Suspicious Presentation Detection in Face Recognition

ISyE 6740 – Summer 2020

Final Report

Jacob Lee
jacobjlee@gatech.edu

Mehmet Gumus
mgumus3@gatech.edu

Matthew Mitchell
mmitchell95@gatech.edu

Team Id - 171

Abstract*—*Face recognition algorithms are being used to verify the identities of individuals via the internet on a global scale, with an example being the Georgia Institute of Technology's remote proctoring software. This leads to the opportunity for subversion through the presentation of images or mobile devices in place of real faces. This project uses the OULU face attack dataset and a number of computational data analytics methods to explore the effectiveness of algorithmic detection of such attacks.

Table of Contents

Problem Statement

Since the inception of biometric recognition algorithms, there have been countless actors, both academic and malicious, who have tested the boundaries of this technology with the objective of compromising it. One such method is noted as a Biometric Presentation Attack *(BioPAth, B., 2020)*, where this attack is commonly used in the context of face biometrics. More specifically, this attack involves an actor being falsely identified as another individual, where said actor has presented a picture or video of that individual to a system. Many notable organizations, including the Georgia Institute of Technology (Georgia Tech), have adopted technologies that detect this form of attack.

In Georgia Tech's case, it has the remote proctoring software known as Proctortrack, which is used to identify any cases of cheating during when a student conducts an exam remotely. In this scenario, the software records a web-camera feed of the student conducting an exam, which is then manually reviewed for attacks and other forms of cheating. Since the student is in an uncontrolled environment and not physically present, this is a necessary precaution and mitigates many vulnerabilities such as the Biometric Presentation Attack.

This method of manually resolving cases is not the only means of identifying this form of attack, where data science models have been developed to automate the process of flagging suspicious activity. These models come in many different forms; however, the most widely used method is through creating Convolution Neural Networks (CNN), which have only briefly been explored in this class. Therefore, the objective of this project is to compare the models used within this course and CNNs to determine if any of these models could be a suitable for this problem space and for real world application.

Data Source

The dataset used for this analysis is known as OULU-NPU (Boulkenafet et al., 2017), which is a mobile face presentation attack database that provides real-world variations of attacks with the intention of developing classification algorithms. The raw dataset includes a total of 4,950 **videos**, where 990 are genuine and 3,960 are non-genuine presentations. This means that for each genuine sample (video), there are 4 equivalent non-genuine samples, where these non-genuine samples each represent a specific type of presentation attack. These non-genuine samples represent the following types of attacks:

1. Image of the individual printed and shown on a piece of paper
2. Image of the individual print and shown on a piece of paper, where a different printer was used from (1)
3. Video of the individual shown on an electronic device
4. Video of the individual shown on an electronic device, where a different electronic device was used from (3) (note that there were six mobile devices used over three different sessions for collecting this data)

The entire composition of this dataset is shown in Table 1 and an example of a single case (1 genuine, 4 non-genuine) can be seen in Figure 1.

Table 1: Dataset composition

	Users	**Real access**	**Print attacks**	**Video attacks**	**Total**
Training	20	360	720	720	1800
Development	15	270	540	540	1350
Test	20	360	720	720	1800

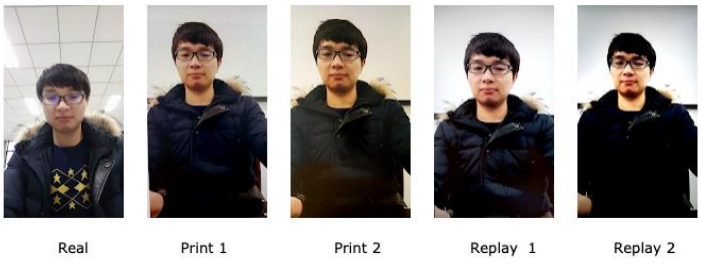

Figure 1: sample of video data for one case

For the purposes of this project, we will be looking at analysing images as opposed to videos. This means that we have had to convert the video samples provided into images, which will be achieved by extracting frames from each video. Since there are 4 times as many non-genuine samples as there are genuine (refer to Table 1), we decided to over sample the genuine samples to result in an even number of genuine and non-genuine samples. We also cropped the images to only have the face of the individual in frame and normalised these face images to be 150 x 150 pixels rather than the original 1920 x 1080 to improve model accuracy and training performance.

The exact process on converting the videos to usable images is shown below:

- For-each genuine video sample:
    - Extract 4 frames from video at random time intervals
    - For each frame:
        - Find the face in frame using a face detection algorithm (Haar Cascade)
        - Crop the face out of the frame
        - Normalize size of face image to be 150 x 150 pixels
        - Save face image to disk
- For-each non-genuine video sample:
    - Extract 1 frame
    - Find face in frame using a face detection algorithm
    - Crop face out of the frame
    - Normalize size of face image to be 150 x 150 pixels
    - Save face image to disk

This has resulted in 3,960 genuine images and 3,960 non-genuine images, which each resembled the images shown in Figure 2. The breakdown of the Train, Development and Test sets were unchanged.

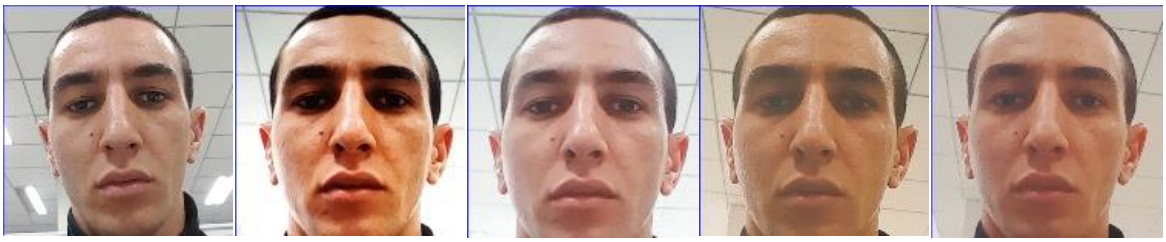

Figure 2: Example of cropped images (Genuine, Print 1 Non-Genuine, Print 2 Non-Genuine, Replay 1 Non-Genuine, Replay 2 Non-Genuine)

