Face Verification System with Liveness Detection

Divyansh Srivastava[1], Priyansh Shukla[1], Ashish Kumar Sahani[2]
[1]*Department of Mechanical Engineering, Indian Institute of Technology Ropar*
[2]*Center for Biomedical Engineering, Indian Institute of Technology Ropar*
2018meb1009@iitrpr.ac.in, 2018meb1022@iitrpr.ac.in

***Abstract* - Many developments have taken place in the field of face-recognition and liveness analysis to improvise various device securities and attendance verification systems. Many approaches have incorporated 3D analysis of the face to predict the liveness of the person in front of it. Our method tries to account for this problem without using advanced 3D imaging techniques or hardware. This results in a solution that is both, more economical and also much easier to deploy. It consists of two parts; the former helps in face verification and the latter to check the liveness of the face. In the first part, we have used a model based on Google's FaceNet Model which learns a mapping from face images to compact Euclidean space distances, which directly correspond to the measure of similarity of the images. Once the space has been produced, face verification can be easily implemented using standard techniques with embeddings as feature vectors. For the second part, we have employed a cascaded multi-task framework that extracts certain features from the facial image which are then used to check for liveness by tracking their relative displacements. These extracted features were used to check the liveness of the person's face by asking them to perform some tasks in a random order like head and facial movements etc.**

I. Introduction

Many technologies like 3-D face shape analysis [1], virtual 3-D face imaging [2] and 3-D facial projection analysis [3] have evolved recently, intending to develop a virtual three-dimensional face of the person in front of the camera to enhance liveness detection accuracy. The development of these technologies requires high capitals, a wide range of technological geniuses, experimentation of avant-garde ideas and technologies, and highly developed workplaces. Coupling the complimentary devices required for this purpose with the camera alone can be a very tedious task. So, in order to account for all these issues and develop a feasible facial recognition technology we have incorporated the inception model coupled with a query-based face motion recognizer hence providing simple liveness detection with very good accuracy.

There have been a lot of recent developments in computer vision including CNN models and Neural Networks in general. Some of the latest works that may be used for facial recognition as well as liveness detection include – Facial Emotion Recognition Using Transfer Learning of AlexNet (2020) [4], Human action recognition using attention-based LSTM network with dilated CNN features (2021) [5] and, SLNet: Stereo face liveness detection via dynamic disparity-maps and convolutional neural network (2020) [6]. Our method incorporates Google's FaceNet Inception Model [7] with an input image size of $96 \times 96 \times 3$ pixels to achieve higher speed results across all devices. The model was trained on a triplet data extracted from google photos database, a large collection of photos from Facebook referred to as the Social Face Classification (SFC) dataset, labeled faces in the Wild, and YouTube faces (that is, extracting frames from videos). The triplets comprised a positive pair (different images of the same person's face) and a negative pair (images of different persons' faces). Stochastic gradient descent on triplet loss was used for optimization.

II. Machine Learning Models

A. Google FaceNet based Inception Model

The first part of our setup focuses on the face recognition part. We have employed the Inception Model which works on a complete end to end deep learning basis. The model is trained based on the triplet loss function which is considered ideal for analyzing two very similar or dissimilar images. This loss is small for similar face images, i.e., the square distance between them is small, whereas the loss is quite large for dissimilar images with non-matching features and square distances between them diverges. This model works on the principles of conversion of an image to an encoded 128-dimensions rich feature representation, which is closer for similar images in terms of Euclidean distances and farther in case of non-similar images. It works on the principle of one-shot learning in which we provide only one image of the person whose face is to be recognized. The model tries to match this image's embedding vector to all images presented to it and if the embeddings are very close in terms of squared distance, then the model detects the face to be of the same person.

