Professor Hwang
EE 443
June 6, 2023
Linh Truong
Sebastian Aggussoegito
Nathanael Judah Hartanto

Face Verification: Siamese w/ Triplet Loss

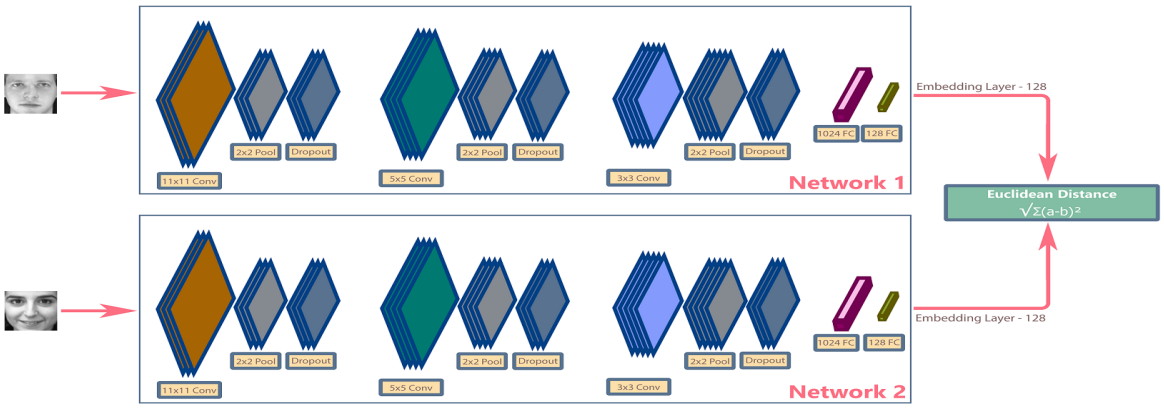
Introduction

Face verification is an advanced technology that aims to accurately identify and authenticate individuals based on their unique facial features. With the increasing demand for secure and reliable identification systems, face verification has gained significant attention and practical applications in various industries, including security, surveillance, access control, and personal device authentication.
In this project, we will be implementing and evaluating a face verification system using the Siamese network approach. The Siamese network is a deep learning architecture that learns to measure the similarity between two input images. It has proven to be effective in face recognition tasks by learning discriminative features that can differentiate between different individuals.

Siamese Network for Face Recognition

The Siamese network consists of two identical subnetworks that share weights and are trained simultaneously. It takes pairs of images as input and learns to output a similarity score that indicates how similar the two input images are. The network learns to map images of the same person closer together in the feature space while pushing images of different people apart.

The Siamese network learns to extract discriminative features from face images by optimizing a loss function that encourages the network to minimize the distance between positive pairs (images of the same person) and maximize the distance between negative pairs (images of different people). This way, it learns a similarity metric that can effectively classify whether two images belong to the same person or not.



Dataset: Labeled Faces in the Wild (LFW)

For this project, we will be using the Labeled Faces in the Wild (LFW) dataset. The LFW dataset is a widely used benchmark dataset in the field of face recognition. It contains a large collection of labeled face images of different individuals captured in unconstrained conditions.
The LFW dataset comprises more than 13,000 images of faces from various sources, including news articles, celebrity photos, and other publicly available images. It covers a wide range of identities and includes variations in pose, lighting, expression, and image quality. The dataset is carefully curated and provides ground-truth annotations for evaluation purposes.

Project Goals

The main goals of this project are:

- Implement a face verification system using the Siamese network approach.
- Preprocess the LFW dataset for training and evaluation.
- Train the Siamese network on the LFW dataset to learn discriminative face features.
- Evaluate the performance of the face verification system using appropriate evaluation metrics.
- Explore different techniques and variations of the Siamese network to improve the system's performance.
- Gain a deeper understanding of the underlying principles of face verification and contribute to the development of robust and efficient identification systems.

Through this project, we aim to leverage the power of deep learning and computer vision techniques to develop a reliable and accurate face verification system. By utilizing the Siamese network architecture and the LFW dataset, we can extract meaningful facial features and create an effective model for identifying and authenticating individuals based on their facial characteristics.

Data Preprocessing

The dataset we are using contains 1324 different individuals with 2-50 images per person. Each image is encoded in RGB and has a shape of (128, 128, 3). Each folder and image is named with a number, i.e 0.jpg, 1.jpg. Since we are using triplet loss, we want to separate our train data into triplets of (anchor, positive, negative) face data where the positive is the same person and the negative is a different person from the anchor.

```
Examples of triplets:
(('1308', '6.jpg'), ('1308', '7.jpg'), ('736', '0.jpg'))
(('422', '1.jpg'), ('422', '2.jpg'), ('1616', '12.jpg'))
(('1410', '3.jpg'), ('1410', '5.jpg'), ('1400', '1.jpg'))
(('1597', '3.jpg'), ('1597', '4.jpg'), ('495', '1.jpg'))
(('1580', '3.jpg'), ('1580', '4.jpg'), ('1072', '0.jpg'))
```

Next, we make a batch generator that converts a triplet input into batches of face data and preprocesses it before returning the data in separate lists.

```python
def get_batch(triplet_list, batch_size=256, preprocess=True):
    batch_steps = len(triplet_list)//batch_size

    for i in range(batch_steps+1):
        anchor   = []
        positive = []
        negative = []

        j = i*batch_size
        while j<(i+1)*batch_size and j<len(triplet_list):
            a, p, n = triplet_list[j]
            anchor.append(read_image(a))
            positive.append(read_image(p))
            negative.append(read_image(n))
            j+=1

        anchor = np.array(anchor)
        positive = np.array(positive)
        negative = np.array(negative)

        if preprocess:
            anchor = preprocess_input(anchor)
            positive = preprocess_input(positive)
            negative = preprocess_input(negative)

        yield ([anchor, positive, negative])
```

Encoder

The Encoder plays a crucial role in converting input images into their corresponding feature vectors. In this implementation, we leverage a pre-trained Xception model, which is built upon

the Inception_V3 model. By utilizing transfer learning, we can make use of the already learned features and expedite the training process while reducing the dataset's size requirements.

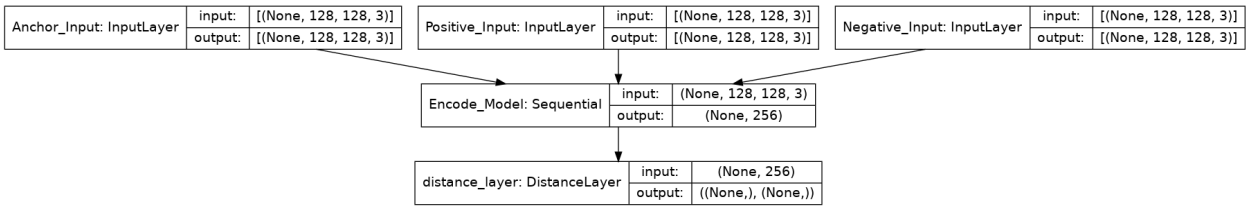
The Model incorporates a series of Fully Connected (Dense) layers, culminating in a final layer that applies L2 Normalisation to the data. L2 Normalisation is a technique that adjusts the values within the dataset so that the sum of the squares in each row is always equal to or less than 1. This normalization process helps ensure that the data is standardized and ready for further analysis or classification tasks.

Training the model

For our model, we use a Siamese Network that operates on three input images: anchor, positive, and negative. These images are processed by the encoder to obtain their respective feature vectors. These feature vectors are then fed into a custom distance layer, which calculates the distance between the anchor and positive pairs, as well as the anchor and negative pairs. A custom layer will be used to compute the distance.

Distance Formula:

$$n = \| f_i^a - f_i^n \|_2^2$$
$$p = \| f_i^a - f_i^p \|_2^2$$



To compute the triplet loss using the embeddings generated by the Siamese network, we will implement a model with a custom training loop and loss function. This will allow us to calculate the loss based on the three embeddings: anchor, positive, and negative. Additionally, we will utilize a Mean metric instance to keep track of the loss during the training process. This metric will help us monitor the average loss across multiple batches and epochs. By using a custom training loop and loss function, we have the flexibility to incorporate the triplet loss calculation and track the loss metric effectively.

Triplet Loss Function:

$$Loss = \sum_{i=1}^{N} \left[\|f_i^a - f_i^p\|_2^2 - \|f_i^a - f_i^n\|_2^2 + \alpha \right]_+$$

Now we will proceed with training the siamese_model on batches of triplets. During training, we will print the training loss and evaluate additional metrics by testing the model at the end of each epoch. Furthermore, we will implement a mechanism to save the model weights whenever it achieves a higher accuracy than the previously recorded maximum accuracy. This will allow us to keep track of the best-performing model throughout the training process.

Testing the model

We also create a function that will make predictions on the test data and collect metrics such as accuracy, means, and standard deviations. Additionally, it will print out the accuracy of the model after the testing process.

```
EPOCH: 1          (Epoch done in 135 sec)
Loss on train     = 0.50190
Accuracy on test = 0.86640

EPOCH: 2          (Epoch done in 92 sec)
Loss on train     = 0.30146
Accuracy on test = 0.89474
```

After training the model, we need to extract the encoder to use it to encode images and then get the feature vectors to compute the distance between those images as well as save it for later use.

To compute the distance between the encodings of the images, we can use a distance formula such as Euclidean distance. By comparing the distance with a predefined threshold, we can determine whether the images are considered "different" or "same".

Conclusion

We investigated the topic of face verification in this research, focusing on the creation of a face recognition system utilizing the Siamese approach and triplet loss. We extracted facial traits using deep learning algorithms and computer vision techniques, compared them to a database of known faces, and provided reliable verification findings.

The Siamese method, which used twin neural networks to build good representations of faces, proved to be a powerful strategy for face recognition. We encouraged the network to map

embeddings of the same persons near together and those of different individuals far away in the feature space by training the networks with triplet loss. This improved the network's discriminative ability and enabled accurate identification and verification of individuals.

We obtained a better grasp of the basic principles of face verification, such as picture preprocessing, deep neural networks, and loss functions, as a result of this effort. We investigated the significance of dataset selection as well as the availability of publically available datasets like CelebA and LFW.

In addition, we emphasized the importance of assessment measures like True Positive Rate (TPR) and False Positive Rate (FPR) in evaluating the performance of face verification systems. These metrics offered information about the system's capacity to accurately validate genuine faces and reject fake faces.

Overall, this research led to the creation of strong and efficient identification systems, addressing the growing demand in numerous industries for secure and trustworthy identifying approaches. The Siamese technique with triplet loss demonstrated its efficacy in face recognition tasks and provided avenues for further research and development in the field of face verification.

References

- FaceNet: A Unified Embedding for Face Recognition and Clustering: https://arxiv.org/abs/1503.03832
- Image similarity estimation using a Siamese Network with a triplet loss: https://keras.io/examples/vision/siamese_network/
- Celebrity Face Recognition: https://www.kaggle.com/ravehgillmore/celebrity-face-recognition/
- Face Recognition using Siamese Networks: https://medium.com/wicds/face-recognition-using-siamese-networks-84d6f2e54ea4
- Face Recognition: Siamese w/ Triplet loss: https://www.kaggle.com/code/stoicstatic/face-recognition-siamese-w-triplet-loss/notebook