Facial Liveness Testing For The Web

Ryan Collins
Department of Computer Science, Durham University

What are Facial Liveness Tests?

Facial liveness tests are methods of detecting whether the person in front of the camera is, in fact, a real person. Common methods of fooling existing recognition systems include displaying a printed piece of paper to a camera. Alternative methods involving high-resolution screens, or video playback are highly successful variants of these spoofing methods. As the facial liveness testing field becomes more innovative and better and detecting liveness, facial recognition systems can become more secure. While spoofing attacks might appear realistic, there are numerous methods of detecting them. On a single image, one can consider the image quality, as spoofing attacks will contain fewer high frequencies. Furthermore, texture differences and facial structure differences can also be detected. With videos, movement between frames can be used to determine the depth or analyze movement.

Differences between a spoofing attack (using the paper-based attack method), and a real input, can be seen in the images below.

To the untrained eye, the below example is obvious. However, in cases which only show the face, this classification can become much more difficult.

Figure: A real image of a person. Image from the NUAA dataset.

Figure: A spoofed image with a paper based replay attack. Image from the NUAA dataset.

Why is my work important?

Facial recognition systems are only as good as the liveness systems that they employ. Better liveness systems lead to more secure facial recognition systems. Throughout this project, the aim was to develop liveness tests that can work with standard hardware available and allow use in real time. By doing this, future recognition systems will become more secure, while also not being too burdensome on the users.

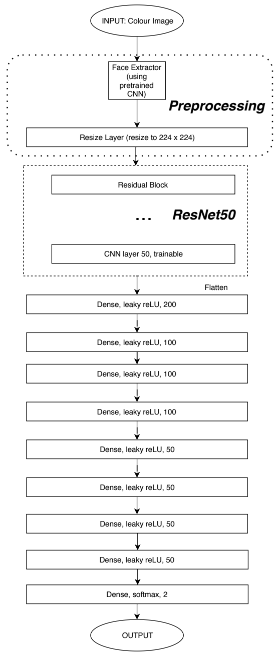
Consolidation Layer - A method to fuse liveness test results together

Using a similar approach as the Image Quality Assessment method, bringing together the results of several liveness tests could yield better results. As such, analysis was undertaken on the results of the two liveness metrics produced.

It was found that the liveness test results yielded a linearly separable pattern. This means that on a graph of one liveness test against another, a single line can be used to divide the two. While the entire dataset isn't fully linearly separable, there is a component. However, since this only works for 2 tests, it was decided to use Linear Discriminant Analysis to provide our liveness test fusion functionality.

The result yielded reasonable results, gaining 75% accuracy over the Replay-Attack test dataset, although more work is still necessary to determine the benefits gained over different datasets and a larger variety of liveness tests, since two isn't enough.

Method 1: CNN based liveness detector



Recently, Convolutional Neural Networks have played a major part in image classification. Residual Networks have performed very well on the ImageNet dataset, classifying types of objects based on a 2D image. Since liveness detection is an image classification problem, this can be adapted to predict spoofing, specifically within a facial region.

The facial area was obtained using a pre-trained CNN, which returned a bounding box containing the face. The 224x224 size image obtained was used to classify the liveness.

Training a 50 layer ResNet would take a very long time, and therefore a pre-trained model was used to improve results and reduce training time. The first 49 layers were left frozen, while the last layer was allowed to be modified. Attached to the ResNet 50 model was a feed-forward neural network, used to obtain a metric of 'realness': how real the subject in the image is.

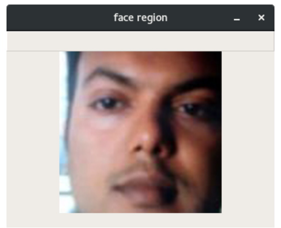


Figure: The result of the preprocessing step, yielding a cropped face.

Accuracy yielded was adequate, with a 71% accuracy on the Replay Attack test dataset. However, the model had a very low false positive rate, with most errors being caused by false negatives (which is annoying for a user but not a security threat).

Method 2: Whole Image Quality Assessment

This method considers the whole image quality, as a spoofed image is likely to have less details than a non-spoofed image.

This liveness test uses 24 different methods of measuring image quality. Some metrics are full reference, which require the use of a Gaussian Blurred version of the image to compare to, which analyze the overall pixel based error, image noise, and frequency ranges. Other quality metrics use pretrained classifiers to analyze quality.