Triplet Loss - The embedding can be expressed as $f(x) \in \mathbb{R}^d$. It maps an image x to a point on a d-dimensional Euclidean space. Also, the embedding is constrained to the d-dimensional hypersphere, i.e. $\left||f(x)|\right|_2 = 1$. This loss is motivated in [7] in the context of nearest-neighbor classification. Here it is ensured that an image x_i^a (anchor) of a specific person resembles closely in terms of distance to all other images x_i^p (positive) of the same person and different to any image x_i^n (negative) of any other person. Thus, required equation can be expressed as –

$$\left||f(x_i^a) - f(x_i^p)|\right|_2^2 + \alpha < \left||f(x_i^a) - f(x_i^n)|\right|_2^2 \qquad (1)$$
$$\forall \left(f(x_i^a), f(x_i^p), f(x_i^n)\right) \in T.$$

978-1-6654-4175-9/21/$31.00 © 2021 IEEE

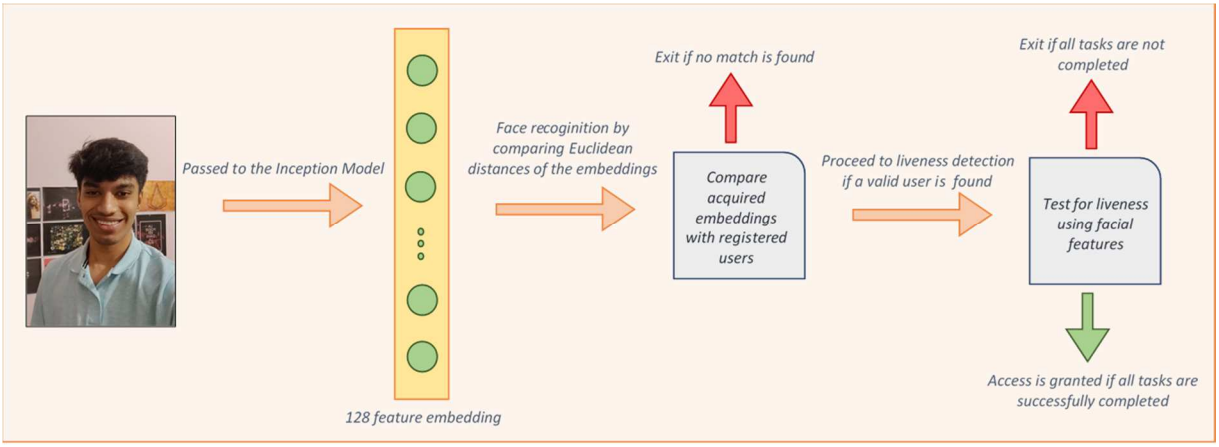

Figure 1: Brief Workflow of the Setup

Here α is a range of error that is allowed between positive and negative pairs, T represents all possible triplets in the training set and has a cardinality of N. Therefore, final loss that is to be minimized is given by –

$$L = \sum_i^N \left[\left\| f(x_i^a) - f(x_i^p) \right\|_2^2 - \left\| f(x_i^a) - f(x_i^n) \right\|_2^2 + \alpha \right]_+ \quad (2)$$

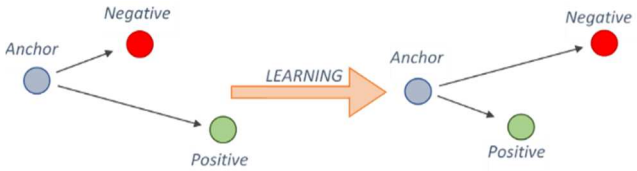

Figure 2: Representation of Triplets based on Euclidean Distances

Triplet selection is one of the most important parameters when training the model. If similar triplets are provided as input to the model, the Euclidean distances between the facial images will be small leading to an increase in the number of false-positive training examples. Also, the convergence will be very slow. To tackle this problem, hard triplets are chosen for model training. This means that the similarity between an anchor and its positive sample is low and that of its negative sample is fairly large. Thus, the triplet loss is greater and optimization steps (weights adjustments) taken for training the model are sizeable, resulting in a quicker convergence and better accuracies.

Inception Model **-** We have implemented the inception model with an input size of $\mathbf{96 \times 96 \times 3}$**.** This results in lower accuracies as compared to the original model (due to discounted input features) but performs faster on less computationally competent devices with decent accuracy. A single-layered component of the network comprises of $\mathbf{1 \times 1}$ convolution followed by $\mathbf{3 \times 3}$ and $\mathbf{5 \times 5}$ convolutions along with a $\mathbf{3 \times 3}$ MaxPool layer. The network comprises of 22 such layers, thus helping to create healthy image embeddings. Average pooling layers were also incorporated at the end along with a couple of dropout layers added throughout the model to help prevent overfitting.