Methodology

As mentioned previously, the purpose of this project is to compare the effectiveness of machine learning techniques learnt in this course and convolutional neural networks. Therefore, the project will compare two separate approaches for achieving image classification:

1. Process image distortion metrics and then complete classification on this tabular data using a variety of techniques taught in this course.
2. Complete minimal image preprocessing and perform classification using a CNN

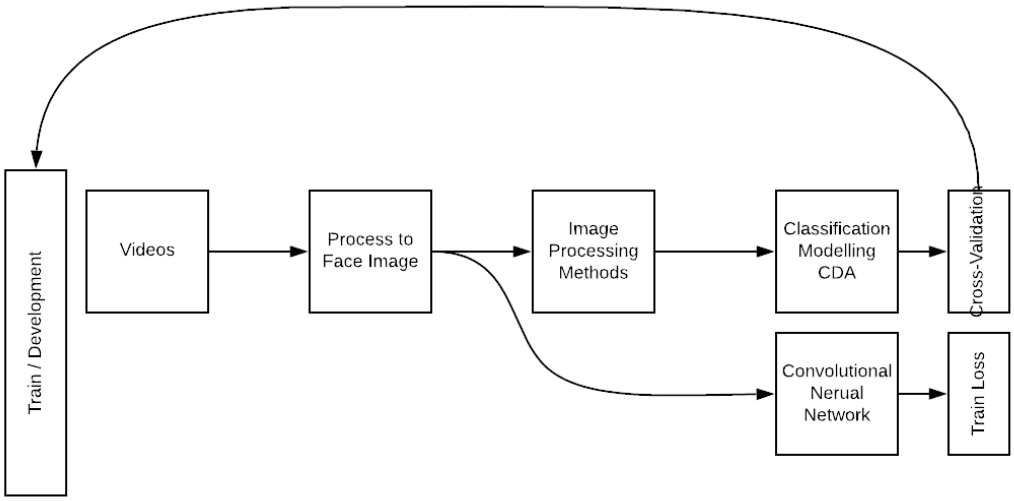

Figure 3: Conceptual diagram of both approaches for training models

The above figure shows how the traditional ML models (approach 1) and CNN (approach 2) will be trained. For approach 1 models, they will require creating a derived dataset that contains specific image quality metrics, rather than using the pixel values of images directly. This is discussed later in this section on why this is done and what metrics are used. No such feature engineering is required for approach 2 (CNN) and the normalised face images are used directly to train the network.

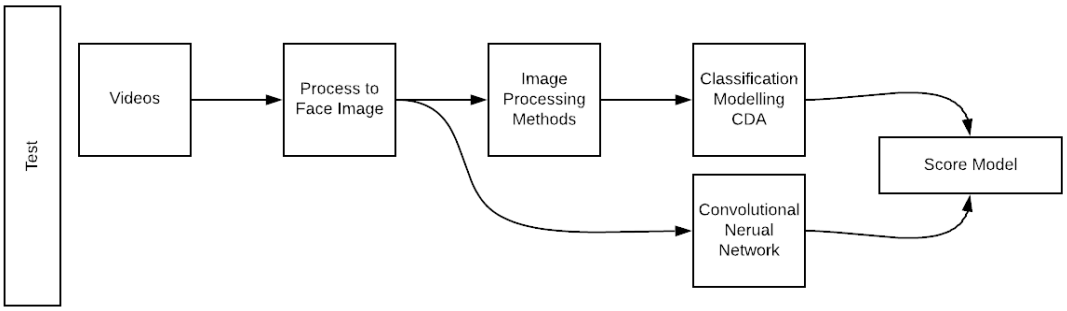

Figure 4: Conceptual diagram of both approaches for testing models

The above figure shows the testing phase that will be conducted for each approach, where this phase will begin after all models are created in the training phase. The scoring and metrics from this phase will then be presented for comparison in the Results section of this paper.

Approach 1: Image Feature Engineering and Various Machine Learning Algorithms

The first method will explore the set of classification algorithms taught in the course (ISyE 6740: Computational Data Analytics), which includes:

- Logistic regression
- K-nearest neighbor
- Naïve Bayes
- Support Vector Machines (SVM)
- Neural Networks
- Adaboost
- Random Forest

To evaluate these algorithms, the following steps will be taken for each algorithm:

1. Train the algorithm over the Train and Dev datasets to create a model
    a. In some cases, additional techniques such as below may be used to boost performance:
        i. Principle Component analysis (PCA)
        ii. Cross Validation – K-folds
2. Compute all metrics in Table 2
    a. The feature engineering undertaken was to extract image distortion metrics from the processed and normalized data as specified in Galbally et al *(2014)*. The intuition presented in this research is that presentation attack images have different levels of image distortion compared to Genuine images. Not all metrics in Galbally et al *(2014)* were able to be provided due to computational limitations however the completed metrics are described in Table 2.
3. Generate ROC graphs using metrics (from step 2)
4. Generate confusion matrix by running the model (from step 1) over the Train dataset

