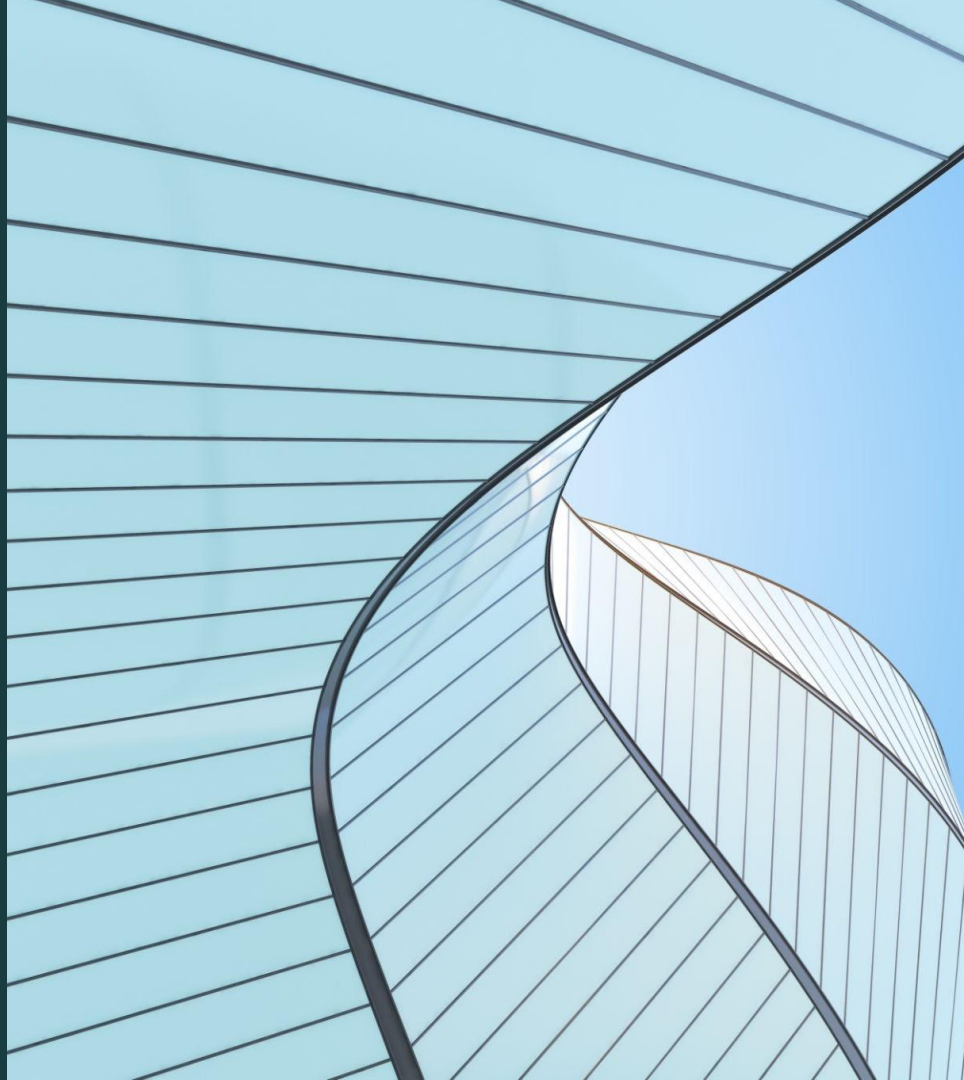
SENG 4610 - Applications of Machine Learning to Software Engineering

Machine-Learning Based Cyber Security

Ruth Befikadu: T00696672
Shaylee Broadfoot: T00551934



Content *overview*

Introduction

- Cyber-attacks are rapidly increasing, with global losses reaching US $16.6B in 2024 [1].
- Traditional rule-based IDS struggle with new or evolving attacks, leading to high false-negative rates [2].
- ML-based detection learns behavioural patterns, improving accuracy across attack types.
- Challenges include data requirements, overfitting, and computational limits for low-power devices [3].
- This project builds an efficient ML attack classifier to detect threats traditional systems miss.

