

Comparison between CNN and RF

CNN gives me the impression that it really learned and started to think on its own when checking if there is an attack or not, compared with Random Forest.

CNN at some attacks, gives an 100% confidence that it is an attack, while the random forest has the confidence of 70% at the same attack.

This is no overfitting. I tested it with safe queries, where it had the confidence very close to 0 (e^{-15}) that that is an attack, resulting in returning that is a safe query.

Random forest is showing poorer result when given a new attack, however, both of them had a recall of 100% on training and testing the dataset.

In an edge case, i used the word “supercalifragilisticexpialidocious” to see what their response was. RF was 100% certain this is an attack, showing that it didn't learn much, but merely is relying on memorization . If he doesn't know a word, it just goes into the extreme that that is an attack. The CNN on the other hand, had 95% confidence that this was an attack. This tells me that he thought about it, he doesnt know the word, but because he doesn't know it, he chooses to classify it as an attack. He is also biased toward attack classifiers, because i boosted the attack weight with 1.5.

CNN:

```
Epoch 1/5
2500/2500 ————— 21s 8ms/step - accuracy: 0.9944 - loss: 0.0239 - val_accuracy: 1.0000 - val_loss: 1.9671e-09
Epoch 2/5
2500/2500 ————— 11s 4ms/step - accuracy: 1.0000 - loss: 0.0046 - val_accuracy: 1.0000 - val_loss: 2.4389e-11
Epoch 3/5
2500/2500 ————— 10s 4ms/step - accuracy: 1.0000 - loss: 0.0020 - val_accuracy: 1.0000 - val_loss: 2.4374e-11
Epoch 4/5
2500/2500 ————— 10s 4ms/step - accuracy: 1.0000 - loss: 8.8104e-04 - val_accuracy: 1.0000 - val_loss: 2.9618e-13
Epoch 5/5
2500/2500 ————— 10s 4ms/step - accuracy: 1.0000 - loss: 3.8407e-04 - val_accuracy: 1.0000 - val_loss: 1.5360e-13
625/625 ————— 1s 1ms/step
Confusion Matrix:
[[10065   0]
 [   0 9935]]
Recall: 1.0000.
```

```
PS C:\Users\sbghe> D:
PS D:\> cd Master\An2\ICS
PS D:\Master\An2\ICS> python attack_test.py
Testing 100000 queries...
Threads: 20 | Target: http://localhost:5000/check
Progress: 99.5% | Speed: 9 req/sec | Detected: 99501
```

```
=====
FINAL REPORT (Time: 11719.57s)
=====
Total Queries:      100000
Detected Attacks:   100000
Recall Score:       100.00%
=====
```

CNN with cleaning:

```
Progress: 99.8% | Speed: 9 req/s | Detected: 99793

=====
FINAL RESULTS (Time: 10593.06s)
=====
Total Queries:      100000
Detected Attacks:   99992
RECALL SCORE:       99.99%
=====
```

Testing Queries

CNN:

Query	safe/attack	classified
INSERT INTO books (title) VALUES ('The Union of Select Few') SELECT * FROM products WHERE description = 'This supports 1=1 connection' UPDATE settings SET val = 'Do not forget to /* comment */ your code' SEARCH 'admin'	safe	safe
OR '500'='500	attack	attack
OR 1=1 /*	attack	attack
admin' '1	attack	attack
' UNION SELECT 1	attack	attack
' un%69on se%6cect 1	attack	attack
SE%20LECT * FROM use%72s	attack	attack
UPDATE calculator SET result = 100 WHERE formula = '10*10=100'	safe	safe
'SELECT * FROM products WHERE description LIKE '%0x%'	safe	safe
supercalifragilisticexpialidocio us	safe	attack

Random Forest

```
Accuracy: 1.0000
Recall: 1.0000
Confusion Matrix:
[[10065    0]
 [    0 9935]]
```

Query	safe/attack	classified
INSERT INTO books (title) VALUES ('The Union of Select Few') SELECT * FROM products WHERE description = 'This supports 1=1 connection' UPDATE settings SET val = 'Do not forget to /* comment */ your code' SEARCH 'admin'	safe	safe
OR '500'='500	attack	attack
OR 1=1 /*	attack	attack
admin' '1	attack	attack
' UNION SELECT 1	attack	Attack (coff 0.6)
' un%69on se%6cect 1	attack	attack
SE%20LECT * FROM use%72s	attack	attack(0.7)
UPDATE calculator SET result = 100 WHERE formula = '10*10=100'	safe	attack(0.5)
supercalifragilisticexpialidocio us	safe	attack