B. Feature Extraction and Liveness Detection

So as to extract the real-time positions of the facial landmarks, we used Multi-task Cascaded Convolutional Network (MTCNN) ***[8]***. It divides the task of face detection and facial landmark detection into three smaller subtasks or levels to aid achieve the best output and minimize false results. At the first level, it uses a quicker, shallower convolutional neural network to roughly identify all the windows in which faces can be present. At the second stage, a more complex convolutional neural network is incorporated which filters these windows and outputs the bounding box coordinates of the windows which contains a face with more than 80% surety. Further, at the third level, an advanced convolutional neural network is used to extract the facial feature coordinates from the faces present in the detected windows.

A given image is first resized to the required dimensions to build an image suitable for input, which is further fed to the three-stage cascaded network as follows:

Stage 1: It uses a fully convolutional network, called Proposal Network (P-Net), that extracts the facial windows and bounding box vectors or coordinates for each. Then we use the predicted bounding box vectors to standardize the candidates. After that, it employs non-maximum suppression (NMS) whose function is to eliminate the highly overlapping boxes which possibly represent the same portions of the image.

Stage 2: All of the above are passed to another CNN, called Refine Network (R-Net), which disqualifies a large number of wrong candidates, corrects and finalizes the bounding box enclosing faces, and Non-Max Suppression merging of the boxes.

Stage 3: This stage is similar to the second stage, but in this stage, the aim is to describe the face in further detail. The network will predict the locations of five facial landmarks, namely the coordinates of the two eyes, nose, and edges of the mouth.

MTCNNs perform a bit slower as compared to other techniques like haar-cascaded facial detection ***[9]*** techniques etc. but they result in superior accuracies in real-time

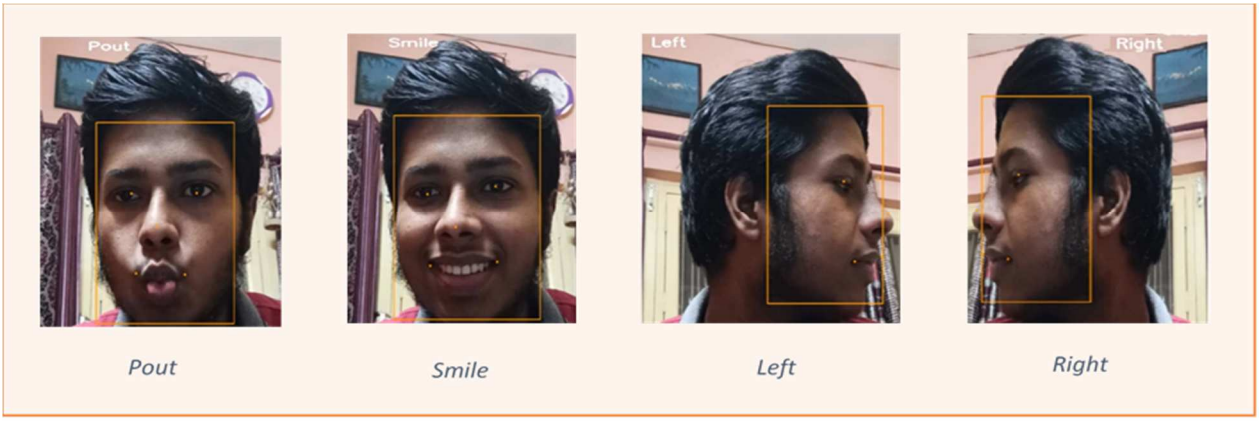

Figure 3: Some Liveness Testing Demonstration. Bounding box for the face and location of features are also marked

detections. And for our setup to work effectively, we do require high accuracy feature tracing in real-time to analyze liveness using their relative positions.

After extracting the relevant features, the task was to track the changes in their relative positions to detect the motion of the head, lips, etc, which are considered to predict the liveness of the image in front of the camera. The output from the MTCNN models gives the exact coordinates (pixel position) of these facial features even in real-time. Hence, we can detect the head movements by somehow keeping the track of the change in the relative position of facial features on the pixel plane. This can be done by determining distances like the inter-eye gap and the relative change in the pixel coordinate of nose etc. Lip dependent movements like smile and pout are also used as a task to unlock a device. The user can be asked to perform these head and lip movements in any random order before the access is provided to him/her. If the algorithm detects a deviation from the instructions in any frame, the process is terminated at that very instance thus enhancing the security.

