Secret Squirrels Team Project - Evaluating an Image Classifier

```
!git clone https://github.com/NIU-Data-Science/CNN-exercise.git
#reference:To import .py files in Colab, reference this post on Stackoverflow:https://stackoverflow.com/questions/4890
# NEEDS TO BE DONE EVERY TIME YOU REOPEN THE FILE
```

```
Cloning into 'CNN-exercise'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (9/9), done.
```

```
# Import package to use Google Drive API - not installed in Colab VM by default
# PyDrive is a wrapper library of google-api-python-client that simplifies many common Google Drive API tasks. For thi
from pydrive.auth import GoogleAuth
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
# Other necessary packages
from google.colab import auth
from oauth2client.client import GoogleCredentials
from tensorflow.keras.callbacks import ModelCheckpoint
```

```
# Follow prompt in the authorization process

auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
```

```
%cd /content/CNN-exercise
```

```
/content/CNN-exercise
```

```
%ls
```

```
classifier_function.py  CNN_trainer.py  output.txt    test_set/
CNN-exercise/           images.zip      __pycache__/  training_set/
CNN_model.h5            output.csv      README.md
```

```
import classifier_function
```

```
out_list = classifier_function.image_classifier('/content/CNN-exercise/test_set/', '/content/CNN-exercise/CNN_model.h5
```

```
/usr/local/lib/python3.6/dist-packages/PIL/Image.py:932: UserWarning: Palette images wit
  "Palette images with Transparency expressed in bytes should be "
```

```
out_list
```

```
len(out_list)
```

```
import pandas as pd
import math
```

```
#df= pd.read_csv('output.txt', header = None)
### data= pd.read_csv('output.txt', header = None, delimiter = ' ')  This version did a great job splitting them all bu
#df=df.transpose()
#df.head()
```

```
## REF:  https://stackoverflow.com/questions/33634142/pandas-how-to-delete-alternate-rows
#df2= df.iloc[::2]  # this gets rid of every second row that just had the data type of the previous row
#pd.set_option('display.max_colwidth', None)
#print(df2.iloc[200:220])
```

```
out_list_predict = {}
out_list_actual = {}
for i, key in enumerate(out_list):
    out_list_predict[key] = int(out_list[key][0][0])
    if '/not_tank/' in key:
        out_list_actual[key] = 0
    elif '/tank/' in key:
        out_list_actual[key] = 1
```

```
df_predicted = pd.DataFrame.from_dict(out_list_predict, orient='index', columns=['predicted'])
df_actual = pd.DataFrame.from_dict(out_list_actual, orient='index', columns=['actual'])
df_results = pd.concat([df_predicted, df_actual], axis=1)
df_results
```

```
TP=0
FP=0
TN=0
FN=0
```

```
for r, a in df_results.iterrows():
  if a['predicted']==0:
    if a['actual']==0:
      TN+=1
    elif a['actual']==1:
      FP+=1
  elif a['predicted']==1:
    if a['actual']==0:
      FN+=1
    elif a['actual']==1:
      TP+=1
```

```
print(TP)
print(FP)
print(TN)
print(FN)
```

```
MCC = ((TP*TN)-(FP*FN))/math.sqrt((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN))
print(MCC)
```

```
P0 = (TP+TN)/(TP+TN+FP+FN)
PY = ((TP+FN)/(TP+TN+FP+FN))*((TP+FP)/(TP+TN+FP+FN))
PN = ((FP+TN)/(TP+TN+FP+FN))*((FN+TN)/(TP+TN+FP+FN))
PN = ((FP+TN)/(TP+TN+FP+FN))*((FN+TN)/(TP+TN+FP+FN))
PE = PY+PN
Cohens_Kappa=(P0-PE)/(1-PE)
print(Cohens_Kappa)
```

On the MCC scale of -1 to 1, the given CNN is fairly good. It is a largely accurate predictive model for c
The Cohen's Kappa indicates "Substantial Agreement."