Design Requirements

Functions

- The system will read and process data
- Clean and normalize data
- Extract features
- Train on one or more model
- Store and display classification
- Predict unknown class
- Allow parameter adjustment

Objectives

- The design should make accurate and consistent predictions
- Minimize false alarms
- Operate efficiently
- Flexible
- Ensure data integrity
- Maintainable
- Clear output

Design Requirements

Constraints

- Must use datasets and features provided
- Use python
- Fit within timeframe
- Follow ethical and academic guidelines
- Reproducible design

Solution 1

Support Vector Machine

Why a Support Vector Machine (SVM)?

- Strong performance on high-dimensional data
- Effective at separating classes using margins
- RBF kernel can model non-linear attack patterns
- Well-established baseline for intrusion detection research

SVM Preprocessing Overview

- Cleaned service/proto/state and grouped them into broader categories
- Added simple engineered features (bytes total, packet ratios, TTL diff, etc.)
- Removed noisy/unhelpful fields (IPs, timestamps)
- Log-transformed and scaled numeric features (critical for RBF kernel)
- One-hot encoded category groups

Binary SVM Results

Hyperparameters

- **Kernel:** RBF
- **C:** 9.0
- **Gamma:** 0.05
- **Class Weight:** Balanced
- **Decision Function Shape:** One-vs-Rest (OvR)
- **Probability Estimates:** False

Performance

```
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.96      0.91     11200
           1       0.98      0.93      0.96     23869

    accuracy                           0.94     35069
   macro avg       0.93      0.95      0.93     35069
weighted avg       0.95      0.94      0.94     35069
```

Multiclass SVM Results

Hyperparameters

- **Kernel:** RBF
- **C:** 1.0
- **Gamma:** 0.075
- **Class Weight:** None
- **Decision Function Shape:** One-vs-Rest (OvR)
- **Probability Estimates:** False

Performance

Classification Report:

	precision	recall	f1-score	support
0	0.71	0.19	0.30	400
1	0.91	0.06	0.11	349
2	0.63	0.03	0.06	2453
3	0.62	0.96	0.75	6679
4	0.91	0.87	0.89	3637
5	1.00	0.98	0.99	8000
7	0.86	0.73	0.79	2098
8	0.62	0.41	0.50	227
9	0.80	0.15	0.26	26
accuracy			0.80	23869
macro avg	0.78	0.49	0.52	23869
weighted avg	0.82	0.80	0.76	23869

SVM Efficiency & Generalization

Binary SVM

- **Model File Size:** 9.6 MB
- **Training Speed:** 205.265 s
- **Prediction Speed:** 35.387 s
- **Peak Memory During Prediction:** 49.025 MB
- **Train RMSE:** 0.2356
- **Test RMSE:** 0.2417

Multiclass SVM

- **Model File Size:** 24.194 MB
- **Training Speed:** 150.383 s
- **Prediction Speed:** 56.629 s (2.3725 ms per sample)
- **Peak Memory During Prediction:** 50.452 MB
- **Train RMSE:** 0.9976
- **Test RMSE:** 0.9908

SVM Summary

- Heaviest preprocessing required
- Largest model file size
- Moderate training speed (slower than solution 2, faster than solution 3
- Highest peak memory usage during prediction
- Slowest prediction times
- Best generalization
- Lowest binary and multiclass scores except binary average recall (2nd best), although very comparable to solution 2

Solution 2

Random Forest

Why Random Forest (RF)?

