EDA JSMA

November 16, 2020

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import keras
```

```
Using TensorFlow backend.
```

[3]:
```
x_test = np.load('./ATTACKS/JSMA/X_TEST_JSMA.npy')
y_test = np.load('./ATTACKS/JSMA/Y_TEST_JSMA.npy')
x_n_test = np.load('./ATTACKS/JSMA/X_TEST_ATTACKED_JSMA.npy')
```

1 MAL AND BEN ROWS

[4]:
```
MAL = []
BEN = []
for i in range(y_test.shape[0]):
    if y_test[i]==1:
        MAL.append(i)
    else:
        BEN.append(i)
print(len(MAL),len(BEN))
```

```
1667 3333
```

2 Classifier

[5]:
```
from keras.utils import to_categorical
test_labels = to_categorical(y_test)
```

[6]:
```
network = keras.models.load_model('./ATTACKS/JSMA/JSMA_CLASSIFIER_USED.h5py')
network.summary()
```

```
WARNING:tensorflow:From C:\Users\Pitch\.conda\envs\tf1-gpu\lib\site-
packages\tensorflow_core\python\ops\resource_variable_ops.py:1630: calling
BaseResourceVariable.__init__ (from tensorflow.python.ops.resource_variable_ops)
with constraint is deprecated and will be removed in a future version.
Instructions for updating:
```

```
If using Keras pass *_constraint arguments to layers.
WARNING:tensorflow:From C:\Users\Pitch\.conda\envs\tf1-gpu\lib\site-
packages\keras\backend\tensorflow_backend.py:422: The name tf.global_variables
is deprecated. Please use tf.compat.v1.global_variables instead.

Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
reshape_1 (Reshape)          (None, 2304)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               1180160
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 1026
=================================================================
Total params: 1,181,186
Trainable params: 1,181,186
Non-trainable params: 0
_________________________________________________________________
```

3 Find basic overall drop in accuracy

[7]:
```
network.evaluate(x_test,test_labels)
```

```
5000/5000 [==============================] - 3s 567us/step
```

[7]:
```
[0.19247934875786304, 0.9476000070571899]
```

[8]:
```
network.evaluate(x_n_test,test_labels)
```

```
5000/5000 [==============================] - 0s 53us/step
```

[8]:
```
[0.8315978041648865, 0.6453999876976013]
```

[9]:
```
print("DROP")
print((network.evaluate(x_test,test_labels)[1]-network.
 ↪evaluate(x_n_test,test_labels)[1])*100)
```

```
DROP
5000/5000 [==============================] - 0s 53us/step
5000/5000 [==============================] - 0s 48us/step
30.220001935958862
```

4 But we just attacked Malware. So lets find that particular drop

[10]:
```
network.evaluate(x_test[MAL],test_labels[MAL])
```

```
1667/1667 [==============================] - 0s 79us/step
```

[10]:
```
[0.3696391213490853, 0.9064187407493591]
```

[11]:
```
network.evaluate(x_n_test[MAL],test_labels[MAL])
```

```
1667/1667 [==============================] - 0s 86us/step
```

[11]:
```
[2.2866111119206822, 0.0]
```

[12]:
```
print("DROP")
print((network.evaluate(x_test[MAL],test_labels[MAL])[1]-network.
 ↪evaluate(x_n_test[MAL],test_labels[MAL])[1])*100)
```

```
DROP
1667/1667 [==============================] - 0s 74us/step
1667/1667 [==============================] - 0s 67us/step
90.64187407493591
```

4.0.1 In JSMA, we made sure that a MAL becomes a BEN so we see an ideal drop of the whole 90.64187%

5 First, We are gonna construct a change matrix, the changes from x_test and x_n_test.

5.1 Sanity Check

[13]:
```
same = 0
one2zero = 0
zero2one = 0
nosense = 0
for i in range(x_test.shape[0]):
    for j in range(x_test.shape[1]):
        for k in range(x_test.shape[2]):
#             print(x_test[i][j][k],x_n_test[i][j][k])
            if x_test[i][j][k] == x_n_test[i][j][k]//0.5:
                same+=1
            elif x_test[i][j][k] == 1 and x_n_test[i][j][k]//0.5 == 0:
                one2zero+=1
            elif x_test[i][j][k] == 0 and x_n_test[i][j][k]//0.5 == 1:
                zero2one+=1
            else:
                nosense+=1
print(same,one2zero,zero2one,nosense)
```