Table 2: Performance metrics

Type	*Acronym*	*Name*	*Description*
FR	MSE	Mean Square Error	$MSE(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{i,j} - \hat{I}_{i,j})^2$
FR	PSNR	Peak Signal to Noise Ratio	$PSNR(I,\hat{I}) = 10\log\left(\frac{\max(I^2)}{MSE(I,\hat{I})}\right)$
FR	SNR	Signal to Noise Ratio	$SNR(I,\hat{I}) = 10\log\left(\frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{i,j})^2}{N*M*MSE(I,\hat{I})}\right)$
FR	SC	Structural Content	$SC(I,\hat{I}) = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{i,j})^2}{\sum_{i=1}^{N}\sum_{j=1}^{M}(\hat{I}_{i,j})^2}$
FR	MD	Maximum Difference	$MD(I,\hat{I}) = \max\ \lvert I_{i,j} - \hat{I}_{i,j}\rvert$
FR	AD	Average Difference	$AD(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(I_{i,j} - \hat{I}_{i,j})$
FR	RAMD	R-Averaged MD	$RAMD(I,\hat{I}) = \frac{1}{R}\sum_{r=1}^{R}\max_{r}\lvert I_{i,j} - \hat{I}_{i,j}\rvert$
FR	LMSE	Laplacian MSE	$LMSE(I,\hat{I}) = \frac{\sum_{i=1}^{N-1}\sum_{j=1}^{M-1}(h(I_{i,j}) - h(\hat{I}_{i,j}))^2}{\sum_{i=1}^{N}\sum_{j=1}^{M}h(I_{i,j})^2}$
FR	MAS	Mean Angle Similarity	$MAS(I,\hat{I}) = 1 - \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(\alpha_{i,j})$
FR	MAMS	Mean Angle Magnitude Similarity	$MAMS(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(1 - [1-\alpha_{i,j}][1 - \frac{\lVert I_{i,j} - \hat{I}_{i,j}\rVert}{255}]\right)$
FR	SME	Spectral Magnitude Error	$SME(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}(\lvert F_{i,j}\rvert - \lvert\hat{F}_{i,j}\rvert)^2$
FR	GME	Gradient Magnitude Error	$GME(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\left(\lvert G_{i,j}\rvert - \lvert\hat{G}_{i,j}\rvert\right)^2$
FR	GPE	Gradient Phase Error	$GPE(I,\hat{I}) = \frac{1}{NM}\sum_{i=1}^{N}\sum_{j=1}^{M}\lvert\arg(G_{i,j}) - \arg(\hat{G}_{i,j})\rvert^2$
FR	SSIM	Structural Similarity Index	*See* Galbally et al (2014)[36]

FR	VIF	Visual Information Fidelity	*See* Galbally et al (2014)[38]
	HLFI	High-Low Frequency Index	$HLFI(I) = \frac{\sum_{i=1}^{i_l}\sum_{j=1}^{j_l}\lvert F_{i,j}\rvert - \sum_{i=i_h+1}^{N}\sum_{j=j_h+1}^{M}\lvert F_{i,j}\rvert}{\sum_{i=1}^{N}\sum_{j=1}^{M}\lvert F_{i,j}\rvert}$

Prior to modelling the derived metrics in the table above were analysed to assess their relevance in addressing this papers problem statement, a brief summary of the exploratory data analysis is attached as an annex to this paper.

Approach 2: Convolutional Neural Network

The second method extends the knowledge body of this class by exploring convolutional neural networks as an advanced machine learning technique. This technique has proven to be very effective for a number of challenging computer vision tasks[1].

This will be assessed by performing the following steps:

1. Define and convolutional neural network architecture as specified in the paper [1].
2. Train the algorithm over the Train and Dev datasets to create a model
3. Provide predictions for the test set using the trained model
4. Report on model accuracy, ROC, ROC AUC to compare against the feature engineering approach

A convolutional neural net is a similar technique to the neural networks completed in this course, with the inclusion of the convolution operation. Rather than the fully connected layer worked on, a convolutional layer has shared parameters that process the image iteratively, taking advantage of the hierarchical nature of image data and allowing for a reduction in the required number of parameters in the algorithm. As a result, the kernels activate when it can detect some type of feature as some spatial position in the input (Géron, *A*, 2019).

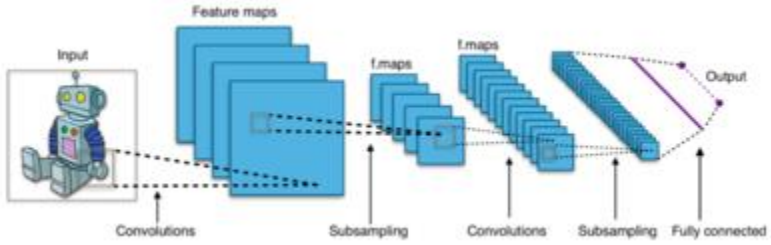

Figure 5: A typical neural network architecture (Nelson, D, 2020)

The model architecture selected for this paper is specified in table 3 And was tuned using the train and development sets starting from a baseline architecture specified in Sermanet et al, 2013.

Comparison

Once both approaches are completed and all metrics / graphs are obtained, the results from these approaches will be analyzed and compared to determine the viability of the models taught in this course against CNNs for this particular task. Once this is determined, it will then be discussed whether the optimal model is considered an effective model for real-world application.

The comparison will be completed with Receiver-Operator Curve as the primary visualizations and will be compared on the following metrics:

- Classification accuracy
- ROC areas under curve
- Precision-Recall area under curve
- Confusion matrix

These metrics and visualizations will be created using only the data from the test set, which is an unbiased set of data that does not include any images used when training the models. This means that any models that are overfitted to the train data are not expected to perform well on this dataset and will also demonstrate accurate metrics for comparison purposes.

Results

The results from following the methodologies outlined previously are shown below and will be utilized for comparative purposes.

Approach 1

Each algorithm was trained using cross validation on the entire set of the image distortion metrics. The accuracy for each holdout set for 5-fold cross validation is visualised in Figure 6 and Table 3 shows the numerical figures to support this visualisation.