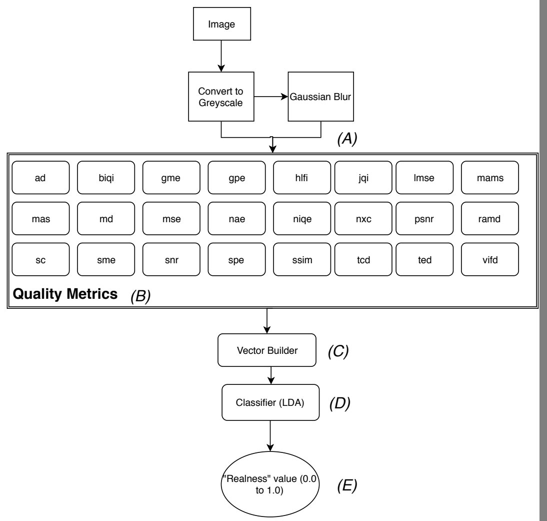
The results of these quality metrics are then fed through a classifier. The classifier used was Linear Discriminant Analysis, with an eigenvector solver. Below, a diagram of how this system works can be seen.

Overall, this method led to 87% accuracy when classifying liveness on the Replay-Attack test dataset.

A high number of true negatives and positives were yielded, but when an error occurred, the classifier was more likely to class spoofed images as real, compared to the other way around, which isn't ideal for security conscious applications.

The current method does have drawbacks, as the training data led to an expected resolution, and with images of different resolutions the prediction is often erroneous. One solution is to include resolution as it's own image quality metric within the classifier. For the demo, the solution was to resize an input image down to (640x480), which solves the problem temporarily, but future improvements should be able to predict liveness without this resize step.

Furthermore, while the time to predict a single image is fairly fast, it could be improved further by migrating to a sklearn based classifier for the BIQI image quality metric. Currently, a libsvm based classifier is used, which is called by accessing the file system. This is very slow.

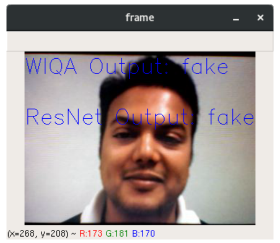


Method 3: 2D reconstruction Mask Attack Detection Liveness Test

This used a pretrained 2D to 3D reconstruction model to produce a 3D model of a face. This 3D model was then fed through a VoxNet based classifier using several convolutional layers with a large number of filters. While an interesting idea, this method required far too much time to reconstruct a face, and required very large amounts of memory to predict. Therefore, training was impossible with the resources available. As such, this model wasn't suitable for a real time approach.

Real-time Liveness System - a proof of concept

The liveness methods above were brought together into a real-time liveness detection system, utilizing an OpenCV based GUI. While simple, this could be further improved in the future to allow for liveness as a service platform, allowing developers to calculate liveness on the cloud, rather than relying on built-in features. For the screenshots below, videos from the Replay-Attack dataset were used, but this system also allows for webcam access in real time. This is a proof of concept of a future liveness system that could be put into production.

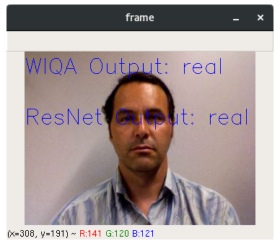


WIQA Output: fake

ResNet Output: fake

(x=268, y=208) ~ R:173 G:181 B:170

Figure: A spoofed image being correctly classified as fake by both liveness tests. Frames obtained from the Replay-Attack test dataset.



WIQA Output: real

ResNet Output: real

(x=308, y=191) ~ R:141 G:120 B:121

Figure: A real image being correctly classified as real. Frames obtained from the Replay-Attack test dataset.

One additional finding by testing in real time was that the CNN based liveness test was often correct in it's prediction, but occasionally would switch to a prediction of 'fake' for a single frame. Therefore, the liveness test could be improved by analyzing several frames, and determine the modal liveness from that set of frames.

Furthermore, and improvement that can be made to this is to speed the calculation of each liveness test up a bit. Currently, each liveness test is called and executed individually, in series. This could be parallelised, either on one machine or by using distributed computing technologies.