```
11029669 0 0 490331
```

[14]:
```
same = 0
one2zero = 0
zero2one = 0
nosense = 0
changed = []
for i in range(x_test.shape[0]):
    for j in range(x_test.shape[1]):
        for k in range(x_test.shape[2]):
            if x_test[i][j][k] == x_n_test[i][j][k]:
                same+=1
            else:
                nosense+=1
print(same,nosense)
```

```
11512400 7600
```

The attacked added a much smaller number (just 7600) of bits compared to FGSM

[15]:
```
same = 0
c1 = 0
c2 = 0
changedRows = []
for i in range(x_test.shape[0]):
    for j in range(x_test.shape[1]):
        for k in range(x_test.shape[2]):
            if x_test[i][j][k] == x_n_test[i][j][k]:
                same+=1
            elif x_test[i][j][k] == 0:
                c1+=1
#                 x_n_test_copy[i][j][k] = 1
            elif x_test[i][j][k] == 1:
                c2+=1
    if x_test[i].tolist() != x_n_test[i].tolist():
        changedRows.append(i)

print(same,c1,c2,len(changedRows))
```

```
11512400 7600 0 1511
```

- No features were removed
- 7600 features were added
- out of the 1667 features, just 1511 were attacked successfully.

Find diff matrix to find which features were added

[16]:
```
changedMat = np.zeros(x_test.shape)
```

[17]:
```
for i in range(x_test.shape[0]):
    for j in range(x_test.shape[1]):
        for k in range(x_test.shape[2]):
            if x_test[i][j][k] != x_n_test[i][j][k]:
                changedMat[i][j][k] = 1
print(same,nosense)
```

```
11512400 7600
```

[18]:
```
np.unique(changedMat,return_counts=True)
```

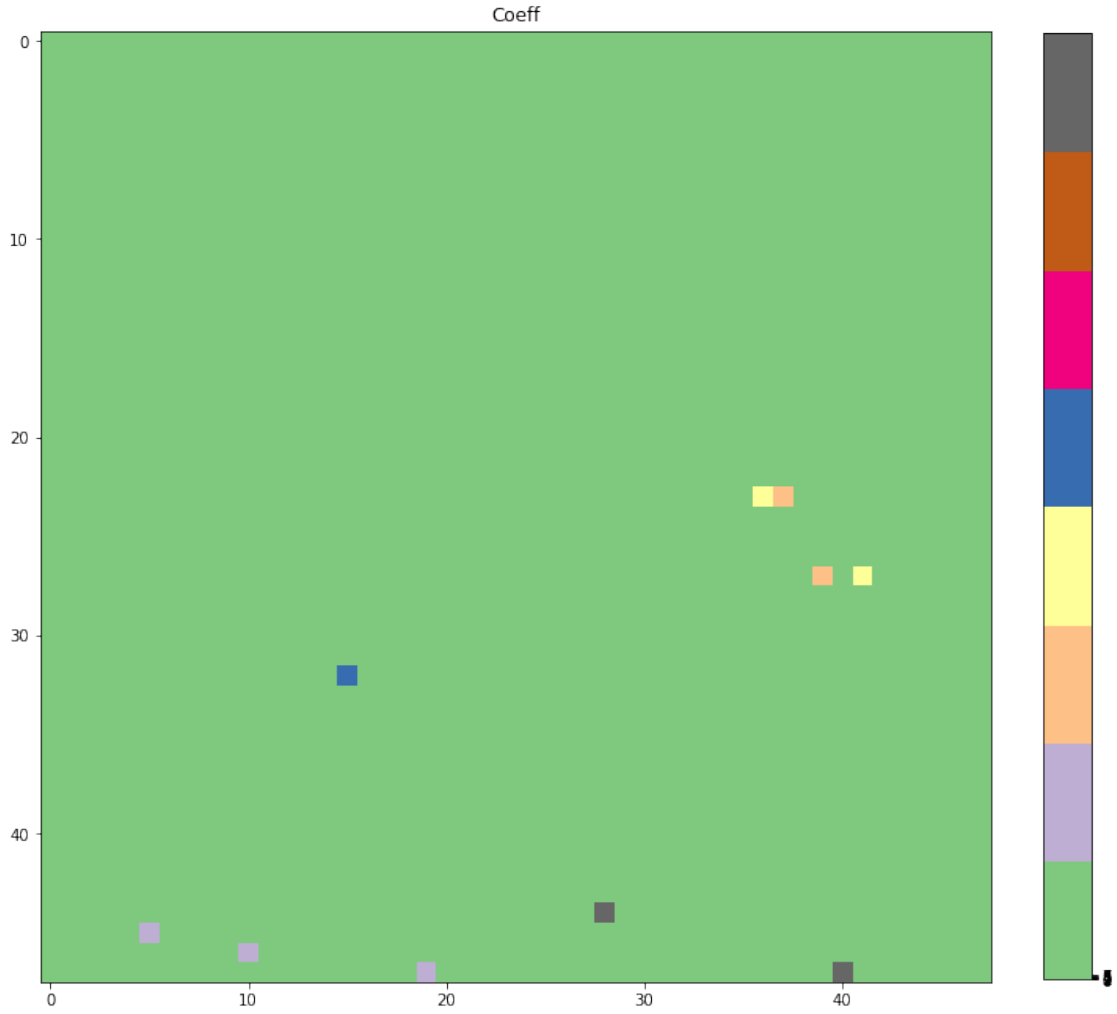
[18]:
```
(array([0., 1.]), array([11512400,     7600], dtype=int64))
```

[19]:
```
fig, ax = plt.subplots(figsize = (12,12))
ax.set_title('Coeff')
cax = ax.imshow(np.sum(changedMat,axis = 0), cmap = plt.cm.Accent)