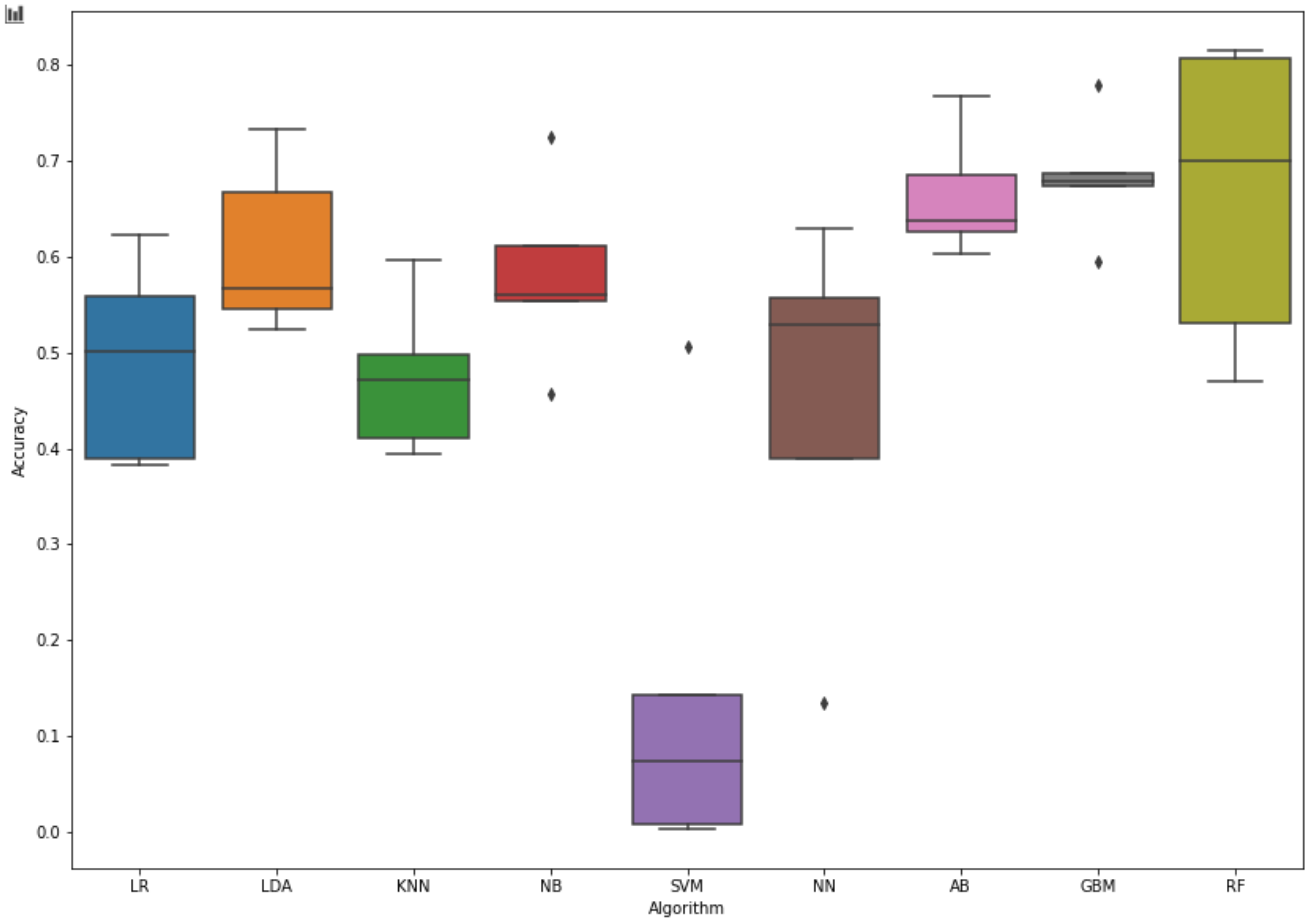

Figure 6: Boxplot for accuracy of each holdout in cross-valdiation

It can be observed in both of the figures that the random forest model (denoted as RF) had the highest mean accuracy on the hold out sets, while adaptive boosting (AB) performed nearly as well with a lower variance in accuracy results.

Table 3: Approach 1 accuracy metrics

Algorithm	*Mean Accuracy on Holdout*	*Standard Deviation of Accuracy on Holdout*
Logistic Regression (LR)	0.490908	0.093605
Linear Discriminant Analysis (LDA)	0.607465	0.079716
K-Nearest Neighbour (KNN)	0.474192	0.072143
Naïve Bayes (NB)	0.581376	0.087591
Support Vector Machines (SVM)	0.146811	0.186716
Fully Connected Neural Network (NN)	0.447907	0.175029
Adaptive Boosting (AB)	0.663868	0.058265
Gradient Boosting Machine (GBM)	0.682455	0.058510
Random Forest (RF)	0.664262	0.141546