C. CORRELATIONS BETWEEN FACIAL FEATURES

All the expressions of our face and head are accompanied by relative movements of its features like eyes, nose, lips, etc. Whenever a person looks towards his/her right, the position of the nose shifts rightwards with respect to its original location. Also, the left and right eyes seem to be in the same line at this instance. Likewise, when a person looks towards left the location of the nose shifts leftwards with respect to its original coordinates, and the right and left eyes seem to be level. Similarly, when a person smiles, the lips extend horizontally, and the distance between the edges of his/her lips increases. Also, the relative distance between the left and right eyes to the left and right lip edges decreases respectively. In the case of a pout, the distance between the edges of the lip decreases, and the relative distance between the left and right eyes to the left and right lip edges increases respectively. Our setup will use these characteristics to determine the liveness of a person along with face verification so that the system cannot be tricked by photographs.

III. DETAILED WORKING OF THE SETUP

We have tried to cover all the salient points of the implementation and handle any exception that may hinder the process. When we first run the setup, it has no prior knowledge about the users who might use it. It builds the model architecture (Inception) in TensorFlow and loads all the weights of the trained model from a .csv file which are necessary for converting the input face image to its respective embeddings. Next, it loads the MTCNN class object for facial feature detection. It then asks if the user is registered for accessing the system or not. If the user enters ‘yes’, then it will proceed to face verification and liveness, else it asks for a pin/password to add the person’s face as a valid user. While adding a new user, the model accepts the name of the user and captures ten frames of the user’s face. Out of these frames, it selects the one with the least noise and passes it to the model to extract its embeddings and store it for future use. This is a one-time job and has to be done at the start of the setup.

Once the user is registered, he may proceed to the verification and unlocking part. Once this process is initiated, the facial frames of the persons are extracted from the webcam, resized, and passed to the model to generate the embeddings for each frame. The mean distance between these embeddings and registered embeddings should be very less or below a certain threshold (α) which is a hyperparameter that can be tuned according to the device and camera specifications. The maximum number of frames extracted for this task is set to be 50. We set a minimum requirement of the frame (10 in our test setup) to satisfy the minimum threshold criterion in order to proceed to the next step. If out of 50 frames, at least 10 frames are not closely matched to any single registered user’s image, then the access is denied and the process is terminated. This minimum number of frames can be tweaked based on requirements and level of security. If a person removes his face from the camera before 50 frames are extracted, the system waits for 10 seconds before terminating. The process approximates around 1 second on a mediocre system processor.

After face verification, the process of liveness analysis commences. It works best when the position of the camera and the person is fixed and does not change during the time of evaluation. As the process starts, we analyze the first frame of the person's face and store the location of the various features in the plane of the frame. All the positions are relative to the size of the frame as extracted by MTCNN. Now we randomly display a series of tasks like moving the head towards the right or left or smile or pout on top of the live camera window open in front of the user. All frames are been continuously analyzed throughout the process. If the camera says 'right', then the task is not changed till the distance between the eyes along the horizontal direction decrease under a certain threshold and the nose moves the right along the horizontal direction relative to the nose location stored at the start of the process. As soon as the algorithm detects a frame that satisfies the above conditions, it displays the next task. Similarly, in the case of a smile, it will compare the distance between the edges of the lips in the starting frame and the current frame. If the distance increases beyond a certain factor of the original distance (e.g., becomes 110% of the original distance), it is detected as a smile. Correspondingly, all the tasks are performed, and if the user passes all of them then the process is completed and access is granted to the user. If the user moves out of the camera in between the process then the process is terminated. This is done so that no person can switch in between the process. A maximum of 150 frames are granted to the user to complete the tasks which seemed sufficient during our testing. All the thresholds are set relative to the original distances instead of hard coding them. This is to eliminate the tuning required in case of a change in the camera's resolution or distance between the camera and the person. This process takes a maximum of 5 seconds on our system and depends on the speed of the processor. It's performance also depends on the quality of the camera as a high pixel camera can pinpoint the feature locations with more accuracy than a low-quality camera, but at the cost of processing speed due to a higher memory requirement of the frame with higher pixels or features.

IV. Performance Evaluation

We tested both the different parts of the model separately as well as a single unit having five different users as valid users. The FaceNet based Inception model gave an accuracy of 83 percent on random test images with some personal images of the users in between with an F1 score of around 0.9. The second part of the model centered around facial feature extraction and their movement detector for liveness accuracy worked very well with an accuracy of around 97%. It worked satisfactorily fine during real-time testing as well. After coupling both face verification and liveness detection algorithms it gave an accuracy of around 90% even during real-time testing.