- Handles raw, unscaled features with minimal preprocessing
- Naturally models non-linear relationships
- Good resistance to noise and overfitting through ensemble averaging
- Fast to train and predict, even on large datasets

RF Preprocessing Overview

- Removed the id column.
- Kept all numeric features as-is (no scaling or transforms)
- One-hot encoded categorical fields (proto, service, state, etc.)
- Used the same binary and multiclass label splits as the other models
- No additional feature engineering or normalization required for RF

Binary RF Results

Hyperparameters

- **Number of Trees:** 20
- **Max Depth:** 16
- **Min Samples Split:** 10
- **Min Samples Leaf:** 5
- **Max Features per Split:** 0.35
- **Class Weight:** Balanced
- **Bootstrap:** False

Performance

```
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.96      0.92     11200
           1       0.98      0.94      0.96     23869

    accuracy                           0.95     35069
   macro avg       0.94      0.95      0.94     35069
weighted avg       0.95      0.95      0.95     35069
```

Multiclass RF Results

Hyperparameters

- **Number of Trees:** 60
- **Max Depth:** 12
- **Min Samples Split:** 8
- **Min Samples Leaf:** 4
- **Max Features per Split:** 0.2
- **Class Weight:** None
- **Bootstrap:** True

Performance

Classification Report:

	precision	recall	f1-score	support
Analysis	0.78	0.21	0.33	400
Backdoor	0.83	0.09	0.16	349
DoS	0.50	0.06	0.10	2453
Exploits	0.62	0.95	0.75	6679
Fuzzers	0.95	0.89	0.92	3637
Generic	1.00	0.98	0.99	8000
Reconnaissance	0.93	0.76	0.84	2098
Shellcode	0.71	0.70	0.70	227
Worms	0.80	0.15	0.26	26
accuracy			0.82	23869
macro avg	0.79	0.53	0.56	23869
weighted avg	0.82	0.82	0.78	23869

RF Efficiency & Generalization

Binary RF

- **Model File Size:** 3.768 MB
- **Training Speed:** 1.235 s
- **Prediction Speed:** 0.023 s (0.0007 ms per sample)
- **Peak Memory During Prediction:** 41.873 MB
- **Train RMSE:** 0.1873
- **Test RMSE:** 0.2245

Multiclass RF

- **Model File Size:** 10.263 MB
- **Training Speed:** 0.890 s
- **Prediction Speed:** 0.033 s (0.0014 ms per sample)
- **Peak Memory During Prediction:** 42.536 MB
- **Train RMSE:** 0.8042
- **Test RMSE:** 0.8238

RF Summary

- Lightest preprocessing required
- Moderate model file size (less than solution 1, more than solution 3)
- Fastest training and prediction speeds
- Moderate peak memory usage during prediction (less than solution 1, more than solution 3)
- Worst binary generalization, although RMSEs still indicate a good fit
- Multiclass generalization about the same as solution 3
- Worst binary recall, but best multiclass precision
- Second best performance metrics in remaining categories

Solution 3

ResNet and XGBoost

Why a Neural Network (NN)?

- Learn complex, non-linear patterns in data that simpler models cannot capture
- Handle high-dimensional features
- automatically extract useful patterns without manual feature engineering
  - embeddings
  - residual layers
  - dense transformations

sklearn MLP and a Keras residual MLP

Why a XGBoost?

- Work well with tabular data
- Sharper
- Good with threshold based data
- Gives us good recall for minority class

sklearn MLP and a Keras residual MLP

Preprocessing Overview

- Drop ID, IP, and time
- Junk data => unknown
- Encode attack or not and then the different attacks
- Separate numeric and categorical columns
  - Numeric => median imputer & standard scaler
  - Categoric => constant imputer & oneHotEncoder

Solution 3

Components

- MLP CLassifier
  - 128, 64 deep

- ResNet (2 models with transfer learning)
  - 2 blocks with 265 neurons
  - Skip connections