cbar = plt.colorbar(cax, ticks=[0, 1, 2, 3, 4, 5, 6, 7],
                    orientation='vertical',
                    fraction=0.045, pad=0.05)

print("GREEN IS LOW, BLACK IS HIGH")
print("GREEN MIN : "+ str(int( min( np.sum(changedMat,axis=0).reshape(-1,1) ) )␣
 ↪))
print("BLAC MAX : "+ str(int( max( np.sum(changedMat,axis=0).reshape(-1,1) ) )␣
 ↪))
```

```
GREEN IS LOW, BLACK IS HIGH
GREEN MIN : 0
BLAC MAX : 1509
```



6 Observation

As we can see, JSMA adds a lot less features but is equally effective. Only downside is the time it takes.

And it is very clear that the only feature it adds often are in the lower half of the plot, i.e, they are benign features

A single BEN feature is added 1509 times out of the 1511 successfull attacks, making it common to 99.867% of the attacks

7 Lets see how any features were ever touched by JSMA

[70]:
```
sumCM = np.sum(changedMat,axis = 0)
cx = []
for i in range(sumCM.shape[0]):
    for j in range(sumCM.shape[1]):
        if sumCM[i][j] != 0:
            cx.append([i,j])
len(cx)
```

[70]: 29

7.1 JSMA only touched 29 of the 2304 features throughout the dataset and still managed to get a drop in accuracy comparable to FGSM.

8 Lets see is any features were changed over 150 times (10% of the dataset)

8.0.1 In FGSM, we checked for apparances over 1525 after realizing that it added a lot of features, but given that JSMA is a much more efficient attack, we are taking a better limit of 10%

[61]:
```
sumCM = np.sum(changedMat,axis = 0)
c150 = []
cx = 0
```

[62]:
```
for i in range(sumCM.shape[0]):
    for j in range(sumCM.shape[1]):
        if sumCM[i][j] >= 150:
            c150.append([i,j])
        if sumCM[i][j] != 0:
            cx+=1
```

[63]:
```
len(c150)
```

[63]: 12

just 12 features were changed over 150 times in the over 1500 attacks. This shows how effective JSMA is in masking a result with very little pertubations

9 Lets see is any features were changed under 15 times (1% of the dataset) and just 5 time

9.0.1 To get an idea of how selective JSMA is

[64]:
```
sumCM = np.sum(changedMat,axis = 0)
c15 = []
c5 = []
```

[65]:
```
for i in range(sumCM.shape[0]):
    for j in range(sumCM.shape[1]):
        if sumCM[i][j] <= 15 and sumCM[i][j] > 0:
            c15.append([i,j])
        if sumCM[i][j] <= 5 and sumCM[i][j] > 0:
            c5.append([i,j])
```

[66]:
```
len(c15)
```

[66]: 11

[67]:
```
len(c5)
```

[67]: 9

9.0.2 We have just considered features that were changed atleast once

11 features were changed less than 15 times out of which 9 features were changed less than 5 times which goes on to show how effective JSMA is

[ ]:
```

```

9.0.3 Lets see all the 29 features that were touched by JSMA with their frequencies

[68]:
```
features = np.load("./DATA/FeatureList.npy", allow_pickle=True)
coeff = np.load("./DATA/coeff_features.npy", allow_pickle=True)
```

[69]:
```
from termcolor import colored
```

[71]:
```
for i,j in cx:
    print("============================================================")
    print("FREQ of addition")
    print(sumCM[i][j])
    print("----------")
    print("NATURE OF FEATURE")
    if coeff[i][j] >= 0:
        print(colored("MAL","red"))
        print(colored(coeff[i][j],'red'))
```

