Exploring Facial Recognition

Mukunda Reddy
AI21BTECH11021

Introduction

Facial recognition is the process of identifying or verifying individuals based on their facial features. It has gained significant attention due to its wide range of applications, such as access control, surveillance, and personalized user experiences. However, achieving accurate and reliable facial recognition systems re- mains challenging, particularly in unconstrained environments with variations in illumination, pose, expression, etc. extend this

Problem Statement

Traditional facial recognition methods often require extensive feature engineering to address challenges related to changes in orientation, pose, and expression of faces. This process can be complex and time-consuming, leading to suboptimal performance in real-world scenarios.

Deep learning techniques offer a promising solution to these challenges by automatically learning relevant features from raw data. However, the effectiveness of deep learning models heavily relies on the availability of large and diverse training datasets. By conducting a comprehensive comparative analysis, we aim to shed light on the strengths and weaknesses of each method.

Experimental Plan

We intend to execute experiments utilizing established facial recognition datasets to assess the efficacy of various methods. Performance evaluation will encompass accuracy, computational efficiency, and resilience to variations in illumination, pose, and expression. Our dataset selection will prioritize diversity, incorporating variances in orientation, illumination, pose, and expression to ensure comprehensive and robust evaluation of the methods under consideration.

Deep Learning Method

We present our approach to facial expression recognition using the Yale dataset, employing a Siamese Network for facial expression recognition using transfer learning with a pre-trained VGG16 model trained with triplet loss. The Yale dataset is chosen for its diversity in facial expressions, providing a robust platform for training and testing our model.

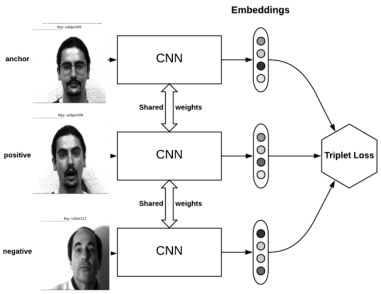
Yale Dataset

The Yale dataset is a widely used benchmark dataset in the field of facial recognition. It contains high-resolution images of individuals under various lighting conditions, poses, and facial expressions. This diversity makes it suitable for training models robust to real-world scenarios.

Network Architecture

Traditional neural network classifiers for face recognition can encounter challenges when dealing with datasets containing numerous individuals. Each person in the dataset becomes a class, leading to a high number of classes, which can be inefficient and demanding in terms of computational

resources. Moreover, training such classifiers on a large dataset requires substantial time and effort. This is also the case with classical techniques.
In contrast, Siamese networks offer an elegant solution to these challenges. Instead of directly classifying images, Siamese networks focus on learning similarities between pairs of images. They achieve this by comparing the features extracted from two input images and determining how similar or dissimilar they are.
A Siamese network is a type of neural network architecture that consists of two identical subnetworks, each taking a different input. These subnetworks learn to extract features from the input data and are then compared at a feature level to determine similarity or dissimilarity.



[t]

Figure 1: Network Architecture

[?]

Transfer learning represents a nuanced approach in machine learning, leveraging insights acquired from training one model to enrich the performance of another model tackling similar tasks. This methodology capitalizes on pre-trained models and their accumulated knowledge to streamline the training process of subsequent models while augmenting their ability to generalize across a spectrum of tasks. In our approach, we employed the VGG-16 architecture as the foundation for our convolutional neural network (CNN). Enhancing this architecture, we appended an additional convolutional layer at the backend and introduced two linear layers at the front end. The final layer of the network outputs embeddings of length 128, encapsulating the essential features extracted from the input data. This fusion of pre-trained knowledge with task-specific adjustments enables our model to swiftly adapt to new tasks while fostering robust generalization capabilities across diverse domains.

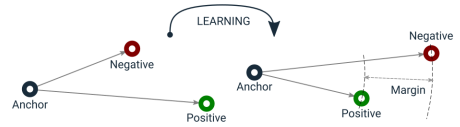
Triplet Loss

Triplet loss is a loss function commonly used in Siamese network training. It works by comparing three samples: an anchor sample, a positive sample (similar to the anchor), and a negative sample (dissimilar to the anchor). The objective is to minimize the distance between the anchor and the positive sample while maximizing the distance between the anchor and the negative sample. This encourages the model to learn embeddings where similar samples are closer together and dissimilar samples are farther apart. [?]

$$\|f(x_{a_i}) - f(x_{p_i})\|_2^2 + \alpha < \|f(x_{a_i}) - f(x_{n_i})\|_2^2,$$
$$\forall (f(x_{a_i}), f(x_{p_i}), f(x_{n_i})) \in T$$

where α represents the enforced margin between positive and negative pairs, and T denotes the set of all possible triplets in the training set with cardinality N. The loss function to be minimized is defined as:

$$L = \sum_{i=1}^{N} \left(\|f(x_{a_i}) - f(x_{p_i})\|_2^2 - \|f(x_{a_i}) - f(x_{n_i})\|_2^2 + \alpha \right)$$



[t]

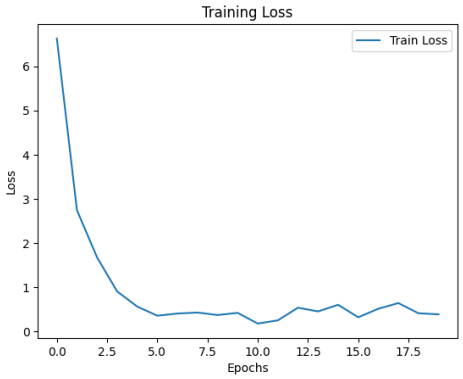
Figure 2: triplet loss

Sample Selection using Semi-Hard Samples

In our implementation, we adopted the strategy of semi-hard sample mining during training. Semi-hard samples are those for which the negative sample is farther from the anchor than the positive

sample but still contributes to the loss. By selecting semi-hard samples, we strike a balance between encouraging meaningful updates to the model and preventing it from being overwhelmed by excessively difficult examples.

our training regimen unfolds with the generation of embeddings after each epoch using the updated model. Subsequently, we curate semi-hard triplets from these embeddings, facilitating the fine-tuning of the network. The training process continues iteratively until the loss converges to a stable state. The ensuing results showcase a discernible pattern wherein the loss stabilizes after a handful of epochs, signifying the convergence of the training process.



[t]

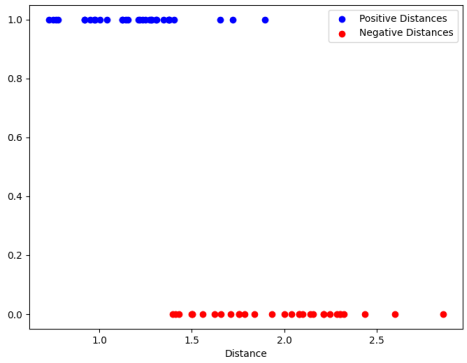
Figure 3: Training Loss

After approximately 18 epochs, employing a batch size of 32 and comprising a total of 50 batches, we observe the error curve reaching a state of stabilization, aligning with our desired outcome. This signifies the convergence of our training process, wherein the network attains a consistent level of performance, indicative of its optimized state.

Validation

Embedding Extraction: Once trained, the Siamese network is used to extract embeddings for each face image in the dataset. These embeddings represent the unique features of each face in a lower-dimensional space.

Distance Measurement: The distance between embeddings is calculated using a chosen metric, such as Euclidean distance or cosine similarity. Smaller distances indicate greater similarity between faces, while larger distances signify dissimilarity.
Threshold Selection for Classification: Now, to use these embeddings for face recognition, we need a threshold to classify whether two faces belong to the same person or not. This threshold is crucial for determining the boundary between positive and negative samples.
Threshold Selection: The threshold is typically selected based on an evaluation metric, such as accuracy, precision, or F1-score, using a validation dataset. The goal is to find the threshold that maximizes the chosen metric.



[t]

Figure 4: distance between the similar images and dissimilar images

Testing

The classification report here evaluates the performance of a model on a binary classification task, where class 0 represents negative samples(images of different people) and class 1 represents positive samples(images of same person). The precision, recall, and F1-score metrics provide insights into how well the model performs for each class and overall.

	precision	recall	f1-score
0	0.85	0.88	0.86
1	0.87	0.84	0.86
accuracy			0.86
macro avg	0.86	0.86	0.86
weighted avg	0.86	0.86	0.86

Table 1: Classification report

Our model demonstrates proficiency in both class 0 and class 1, which is evident in the commendable precision and recall scores. This adeptness extends seamlessly to the testing data, underscoring the model's robust generalization capabilities. Moreover, as the dataset size expands, the efficacy of our model becomes increasingly apparent, yielding markedly improved results.

Conclusion

Local Phase Pattern (LPP) shows significant promise in facial recognition due to its ability to handle variations in illumination, expression, and pose. By capturing both local texture details and overall facial structure, LPP methods offer improved accuracy and robustness. However, optimization of parameters and integration with other techniques could further enhance its effectiveness in real-world applications.

Gabor filters, widely used for facial feature extraction, capture spatial frequency information and are effective in addressing challenges related to illumination and expression changes. They operate by analyzing different frequencies and orientations, which helps in distinguishing facial features despite variations. However, their limitations become apparent when dealing with pose variations. For each variation in images, Gabor filters require specific tuning to accommodate different poses, which can be complex and may not consistently yield optimal results. This need for extensive tuning can reduce their effectiveness in handling diverse real-world scenarios.

Deep learning models stand out for their superior accuracy and ease of implementation. These models eliminate the need for extensive feature engineering as they automatically learn relevant features from raw data. Unlike traditional methods, which require training on predefined facial classes, deep learning models exhibit remarkable generalization, handling pose variations and complex datasets more effectively. Experimental results have consistently demonstrated that deep learning approaches not only address the challenges of facial recognition but also adapt well to evolving datasets, making them the superior choice for contemporary facial recognition tasks.

Moreover, deep learning enables the creation of face embeddings, which are high-dimensional vector representations of facial features. These embeddings allow for efficient and scalable comparison of faces. Modern vector databases, such as Faiss or Annoy, are optimized for high-dimensional nearest neighbor searches and can handle large-scale face comparison tasks efficiently. By storing face embeddings in these vector databases, we can quickly retrieve and compare faces across extensive datasets, significantly improving the scalability and speed of facial recognition systems. In contrast, traditional methods, which rely on direct feature comparisons or predefined facial classes, often struggle with scalability and efficiency when handling large volumes of data, making deep learning-based face embeddings a more effective solution for large-scale face recognition tasks.