Solution 3

- Base accuracy: 85.67%
- Precision: 73.76%

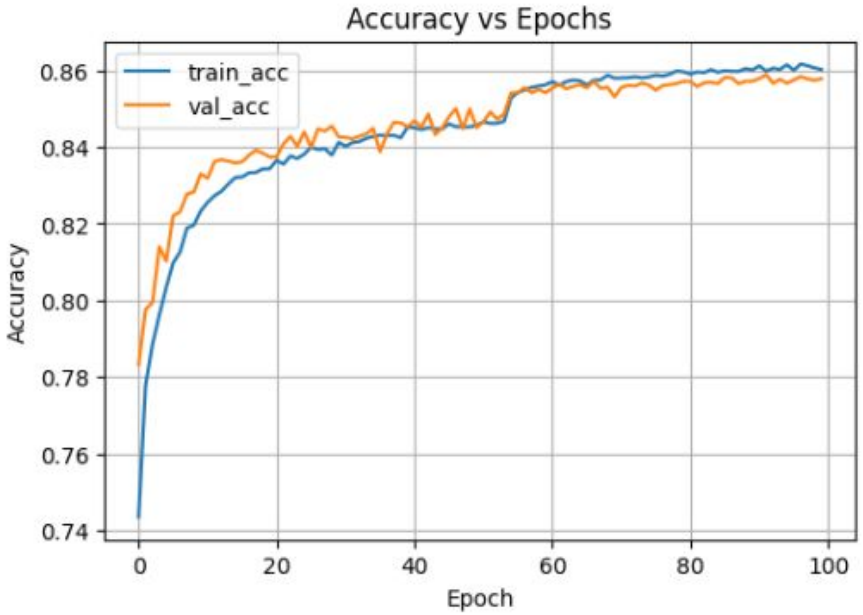
Classification report (Keras multiclass):

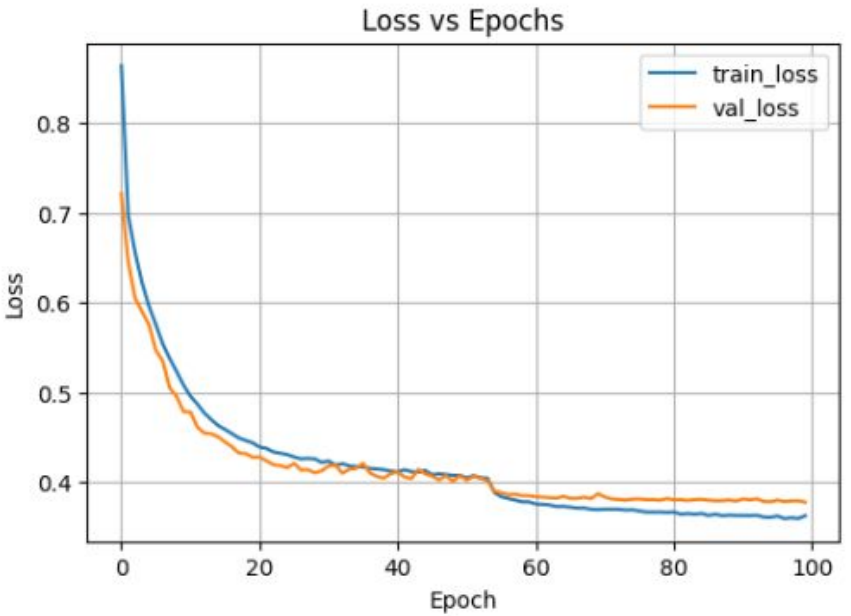
	precision	recall	f1-score	support
0	0.7723	0.1950	0.3114	400
1	0.6533	0.1404	0.2311	349
2	0.4214	0.1093	0.1735	2453
3	0.6285	0.9163	0.7456	6679
4	0.8713	0.8243	0.8471	3637
5	0.9941	0.9849	0.9895	8000
6	0.9763	0.9761	0.9762	11200
7	0.9108	0.7450	0.8196	2098
8	0.6483	0.6740	0.6609	227
9	0.5000	0.0769	0.1333	26
accuracy			0.8567	35069
macro avg	0.7376	0.5642	0.5888	35069
weighted avg	0.8525	0.8567	0.8387	35069