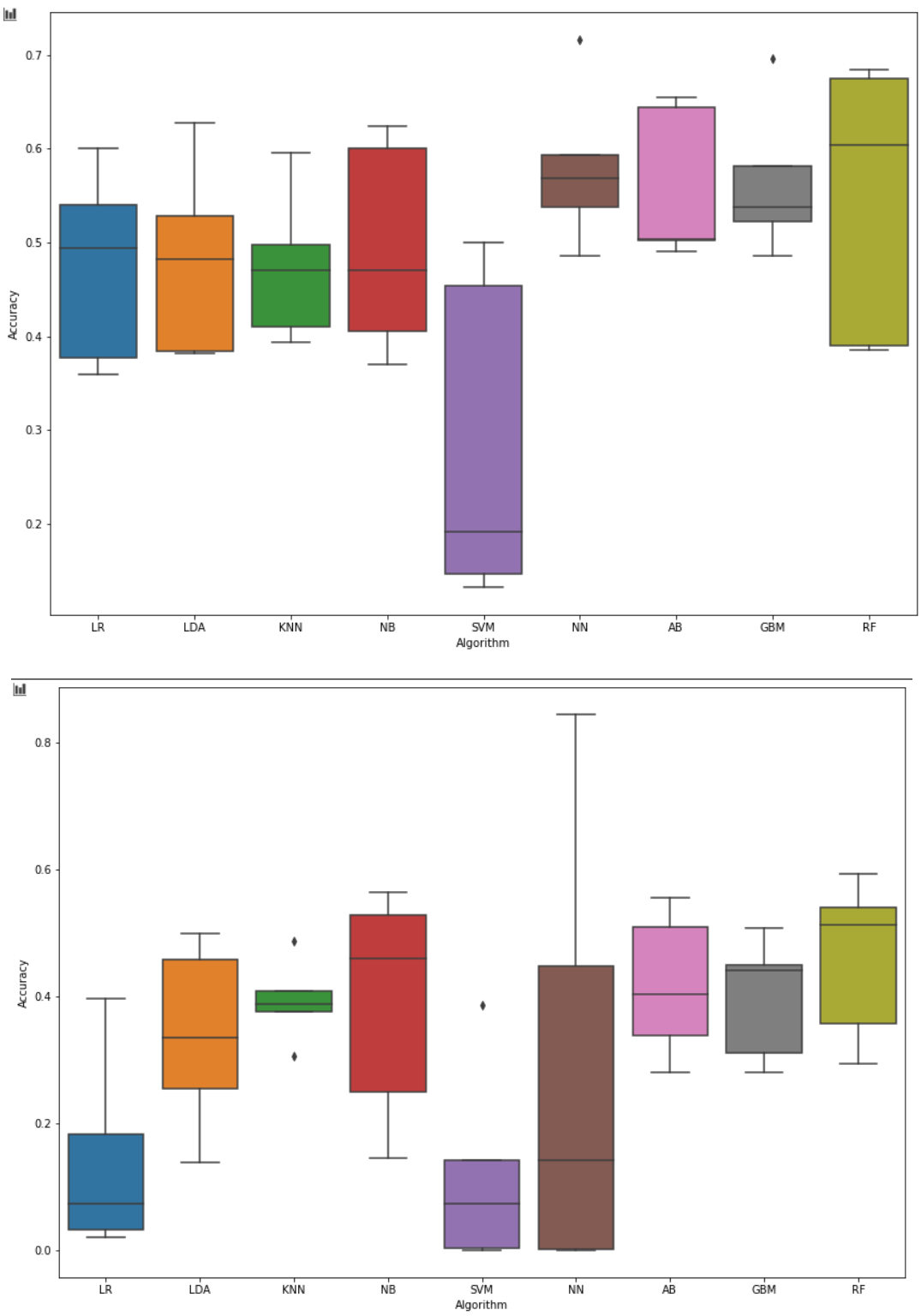

Figure 7: CV results on PCA (Top) and Feature Elimination (Bottom)

Two forms of dimensionality reduction were explored for creating each type of model to improve performance. The first was feature elimination, where we kept only the 5 most informative features as found by the Galbally et al (2014) paper , which is shown in the top image of Figure 7. The second approach was PCA, where we reduced the dimensions to top 5 principal components. In each case of dimensionality reduction, the impact on performance was significant compared to using the full 18 features in the dataset. This can be seen by comparing Figure 6 and Figure 7, where dimensionality reduction significantly reduced performance of all models.

Based on these figures and assessment, we stopped exploring all but the top two models, and began the process of parameter tuning on said models. These are the Random Forest and Adaptive Boosting models, where they have the following traits that made them the top two candidates to compete with CNNs:

- They were highly accurate in the cross-validation activity
- They were a core part of the course curriculum (as opposed to Gradient Boosted Machines)

The dimensionality reduction techniques were not applied for both models going forward as their performance decreased using these techniques.

For AdaBoost the hyperparameter for the number of decision stumps was explored between 50 and 950. The optimal parameter was found to be 900 (as shown in Figure 9) with a holdout set accuracy of 67.1% with a standard deviation of 5.4%.

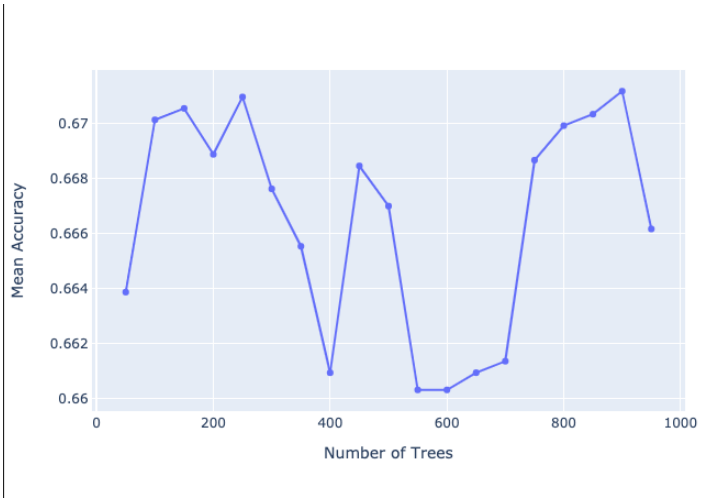

Figure 8: Random forest parameter tuning

Random forest was tuned for both the number of trees and the depth of the trees. The grid was built for the maximum depth of the tree between 10 and 190 in intervals of 10, while the number of trees was between 50 and 450 in intervals of 50 (as shown in Figure 9). The optimal parameters were found to be a max depth of 10 with 200 trees at 66.7% accuracy and a standard deviation 7.2%.

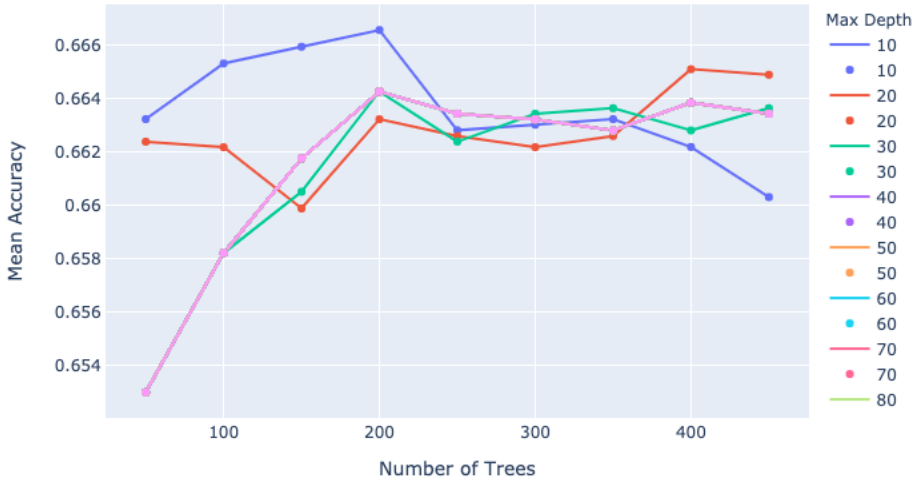

Figure 9: Accuracy vs Number of trees plot for Random Forest

By comparing both the Adaboost and Random forest model's performance (shown in Table 4), we can see that Adaboost performs slightly better on the holdout fold during cross validation. A final classifier was therefore built on the entire train and development set using Adaptive Boosting as the model of choice, for the purpose of model scoring in the comparison section of this report.

