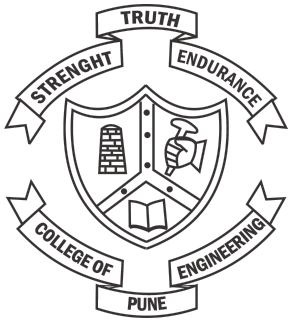

Improving Face Recognition with Domain Adaptation

A Project Report

Submitted by

Anjali Sharma

Sonali Shinde

Submitted to

Prof. Aparna Biswas

in partial fulfillment for the award of the degree of Master in Technology in Computer Engineering

1. ABSTRACT

The face recognition algorithms used in the past are based on various datasets that achieved over 99% accuracy. But the performance of these algorithms is still not enough for the real world applications. one of the problems is that these datasets come from web collected datasets of celebrities which are quite different from the faces of normal people captured in daily life. In a way, they are different in face distribution. so, replacing the dataset with the right distribution will be an optimal solution. The next problem would be collecting the pictures of common people because of privacy concerns. So it is useful to develop a method that transfers the knowledge in the data of different face distributions to help improve the final performance. Here, we will try to move a large face dataset whose distribution is different from traditional sets and show the improvement of accuracy with a simple domain adaptation technique. This will be the latest attempt at applying domain adaptation in the constraintless face recognition problem with million scale data.

2. INTRODUCTION

Face recognition is not only the concept of merely detecting the presence of a human face but it is more of identifying a specific individual. It is widely used in public security, finance security, commercial domain and so on. Due to its wide applications, face recognition has become a core problem and one of the most popular research topics in computer vision. It includes two different but related tasks, face verification (are these two pictures the same person) and face identification (who is this person). Face verification can be extended to solve face identification tasks by repeating one-on-one comparison.

Nearly all recent methods have been evaluated on the Labeled Faces in the Wild (LFW) dataset. But its performance is still not enough for real-world applications. One problem is the data bias. The faces in LFW and other web collected datasets come from celebrities. They are quite different from the faces of a normal person captured in daily life. In other words, they are different in the face distribution. Replacing the training data with the right distribution is a simple solution. But the photos of common people are much harder to collect because of privacy concerns. Besides, a generic recognition system is required to be transferred to a domain specific application for performance. Both can be formulated as Domain Adaptation, which transfers the knowledge in the source domain to the target domain.

In the testing phase of face verification, the distance between face pairs is compared with a pre-computed threshold. If the threshold is greater, the face pair is regarded as from the same person, otherwise from a different person. There is a similar threshold in open-set face identification. Most face recognition methods don't consider the threshold in their optimization process explicitly. So there exists an optimization gap in their methods.

Data augmentation is a very common preprocessing step for CNN based method, as a CNN model contains millions of parameters and is prone to overfitting. Most face recognition methods align faces in both training and testing phases. It seems contradictory to apply data augmentation after face alignment. In this paper, we replace face alignment in the training phase with aggressive data augmentation. Surprisingly, similar accuracy is achieved on LFW benchmark with or without face alignment during testing, which is different from prior results.

3. Literature Survey

Owing to deep learning, lots of breakthroughs have been made in recent years in face recognition. Some researchers train a face feature extractor by employing classification loss. then uses weighted distance as a face verification metric which is trained using a linear SVM. Other research reduces the feature dimension to 150 by PCA and learns a Joint Bayesian model with the features. Some tune the extracted feature for verification in Euclidean space by using a metric learning method with a triplet loss training scheme. In order to develop more effective feature representations, train the feature extractor with joint classification and verification loss. A research proposes a new supervision signal called center loss. By combining classification loss and center loss, they train a robust CNN to obtain discriminative features. In addition to the preceding two stages methods, employs an end-to-end learning process which is the same as ours. It directly learns an embedding into an Euclidean space for face verification by triplet loss.

Most face recognition methods align faces in both training and testing phases. Several complex face alignment methods have been developed, e.g. 2D similarity transformation, 3D alignment, frontalization. Some found that using 2D alignment on training data only provides slightly or no performance improvement but performing 2D alignment on testing images does improve some performance. One of the methods augments data by random cropping and flipping, but most other methods don't as they have already aligned the face images.