```
    else:
        print(colored("BEN","blue"))
        print(colored(coeff[i][j],'blue'))
    print("----------")
    print("FEATURE NAME")
    print(features[i][j])
    print("============================================================")
```

```
============================================================
FREQ of addition
19.0
----------
NATURE OF FEATURE
MAL
0.09855355781714814
----------
FEATURE NAME
urls::https://play_google_com/store/apps/developer?id=
============================================================
============================================================
FREQ of addition
122.0
----------
NATURE OF FEATURE
MAL
0.010617190479564713
----------
FEATURE NAME
urls::http://img_youtube_com/vi/
============================================================
============================================================
FREQ of addition
720.0
----------
NATURE OF FEATURE
MAL
5.637851296924623e-18
----------
FEATURE NAME
urls::http://www_amazon_com/gp/mas/dl/
============================================================
============================================================
FREQ of addition
457.0
----------
NATURE OF FEATURE
MAL
```

4.7704895589362195e-18

FEATURE NAME
urls::http://www_andromo_com/?utm_source=about&utm_medium=app&utm_campaign=andro
mo_app
==========
==========
FREQ of addition
536.0

NATURE OF FEATURE
BEN
-1.3877787807814457e-17

FEATURE NAME
activities::com_kingDev_guidefor_howto_Details
==========
==========
FREQ of addition
28.0

NATURE OF FEATURE
BEN
-1.3877787807814457e-17

FEATURE NAME
activities::com_kingDev_guidefor_howto_MainActivity
==========
==========
FREQ of addition
651.0

NATURE OF FEATURE
BEN
-1.3877787807814457e-17

FEATURE NAME
activities::com_kingDev_guidefor_howto_ListViewsItems
==========
==========
FREQ of addition
876.0

NATURE OF FEATURE
BEN
-0.020673951694876916

FEATURE NAME

```
urls::https://airdownload2_adobe_com/air?
============================================================
============================================================
FREQ of addition
185.0
----------
NATURE OF FEATURE
BEN
-0.03919459835429477
----------
FEATURE NAME
interesting_calls::Cipher(Lcom/google/android/gms/internal/zzdss)
============================================================
============================================================
FREQ of addition
8.0
----------
NATURE OF FEATURE
BEN
-0.05050739862398803
----------
FEATURE NAME
urls::http://www_startapp_com/policy/sdk-policy/
============================================================
============================================================
FREQ of addition
3.0
----------
NATURE OF FEATURE
BEN
-0.05313209161302148
----------
FEATURE NAME
activities::ti_modules_titanium_ui_android_TiPreferencesActivity
============================================================
============================================================
FREQ of addition
22.0
----------
NATURE OF FEATURE
BEN
-0.05464158129052617
----------
FEATURE NAME
s_and_r::com_google_android_gcm_GCMBroadcastReceiver
============================================================
============================================================
FREQ of addition
```