Table 4: Algorithm Performance Comparison

Algorithm	*Mean Holdout Fold Accuracy*
Adaboost	67.1%
Random Forest	66.7%

Approach 2

For approach 2, the CNN was implemented using the following architecture using TensorFlow:

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	150 x 150 x 16	448
max_pooling2d (MaxPooling2D)	75 x 75 x 16	0
dropout (Dropout)	75 x 75 x 16	0
conv2d_1 (Conv2D)	75 x 75 x 32	4640
max_pooling2d_1 (Max Pooling 2D)	37 x 37 x 32	0
conv2d_2 (Conv2D)	37 x 37 x 64	18496
max_pooling2d_2 (MaxPooling2	18 x 18 x 64	0
conv2d_3 (Conv2D)	18 x 18 x 128	73856
max_pooling2d_2 (MaxPooling2	9 x 9 x 128	0
dropout_1 (Dropout)	9 x 9 x 128	0
flatten (Flatten)	10368	0
dense (Dense)	256	2654464
dense_1 (Dense)	1	257
Total params: 2,752,161 Trainable params: 2,752,161 Non-trainable params: 0		

This model has four convolution blocks with a max pool layer in each of them. There's a fully connected layer with 256 units on top of it that is activated by a relu activation function. Dropout was also added to the model to reduce overfitting. It is applied to the first and last max pool layers in this model, where it randomly eliminates 20% of the output units from these layers in each training epoch. Then the model was compiled with *ADAM* optimizer and *binary cross entropy loss* function (as shown below) as this is a binary classification problem.

Overfitting is a potential problem with Neural networks and can significantly affect performance. This problem arises in the case of using a small training set. To overcome this issue, the dataset was augmented where the training set was expanded by generating new images from existing images through random transformations, including horizontal flip, rotation, and zooming. This technique helps to create variations of the images that enable the model to generalize better. This was achieved by utilizing the *ImageDataGenerator* function in Keras as and can be observed in the supporting code for this submission. Note that the Test set that we will use later to compare models won't have data augmentation applied since we want to see the performance of the model on the unmodified images.

Once we established the architecture, techniques and datasets for creating a performant CNN, we began the training exercise using train and development set (with augmentations) for creating the model. The model was developed with the parameters *batch size = 128* and *epochs = 40*, where the performance throughout this training process is shown below:

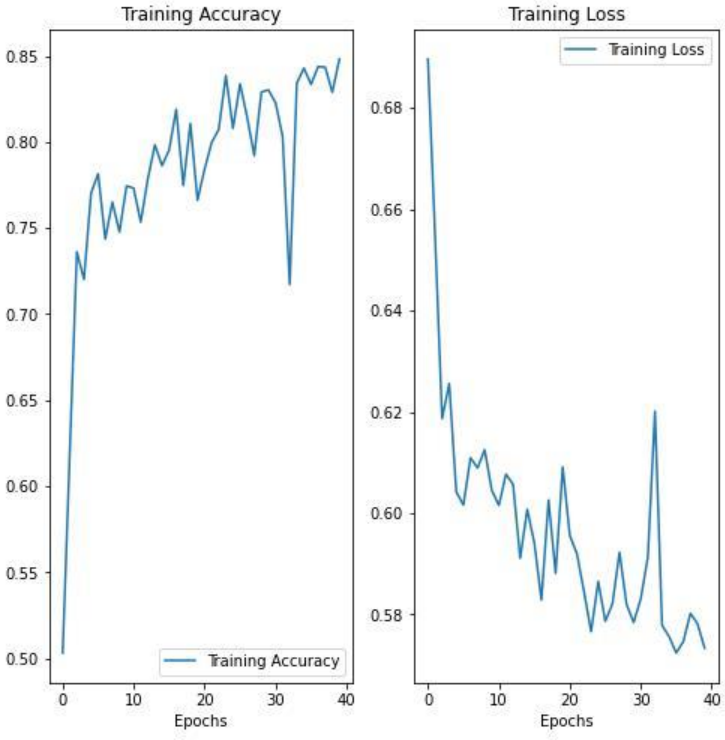

Figure 10: Accuracy and loss over epochs for CNN training

We observe from the learning curves above that the model is capable of learning as training accuracy keeps increasing and training loss keeps decreasing. After 40 epochs modelling training was halted, the model was used to provide predictions on the test set and are discussed in the following sections.

Comparison

We can compare the performance of the CNN model and the AdaBoost model by utilising the test set, which is independent of the training and has not been utilised until now for comparison activities. By making predictions at different threshold values, we can calculate the True Positive Rate (TPR) and False Positive Rate (FPR) at these different values to produce the following plot:

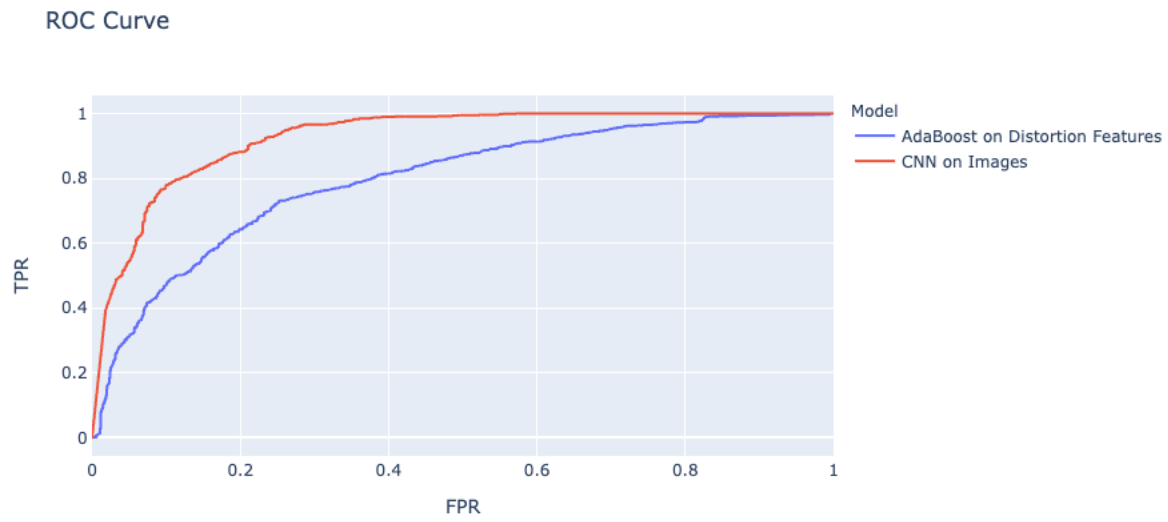

Figure 11: ROC curves for AdaBoost and CNN

The above ROC curve visualises the performance of both models, where there is a clear performance increase for the CNN at all levels. Drilling down to the point at a naïve 0.5 threshold, the confusion matrix can be observed:

Table 5: Confusion Matrix for CNN and AdaBoost

	CNN		*AdaBoost on Distortion Features*	
	True	*False*	*True*	*False*
True	919	358	**991**	**286**
False	**49**	**1207**	408	848

The CNN performs better in correct determinations of presentation attacks, however interestingly the number of false negatives is higher in the CNN. The is reflected in the recall metric in the performance metrics below:

Table 6: Performance Metrics for CNN and AdaBoost

	CNN			*AdaBoost on Distortion Features*		
	Precision	*Recall*	*F1 Score*	*Precision*	*Recall*	*F1 Score*
Presentation Attack	0.95	0.72	0.82	0.71	0.78	0.74
Genuine *Presentation*	0.77	0.96	0.86	0.75	0.68	0.71

Given this performance of higher recall/F1 score for AdaBoost in the presentation attack case this algorithm may be preferable where user facilitation is prioritised, that is to say where a case falsely identified as a presentation attack is considered more costly than an attack going undetected.

Overall, the CNN provides higher performance summarized by the AUC, and the classification accuracy overall is 11.3% higher:

Table 7: Additional Performance metrics for CNN and AdaBoost

Algorithm	*ROC AUC*	*Classification Accuracy*
CNN	0.926	83.9%
AdaBoost on Distortion Features	0.796	72.6%

Comparing the two classifiers for scoring evidences that the error types are different and so for different use cases the AdaBoost algorithm may be preferable; however, overall, the CNN is the higher performing algorithm.

Discussion

It was found that overall the convolution neural network provided a stronger classifier than all of the other models covered in this course. While the traditional machine learning methods proved to have predictive power, the CNN was 11.3% more accuracy with improved area under curve for the ROC plot.

This shows that the more sophisticated model designed for the specific problem space is able to identify more complex patterns and requires a lower level of feature engineering. The applicability to the real world would require further test sets, however there is strong evidence that the characteristics of presentation attacks can be identified using computational data analytics.

This paper provides areas of further research that may be pursued in improving how this computer vision problem is addressed:

- Sequence to classification model using recurrent neural networks on the raw video data
- Other biometric presentation attacks such as for adversarial machine learning, deep fake detection
- While significant effort was allocated to feature engineering and model tuning the traditional machine learning methods, minimal tuning and data augmentation was completed on the CNN. This was completed to evidence that the modelling approaches are inherently different, however further optimizations could be made to the CNN model. This may include the testing of established architectures such LeNet, VGG, ResNet or Inception models (Huang et al, 2017)(Szegedy et al, 2015).

Conclusion

This paper explored methods for classification of face biometric images as fraudulent presentation attacks or genuine images. This is a challenging computer vision problem and the computational data analytic showed promising results in addressing this real-world concern. The application of these models in production could lead to higher integrity identity processes in a number of verticals including education (student exams), online payments, welfare amongst others.

This paper extended established methods previously published by:

1. Using a broader range of machine learning classifiers to determine presentation attacks
2. Exploring dimensionality reduction techniques on the hand engineered features to deal with co-linearity
3. Using a convolutional neural network on normalized face region images

Of all the models, the CNN performed most optimally and allowed for higher classification performance on an unseen test with reduced levels of feature engineering. This method provides reduced inference into the features driving classification decisions and the errors are skewed towards producing false negative errors, as such there may be some circumstances where the hand engineered features are preferable. Further work may be completed in the future to determine if these approaches generalise outside of this dataset, this would allow for use in commercial applications i.e. Proctortrack.

Annex

Exploratory data analysis was completed on the derived image metrics.

There was a relative high level of correlation in some potential features and thus the decision was made to include dimensionality reduction techniques in the paper:

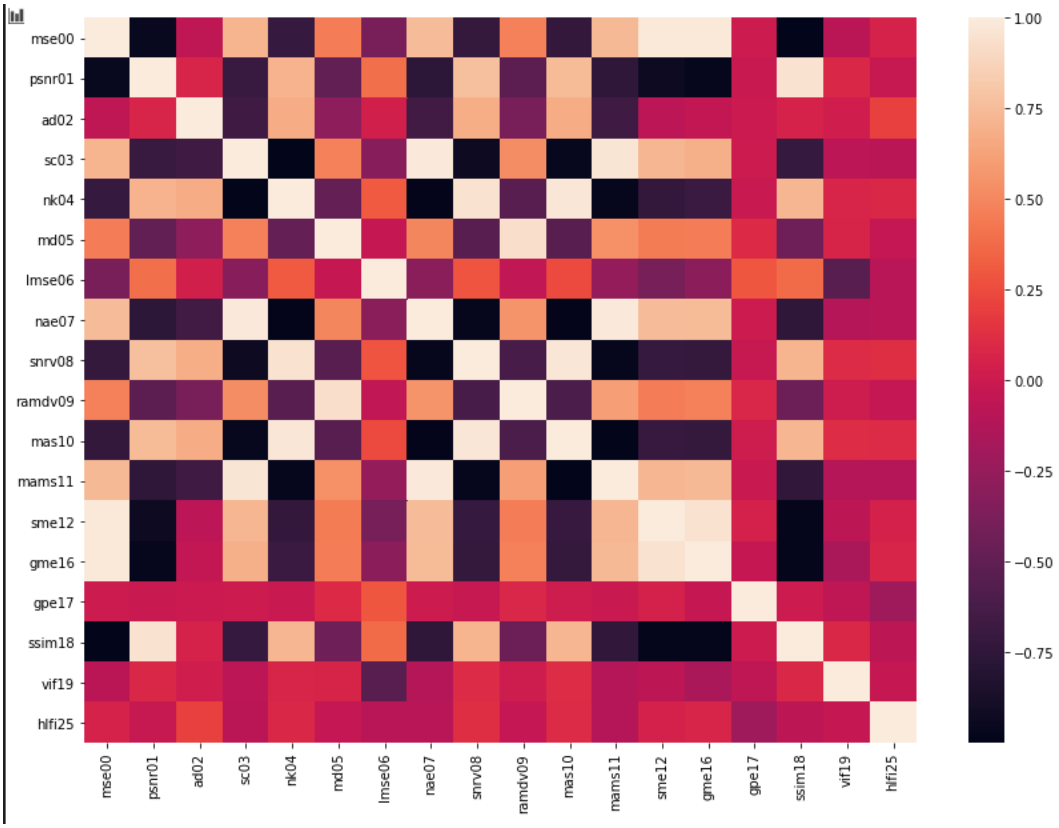

Figure 12: Correlation heatmap of the derived image metrics

The metrics when plotted below and color map is according to the response variable indicate that no single measure would serve as a very strong predictor of the genuine nature of the image, however many measures provide partial information in regards to the nature of the image.