Domain adaptation aims to transfer knowledge between related sources and target domains whose distributions are different. Many domain adaptation (or transfer learning) approaches have been proposed for computer vision applications. They learn the features on the large-scale ImageNet dataset in a supervised setting first, and then transfer them to different tasks with different labels. The key idea is that the internal layers of CNN can act as a generic extractor of image representation, which can be pre-trained on one dataset e.g. ImageNet. One of the papers proposes a novel double-path deep domain adaptation network to model the data of clothes images from constrained and unconstrained conditions jointly. There are several prior works which apply domain adaptation to face recognition.

4. Implementation

Facial Recognition using OpenCV

Algorithm – K-NN algorithm for face recognition (with a small modification to throw out nearest neighbor votes whose distance was above a threshold).

Accuracy- The face recognition model OpenCV uses to compute the 128-d face embeddings comes from the OpenFace project.

The OpenFace model will perform better on faces that have been aligned. Face alignment is the process of:
1. Identifying the geometric structure of faces in images.
2. Attempting to obtain a canonical alignment of the face based on translation, rotation, and scale.

Face detection is more accurate using DLIB while SVM is better in the classification of faces. One of the approaches for better accuracy is using the dlib face recognition embeddings and then training a SVM or Logistic Regression model on top of the embeddings.

We are using 128-d face embeddings for each image in our dataset. Load the pre-computed encodings + face names and then construct the 128-d face encoding for the input image. Load our pickled encodings and face names from the disk. We'll need this data later during the actual face recognition step. Convert the input image to rgb color channel ordering. We attempt to match each face in the input image (encoding) to our known encodings dataset (held in data["encodings"]) using face_recognition.compare_faces.

- Internally, the compare_faces function is computing the Euclidean distance between the candidate embedding and all faces in our dataset:
    a. If the distance is below some tolerance (the smaller the tolerance, the more strict our facial recognition system will be) then we return True , indicating the faces match.
    b. Otherwise, if the distance is above the tolerance threshold we return False as the faces do not match.

Given our matches list we can compute the number of "votes" for each name (number of True values associated with each name), tally up the votes, and select the person's name with the most

corresponding votes:

- We begin looping over the detected face bounding boxes and predicted names . To create an iterable object so we can easily loop through the values, we call zip(boxes, names) resulting in tuples that we can extract the box coordinates and name from.
- We use the box coordinates to draw a green rectangle.
- We also use the coordinates to calculate where we should draw the text for the person's name followed by actually placing the name text on the image. If the face bounding box is at the very top of the image, we need to move the text below the top of the box, otherwise the text would be cut off.

Results-

Fig. 1

Deep face recognition with Keras, Dlib and OpenC.

In this deep CNN is used to extract features from input images. It follows the approach described in with modifications inspired by the OpenFace project. Keras is used for implementing the CNN, Dlib and OpenCV for aligning faces on input images. Face recognition performance is evaluated on a small subset of the LFW dataset which you can replace with your own custom dataset e.g. with images of your family and friends if you want to further experiment with this notebook. After an overview of the CNN architecture and how the model can be trained, it is demonstrated how to:

- Detect, transform, and crop faces on input images. This ensures that faces are aligned before feeding them into the CNN. This preprocessing step is very important for the performance of the neural network.
- Use the CNN to extract 128-dimensional representations, or *embeddings*, of faces from the aligned input images. In embedding space, Euclidean distance directly corresponds to a measure of face similarity.
- Compare input embedding vectors to labeled embedding vectors in a database. Here, a support vector machine (SVM) and a KNN classifier, trained on labeled embedding

vectors, play the role of a database. Face recognition in this context means using these classifiers to predict the labels i.e. identities of new inputs.