```
1.0
----------
NATURE OF FEATURE
BEN
-0.05698942869434507
----------
FEATURE NAME
api_calls::android/app/WallpaperManager;->setBitmap
============================================================
============================================================
FREQ of addition
1.0
----------
NATURE OF FEATURE
BEN
-0.05727251646910284
----------
FEATURE NAME
activities::_LoginActivity
============================================================
============================================================
FREQ of addition
161.0
----------
NATURE OF FEATURE
BEN
-0.0845774816581817
----------
FEATURE NAME
urls::https://www_googleapis_com/games/v1management/achievements/reset?access_to
ken=
============================================================
============================================================
FREQ of addition
53.0
----------
NATURE OF FEATURE
BEN
-0.09662252494844528
----------
FEATURE NAME
urls::https://www_mopub_com/optout/
============================================================
============================================================
FREQ of addition
4.0
----------
NATURE OF FEATURE
```

BEN
-0.1047649361466993

FEATURE NAME
urls::http://tempuri_org/
==
==
FREQ of addition
5.0

NATURE OF FEATURE
BEN
-0.10560445085126714

FEATURE NAME
urls::https://docs_coronalabs_com/guide/monetization/IAP/index_html
==
==
FREQ of addition
1.0

NATURE OF FEATURE
BEN
-0.11285378081057476

FEATURE NAME
urls::https://adservice_google_com/getconfig/pubvendors
==
==
FREQ of addition
1455.0

NATURE OF FEATURE
BEN
-0.14851161650834463

FEATURE NAME
activities::Scanner
==
==
FREQ of addition
313.0

NATURE OF FEATURE
BEN
-0.16523922333638594

FEATURE NAME

```
activities::_ActivitySplash
============================================================
============================================================
FREQ of addition
16.0
----------
NATURE OF FEATURE
BEN
-0.22099689093167246
----------
FEATURE NAME
interesting_calls::Cipher(Lcom/google/android/gms/internal/zzdjd)
============================================================
============================================================
FREQ of addition
2.0
----------
NATURE OF FEATURE
BEN
-0.2272997786853023
----------
FEATURE NAME
s_and_r::cmcm_com_keyboard_themeapk_base_utils_CampaignTrackingReceiver
============================================================
============================================================
FREQ of addition
235.0
----------
NATURE OF FEATURE
BEN
-0.2308951643138094
----------
FEATURE NAME
activities::_SplashScreenActivity
============================================================
============================================================
FREQ of addition
9.0
----------
NATURE OF FEATURE
BEN
-0.2714217136492844
----------
FEATURE NAME
urls::http://cml_ksmobile_com/api/controller_php
============================================================
============================================================
FREQ of addition
```

```
201.0
----------
NATURE OF FEATURE
BEN
-0.3389940651871172
----------
FEATURE NAME
app_permissions::name='android_permission_WRITE_INTERNAL_STORAGE'
============================================================
============================================================
FREQ of addition
3.0
----------
NATURE OF FEATURE
BEN
-0.39802143518660854
----------
FEATURE NAME
interesting_calls::Cipher(DES/ECB/NoPadding)
============================================================
============================================================
FREQ of addition
4.0
----------
NATURE OF FEATURE
BEN
-0.4033787182843053
----------
FEATURE NAME
urls::https://ssl_gstatic_com/accessibility/javascript/android/
============================================================
============================================================
FREQ of addition
1509.0
----------
NATURE OF FEATURE
BEN
-0.4465655338702529
----------
FEATURE NAME
activities::_PTPlayer
============================================================
```

10 Observations

- out of the 29 features that were added, 4 were MAL and 25 were BEN.
- Not to say that MAL additions are rare, feature named

urls::http://www_amazon_com/gp/mas/dl/ was added 720 times and urls::http://www_andromo_com/?utm_source=about&utm_medium=app&utm_campaign=andromo_ap was added 457 times.

- But it is clear that BEN features are added more often, with
  - activities::com_kingDev_guidefor_howto_Details
    - 536 times
  - activities::com_kingDev_guidefor_howto_ListViewsItems
    - 651 times
  - urls::https://airdownload2_adobe_com/air?
    - 876 times
  - activities::Scanner
    - 1455 times
  - activities::_PTPlayer
    - 1509 times -taking all with freq of addition over 450 only

11 Conclusion

[MAKE UP SOMETHING USING ABOVE>

Going over to apply both APEGAN and Cycle GAN on this dataset to see results

[ ]: