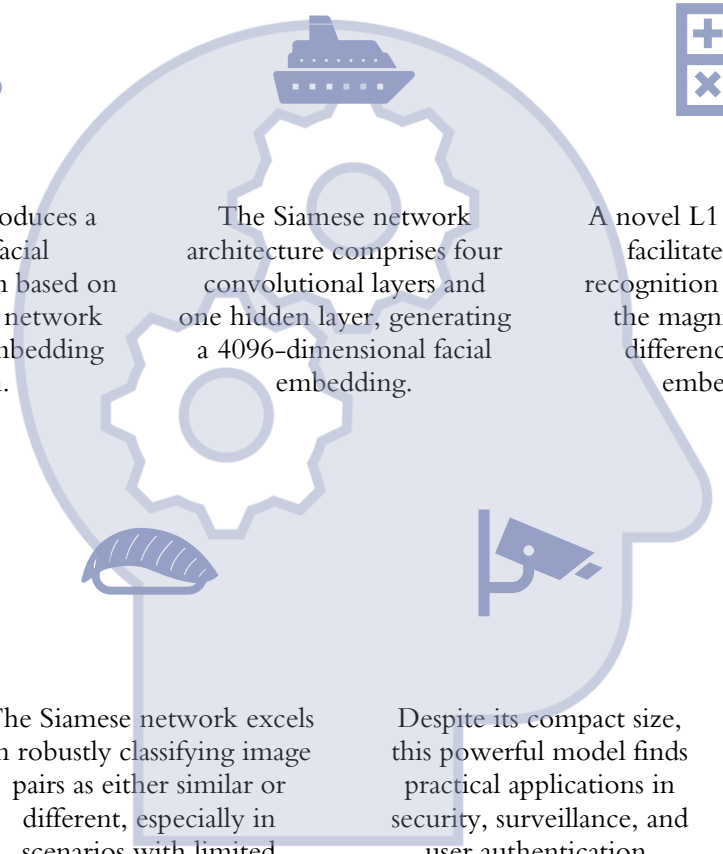
ONE-SHOT FACIAL RECOGNITION: USING SIAMESE NETWORKS

By:-

Krupal M Patel

Yash P Prajapati



The project introduces a single-shot facial recognition system based on a Siamese neural network with a unique embedding approach.

The Siamese network architecture comprises four convolutional layers and one hidden layer, generating a 4096-dimensional facial embedding.

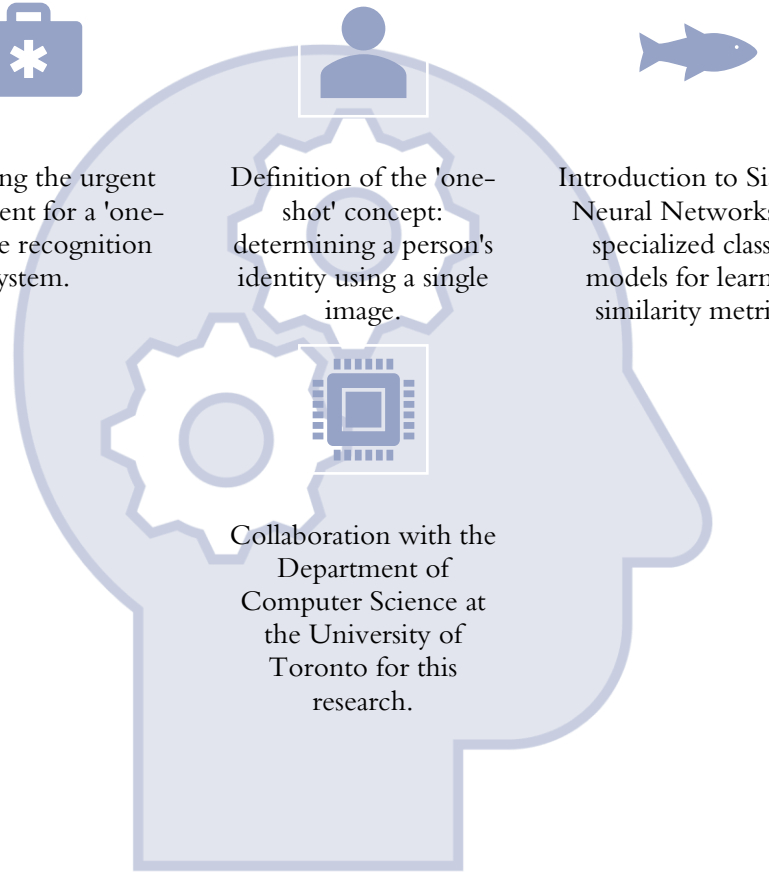
A novel L1 distance layer facilitates one-shot recognition by calculating the magnitude of the difference between embeddings.

The Siamese network excels in robustly classifying image pairs as either similar or different, especially in scenarios with limited training data.

Despite its compact size, this powerful model finds practical applications in security, surveillance, and user authentication, particularly in situations with limited training sets.

ABSTRACT

This project presents a single shot facial recognition system based on Siamese neural network having a different kind of embedding.

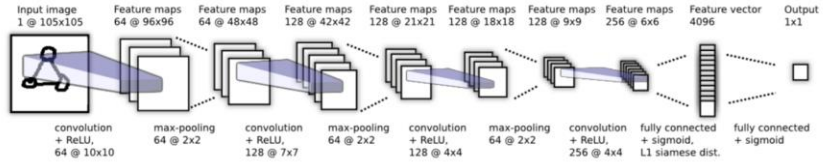


INTRODUCTION

- Overview of the significant advancements in facial recognition technology.
- Identification of common applications, including security systems and user authentication.
- Recognition of the challenge posed by the need for large datasets for effective system operation.

MODEL ARCHITECTURE

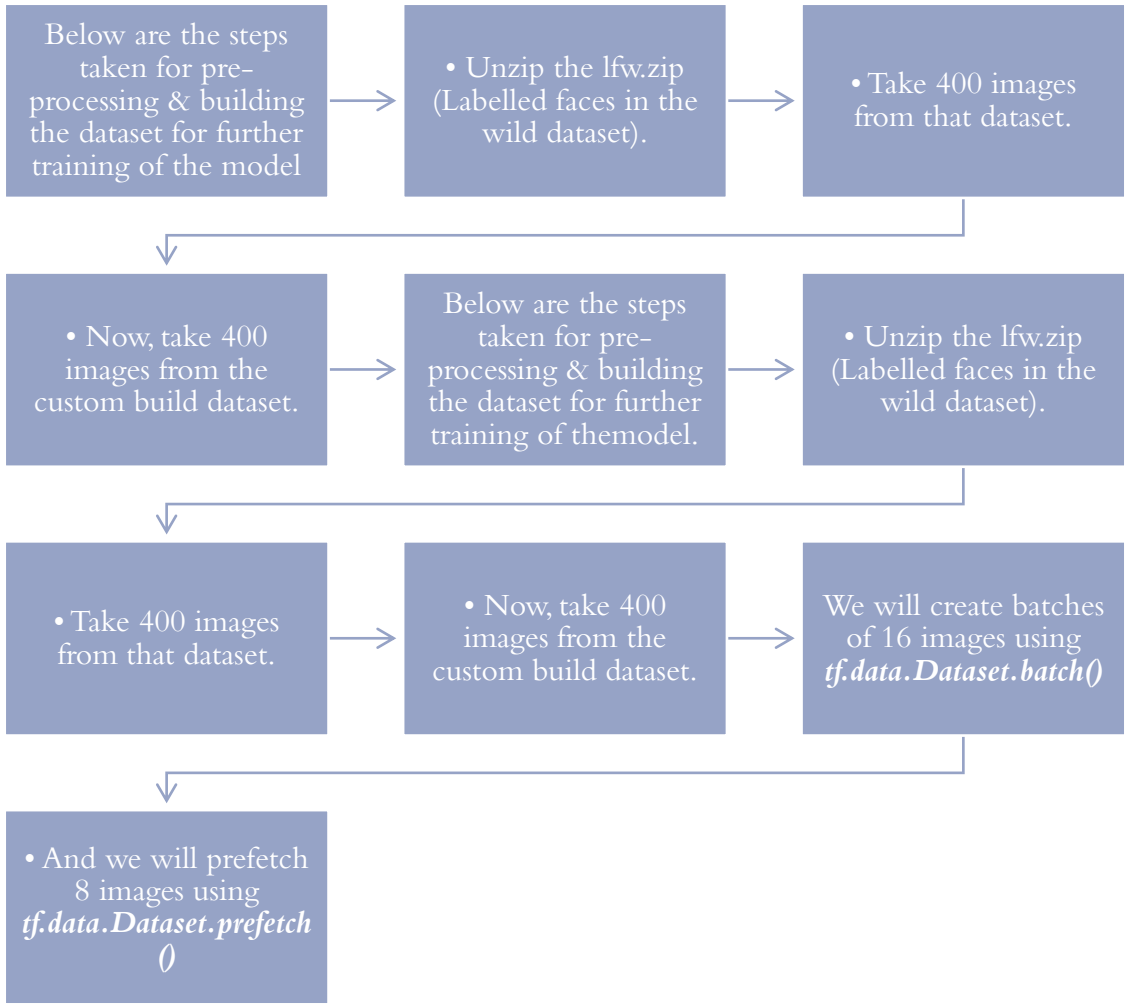
The model architecture consists of Input Layer of shape (105,105,3), 4 2D- convolution layers, 4 Max Pooling layers, L1 siamese distance calc layer, 1 fully connected layer, 1 sigmoid layer and then output layer having single output



DATASET

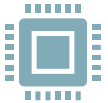
For training & testing of the model we have used Labeled Faces in the Wild, a database of face photographs designed for studying the problem of unconstrained face recognition. The data set contains more than 13,000 images of faces collected from the web. Each face has been labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the data set.

PREPROCESSING



TRANING

In training the model, we will calculate the losses using ***tf.losses.BinaryCrossentropy()*** and we



will use the adam optimizer. As we have performed the training on Apple silicon (M2 Pro chip),

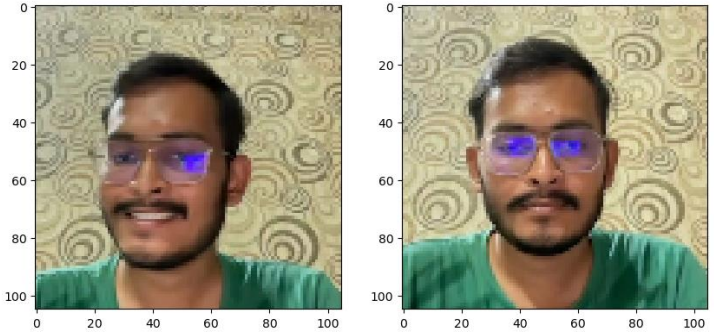
we were advised to use the ***tf.optimizers.legacy.Adam()*** as optimizer. We have customized the

train function according to the batch training. After training the model with learning rate of

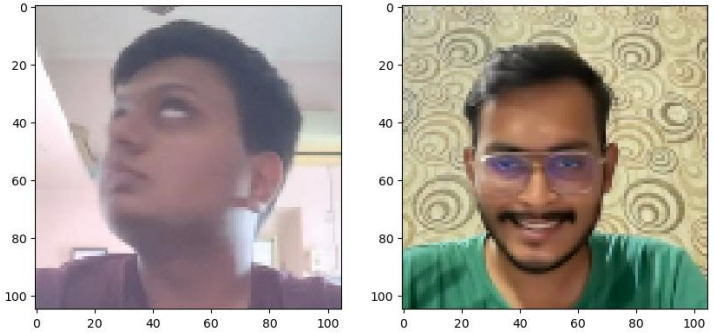
0.0001, and 20 epoches, we got loss: 1.5464309399249032e-05.

RESULT

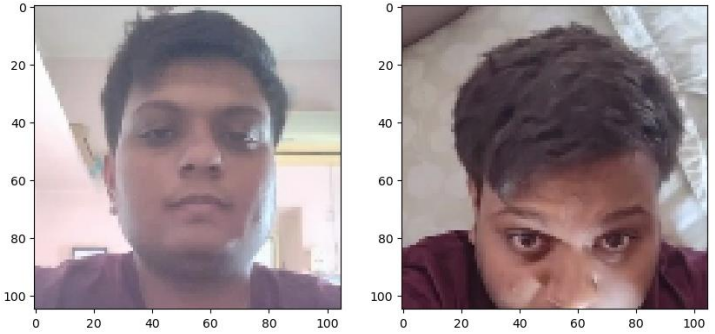
Matched



Not - Matched



Matched



APPLICATION

CCTV Cameras

Door locking system

Secure Acces

THANK YOU