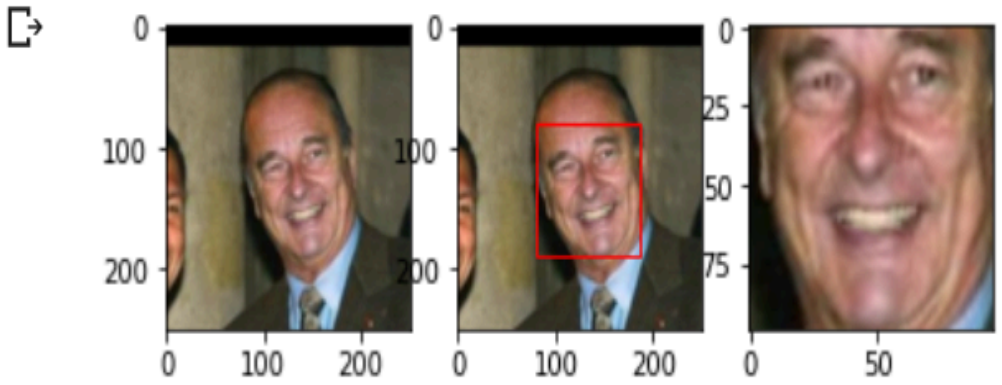

Fig. 2 Landmark indices OUTER_EYES_AND_NOSE are required for model nn4.small2.v1.

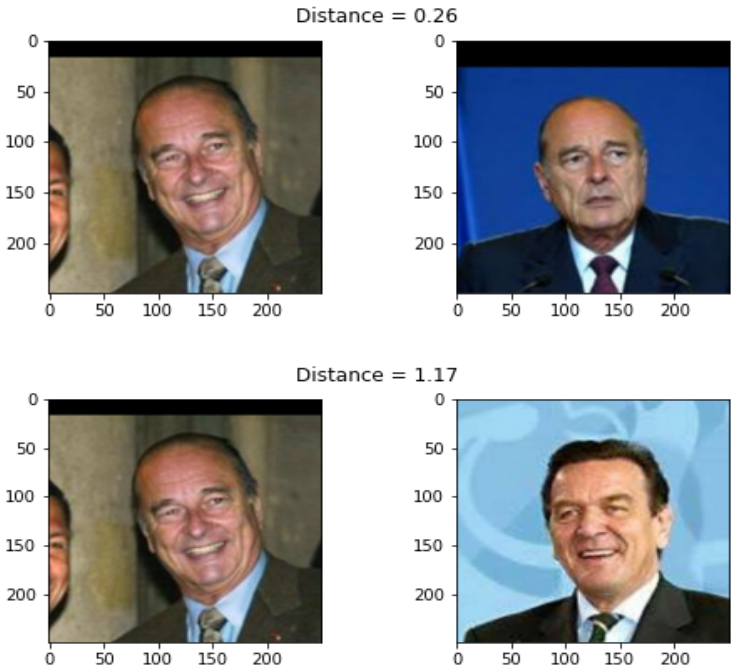

Fig. 3 As expected, the distance between the two images of Jacques Chirac is smaller than the distance between an image of Jacques Chirac and an image of Gerhard Schröder (0.30 < 1.12). But we still do not know what distance threshold τ is the best boundary for making a decision between the same *identity* and *different identity*.

