Liveness Detection

Assumptions Made: -

1. The major and only assumption here is images are ROI Face detector generated. The meaning of this is the images should be run under the ROI face detector. The dataset already consisted of *ROI extracted images,* so I didn't have to run that algorithm here. One can do it by using OpenCV2, Haar Cascade for face detection and extracting the face from images.

Tools Used: -

Python, Matplotlib, Seaborn, TensorFlow, Keras, Scikit-Learn, NumPy, Pandas, OpenCV

Dataset Used: -

The dataset used was NUAA Imposter Dataset, which consists of around *5108 real images and 7535 Fake or Spoofed images.*
http://parnec.nuaa.edu.cn/_upload/tpl/02/db/731/template731/pages/xtan/NUAAImposterDB_download.html

Colab Notebook: -

https://colab.research.google.com/drive/156JHj0pt6KweEcnEuLXNWnpvwjLMQ2jm?usp=sharing

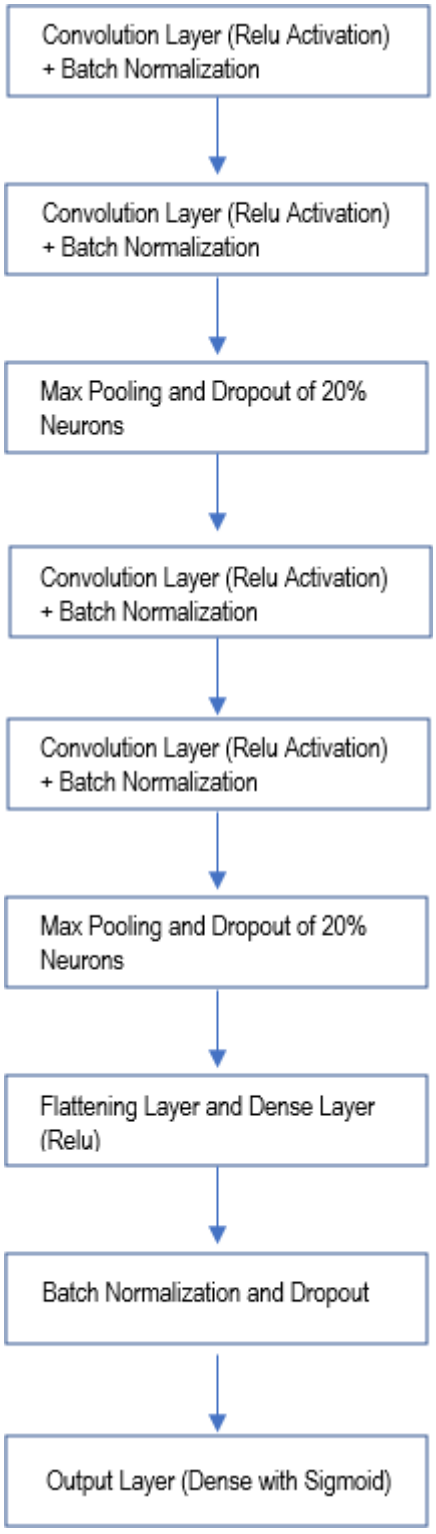
Performance Metrics: -

1. *Accuracy*: - Accuracy was used because we didn't have any skew or imbalance dataset.
2. *Precision*: - Precision tells us how accurate the model is predicting. Higher the precision more accurate the model.
3. *Recall*: - Recall tells how much positives were detected perfectly.
4. *F1 score*: - It is a mixture of both Recall and Precision (Harmonic mean).

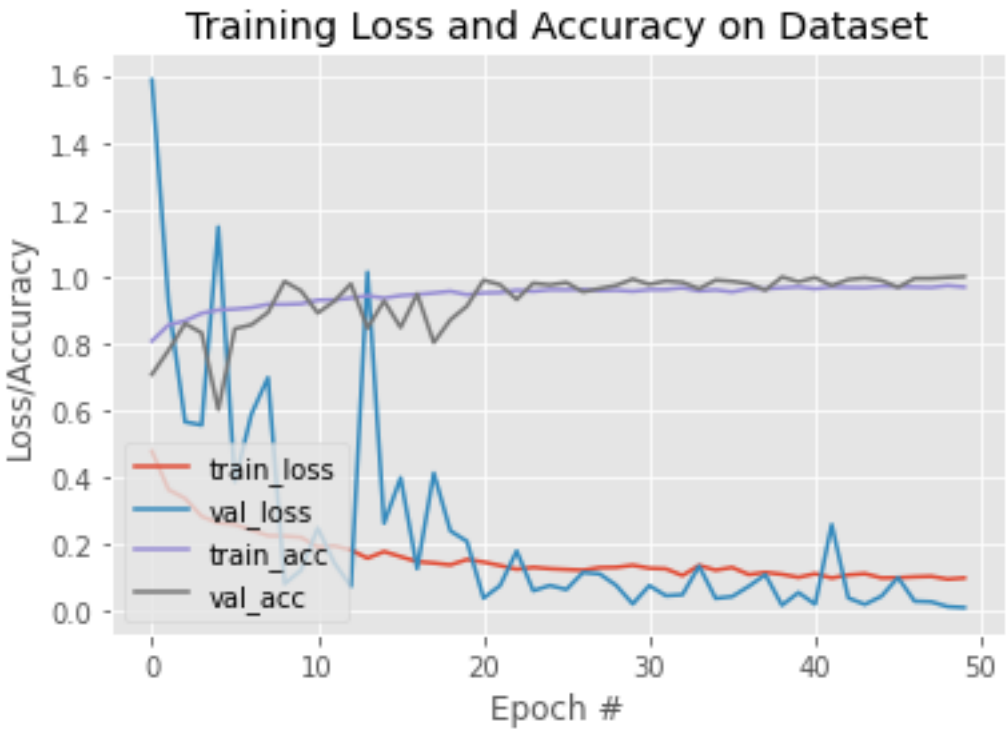
Also, Binary cross-entropy loss or log-loss was used to see how much our prediction varies from labels.

Other metrics which can be used in the classification-based programs are *ROC Curve and AUC score.*

Model Structure:-



Results and Testing on unknown data: -



	precision	recall	f1-score	support
0	1.00	1.00	1.00	73
1	1.00	1.00	1.00	87
accuracy			1.00	160
macro avg	1.00	1.00	1.00	160
weighted avg	1.00	1.00	1.00	160

Conclusion:-
Model accuracy was exceptional and model worked well on test images as well.

Restriction:-
Due to lack of variety in images, model doesn't work well on Non East Asian Images. So for handling this issue, we can create a big dataset for the future projects on the Same topic.

Reference: -

1. https://towardsdatascience.com/real-time-face-liveness-detection-with-python-keras-and-opencv-c35dc70dafd3
2. https://link.springer.com/referenceworkentry/10.1007%2F978-3-642-27733-7_9067-2