Confusion matrix (Keras multiclass):

```
[[   78    16    26   273     0     0     7     0     0     0]
 [    0    49    26   262     3     2     1     2     4     0]
 [    0     4   268  2102    27    13     1    15    23     0]
 [   13     4   231  6120   135    23    22   104    25     2]
 [    6     2    28   329  2998     4   230    15    25     0]
 [    2     0    25    80     8  7879     0     1     5     0]
 [    1     0     1    27   230     0 10932     8     1     0]
 [    1     0    30   498     4     2     0  1563     0     0]
 [    0     0     1    26    35     0     4     8   153     0]
 [    0     0     0    20     1     3     0     0     0     2]]
```

Solution 3


Accuracy vs Epochs
train_acc
val_acc
Accuracy
Epoch
0.86
0.84
0.82
0.80
0.78
0.76
0.74
0
20
40
60
80
100

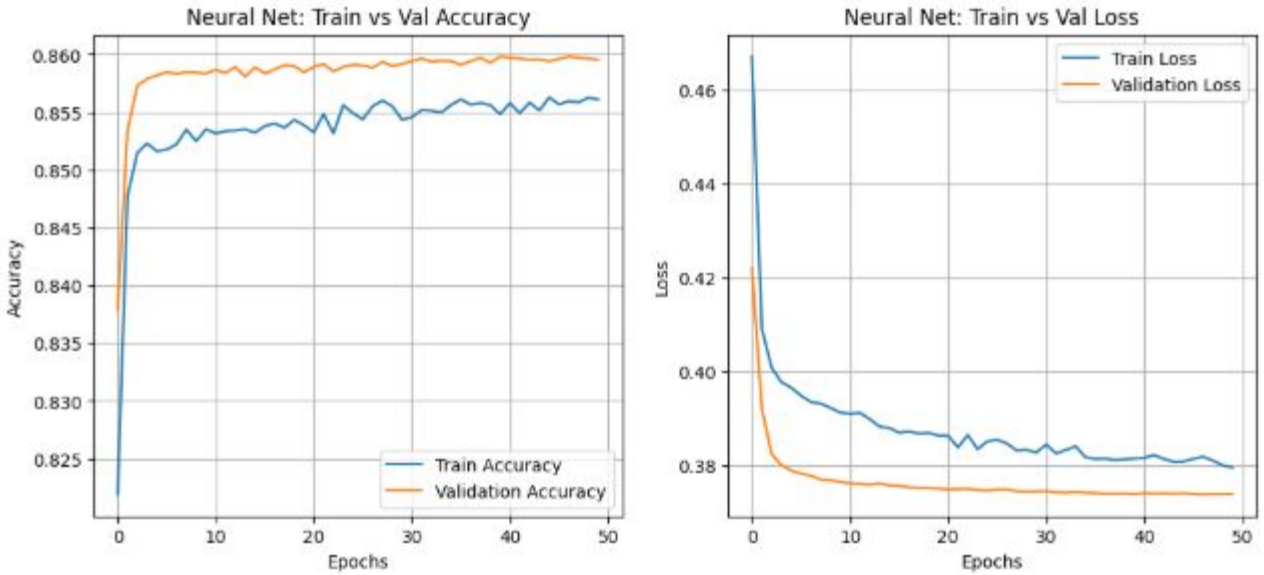

Loss vs Epochs
train_loss
val_loss
Loss
Epoch
0.8
0.7
0.6
0.5
0.4
0
20
40
60
80
100

Solution 3

- Transfer learning to a second model
- Freeze previous layer and add new head
  - 2 dense layer: 256
  - 1 layer: 128
  - Learning rate: 0.001
  - Train

- Unfreeze and train again

Solution 3


Neural Net: Train vs Val Accuracy
Accuracy
Epochs
Train Accuracy
Validation Accuracy
Neural Net: Train vs Val Loss
Loss
Epochs
Train Loss
Validation Loss

Solution 3

- Hybrid ensemble
  - Add XGBoost : sharper and makes a different mistake compared to NN
  - XGBoost was tuned with:
    - 200 estimators
    - Learning rate: 0.05
    - Depth: 10
    - Use of histogram - speed, data compression
  - Assigned probability based split (0.1,0.9)