V. Conclusion

Though this model architecture works fairly well in real-time, there is still scope of certain optimizations and improvements to enhance its performance. We have employed an image size of $96 \times 96 \times 3$ pixels as input for the inception model, but it is speculated that it can give an increased accuracy of about 89 percent for an input image size of $266 \times 226 \times 3$ against the 83 percent of our current model but at the cost of some additional computational power of course. Also, bigger datasets are always better in the case of such deep networks, hence there is a scope of improvement in this field as well. Several new activities (tasks) like the blinking of eyes, looking up and down, opening the mouth, raising eyebrows, etc. can also be added to the existing list of movements to enhance the accuracy and security. But even under limited resources and comparatively less computation and funding, this setup can be used for real-time security and digital device unlocking services as compared to more advanced techniques like infrared and three-dimensional imaging technologies.

An interesting implementation of this technology could be to replacing biometric verification systems that require physical contact with a centrally located device (fingerprint/retinal scanners). Instead, the actual physical presence of an individual employee could be detected by tying in the GPS location of his/her smartphone with the liveness detection in front of its front camera. This will help reduce crowding around any centrally located device (aid social-distancing), and make the attendance contact-free.

VI. References

[1] M. T. M. C. C. F. a. S. S. A. Lagorio, "Liveness detection based on 3D face shape analysis," *2013 International Workshop on Biometrics and Forensics (IWBF),* no. 10.1109/IWBF.2013.6547310, pp. 1-4, 2013.

[2] T. P. J.-M. F. a. F. M. Yi Xu, "Virtual U: Defeating Face Liveness Detection by Building Virtual from Your Public Photos.".

[3] M. N. D. R. a. J. D. M. De Marsico, "Moving face spoofing detection via 3D projective invariants," *2012 5th IAPR International Conference on Biometrics (ICB), New Delhi,* no. 10.1109/ICB.2012.6199761, pp. 73-78, 2012.

[4] S. N. H. A. M. N. M. R. A. S. H. &. A. H. Shaees, "Facial emotion recognition using transfer learning," *2020 International Conference on Computing and Information Technology,* no. ICCIT-1441, pp. 1-5, 2020.

[5] K. e. a. Muhammad, "Human action recognition using attention based LSTM network with dilated CNN features.," *Future Generation Computer Systems,* vol. 125, pp. 820-830, 2021.

[6] Y. A. U. L.-M. P. a. M. L. Rehman, "SLNet: Stereo face liveness detection via dynamic disparity-maps and convolutional neural network.," *Expert Systems with Applications,* vol. 142, no. 113002, 2020.

[7] F. D. K. a. J. P. Schroff, "Facenet: A unified embedding for face recognition and clustering," *Proceedings of the IEEE conference on computer vision and pattern recognition,* 2015.

[8] L. e. a. Zhang, "Multi-task cascaded convolutional networks based intelligent fruit detection for designing automated robot.," *IEEE Access 7,* no. 56028-56038, 2019.

[9] R. T. S. A. a. R. M. R. G. Sharma, "A novel real-time face detection system using modified affine transformation and Haar cascades," *Recent Findings in Intelligent Computing Techniques. Springer, Singapore,* no. 193-204, 2019.

[10] I. S. G. E. H. Alex Krizhevsky, "ImageNet Classification with Deep Convolutional Neural Networks," *Advances in Neural Information Processing Systems 25,* 2012.

[11] I. S. a. G. H. Alex Krizhevsky, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems,* p. 1106–1114, 2012.

[12] D. R. W. a. T. R. Martinez, "The general inefficiency of batch training for gradient descent learning. Neural Networks," no. 16(10):1429–1451, 2003.

[13] Z. L. X. S. J. B. a. G. H. H. Li, "A convolutional neural network cascade for face detection," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 5325-5334, 2015.

[14] S. R. Y. W. X. C. a. J. S. D. Chen, "Joint cascade face detection and alignment," *Proc. ECCV,* 2014.

[15] a. D. R. X. Zhu, "Face detection, pose estimation, and landmark localization in the wild," *IEEE Conference on Computer Vision and Pattern Recognition,* pp. 2879-2886, 2012.