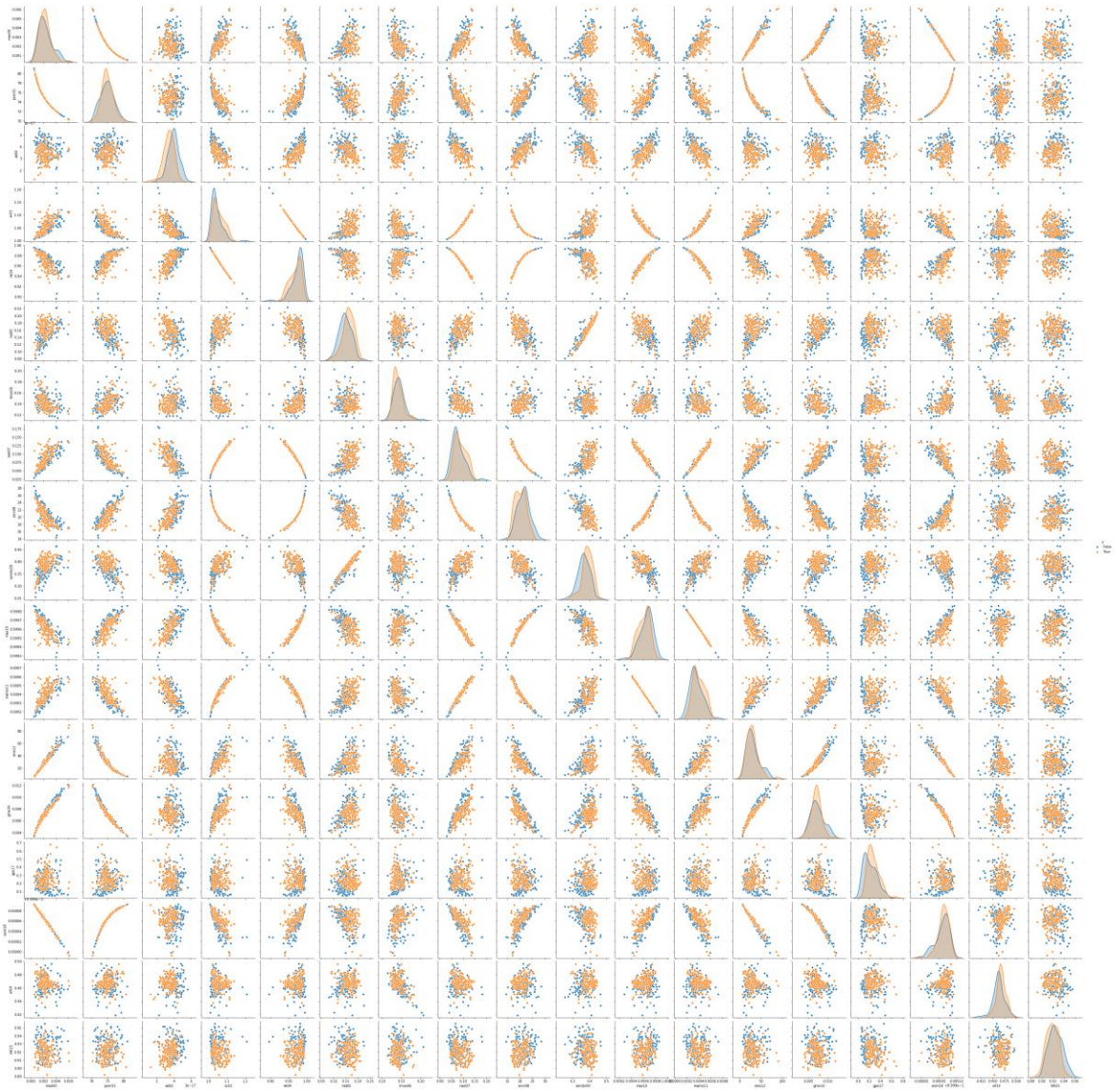

Figure 13: Pair Plot of derived image metrics

References

1. BioPAth, B., 2020. Pattern Recognition Letters. [online] Journals.elsevier.com. Available at: <https://www.journals.elsevier.com/pattern-recognition-letters/call-for-papers/biometric-presentation-attacks> [Accessed 25 July 2020].
2. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y., 2013. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.

3. Galbally, J., Marcel, S. and Fierrez, J., 2014. Image quality assessment for fake biometric detection: Application to iris, fingerprint, and face recognition. *IEEE transactions on image processing*, *23*(2), pp.710-724.
4. Géron, Aurélien (2019). Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow. Sebastopol, CA: O'Reilly Media. *ISBN 9780226484648., pp. 448*
5. Nelson, D., 2020. What Are Convolutional Neural Networks?. [online] Unite.AI. Available at: <https://www.unite.ai/what-are-convolutional-neural-networks/> [Accessed 25 July 2020].
6. Boulkenafet, Z., Komulainen, J., Lei, L., Feng, X. and Hadid, A., 2017. OULU-NPU: A Mobile Face Presentation Attack Database With Real-World Variations / IEEE International Conference On Automatic Face And Gesture Recognition. [video] Available at: <http://www.ee.oulu.fi/research/bmetric/Oulu_NPU/> [Accessed 25 July 2020].
7. Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q., 2017. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).
8. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).