Solution 3

- Accuracy: 87.21%
- Precision: 77.98%

```
================ FINAL OPTIMIZED REPORT ================

              precision    recall  f1-score   support

           0     0.7500    0.2025    0.3189       400
           1     0.7681    0.1519    0.2536       349
           2     0.4077    0.2719    0.3262      2453
           3     0.6670    0.8820    0.7596      6679
           4     0.9414    0.8708    0.9047      3637
           5     0.9979    0.9864    0.9921      8000
           6     0.9900    0.9894    0.9897     11200
           7     0.9205    0.7450    0.8235      2098
           8     0.7306    0.7885    0.7585       227
           9     0.6250    0.3846    0.4762        26

    accuracy                         0.8721     35069
   macro avg     0.7798    0.6273    0.6603     35069
weighted avg     0.8735    0.8721    0.8644     35069
```

```
Confusion Matrix:
[[   81    14    79   217     1     0     8     0     0     0]
 [   12    53    73   198     4     2     1     2     4     0]
 [    3     0   667  1724    18     7     0    15    19     0]
 [    6     0   569  5891    68     7    11   105    17     5]
 [    6     2    85   268  3167     0    88     2    19     0]
 [    0     0    47    54     5  7891     0     0     3     0]
 [    0     0     5    25    80     0 11081     6     2     1]
 [    0     0   110   419     4     0     1  1563     1     0]
 [    0     0     1    24    16     0     3     4   179     0]
 [    0     0     0    12     1     1     0     1     1    10]]
```

ResNet & XGBoost Efficiency & Generalization

Binary

- **Model File Size:** 793 KB
- **Training Speed:** 3.5 minutes
- **Prediction Speed:** < 1 s
- **Peak Memory During Prediction:** 2.22 MB
- **Train RMSE:** 0.1052
- **Test RMSE:** 0.1096

Multiclass

- **Model File Size:** 14 MB + 4.57 MB
- **Training Speed:** ~ 39 minutes
- **Prediction Speed:** ~ 9 s
- **Peak Memory During Prediction:** 11.649 MB
- **Train RMSE:** 0.4962
- **Test RMSE:** 0.6689

Performance Comparison of Solutions (%)

Metric		Solution 1		Solution 2		Solution 3	
		Binary	Multiclass	Binary	Multiclass	Binary	Multiclass
Positive Class	Precision	98	-	98	-	**99**	-
	Recall	93	-	94	-	**99**	-
	F1-Score	96	-	96	-	**99**	-
Macro Average	Precision	93	78	94	**79**	**98**	78
	Recall	95	49	94	53	**98**	**63**
	F1-Score	93	52	94	56	**98**	**66**
	Accuracy	94	80	95	82	**98**	**87**

Environmental Considerations

- Energy reduction:
  - Efficiency
  - Small models
  - Fast timing

Safety Considerations

- Prevent harmful actions
  - Safe
  - Auditable
  - Privacy

Societal Considerations

- Protection
  - Reduce false alarm
  - Reliability
  - ethics

Economic Considerations

- Running effectively
  - Open-source
  - Financial shield
  - Downtime reduction

Limitations

- Heavy dependence on limited data set
- Evolving attacks may not be immediately detected
- Data imbalance and quality issues (weights)
- Relatively heavy computation (ultra low devices)
- Retraining (not automated)

References

[1] Federal Bureau of Investigation, Internet Crime Report 2024, Internet Crime Complaint Center (IC3), Feb. 2025. [Online]. Available: https://www.ic3.gov/AnnualReport/Reports/2024_IC3Report.pdf

[2] World Economic Forum, Global Cybersecurity Outlook 2025, Geneva, Switzerland, Jan. 2025. [Online]. Available: https://reports.weforum.org/docs/WEF_Global_Cybersecurity_Outlook_2025.pdf

[3] IBM, Cost of a Data Breach Report 2024, IBM Security and the Ponemon Institute, Armonk, NY, USA, 2024. [Online]. Available: https://www.ibm.com/reports/data-breach

Thank you