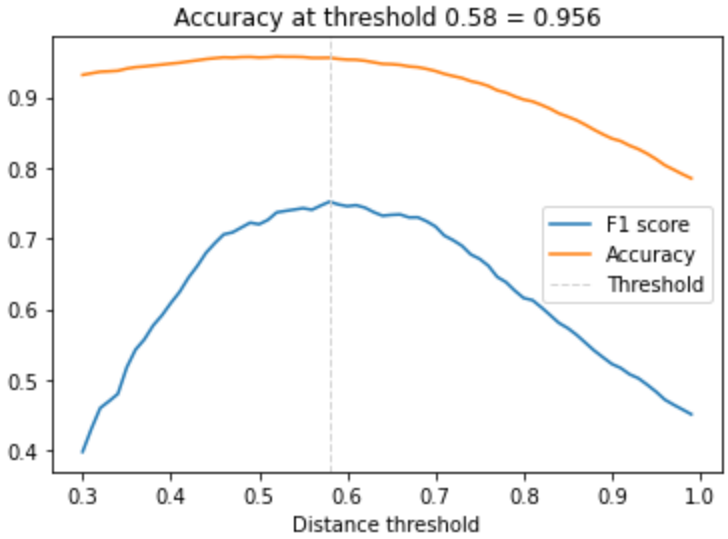

Fig. 4 The face verification accuracy at $\tau = 0.58$ is 95.6%. This is not bad given a baseline of 89% for a classifier that always predicts *different identity* (there are 450 pos. pairs and 4500 neg. pairs) but since nn4.small2.v1 is a relatively small model it is still less than what can be achieved by state-of-the-art models (> 99%).

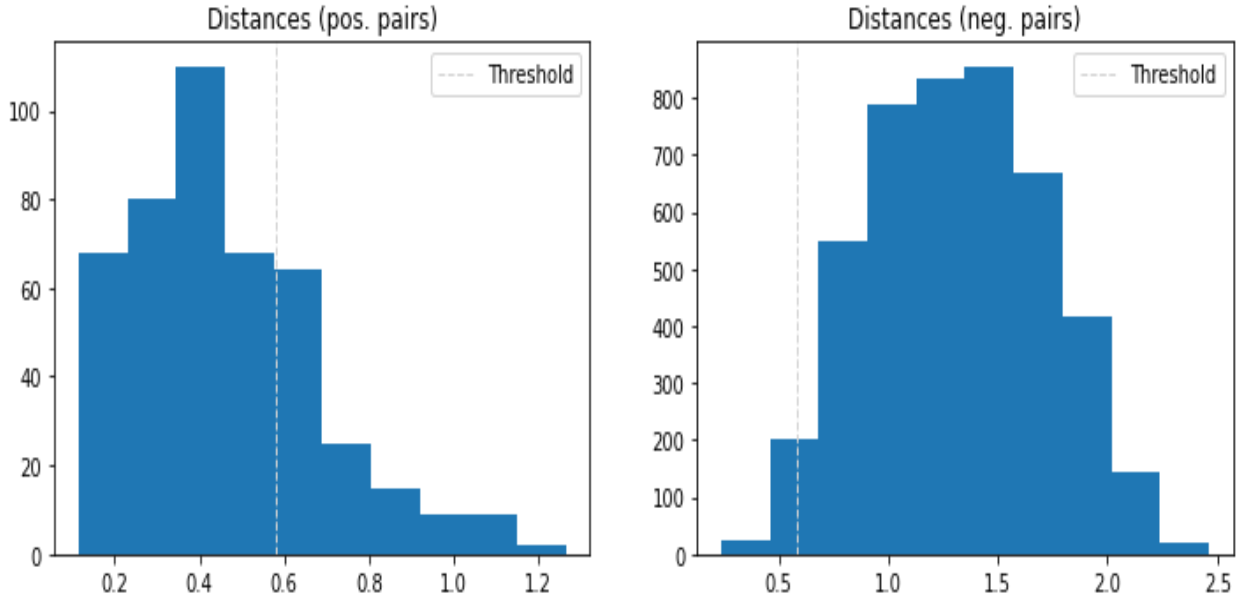

Fig. 5 The two histograms show the distance distributions of positive and negative pairs and the location of the decision boundary. There is a clear separation of these distributions which explains the discriminative performance of the network. One can also spot some strong outliers in the positive pairs class but these are not further analyzed here.

```
29
30 acc_knn = accuracy_score(y_test, knn.predict(X_test))
31 acc_svc = accuracy_score(y_test, svc.predict(X_test))
32
33 print(f'KNN accuracy = {acc_knn}, SVM accuracy = {acc_svc}')

KNN accuracy = 0.96, SVM accuracy = 0.98
```

The KNN classifier achieves an accuracy of 96% on the test set, the SVM classifier 98%. Let's use the SVM classifier to illustrate face recognition on a single example.

Fig. 6 KNN vs. SVM

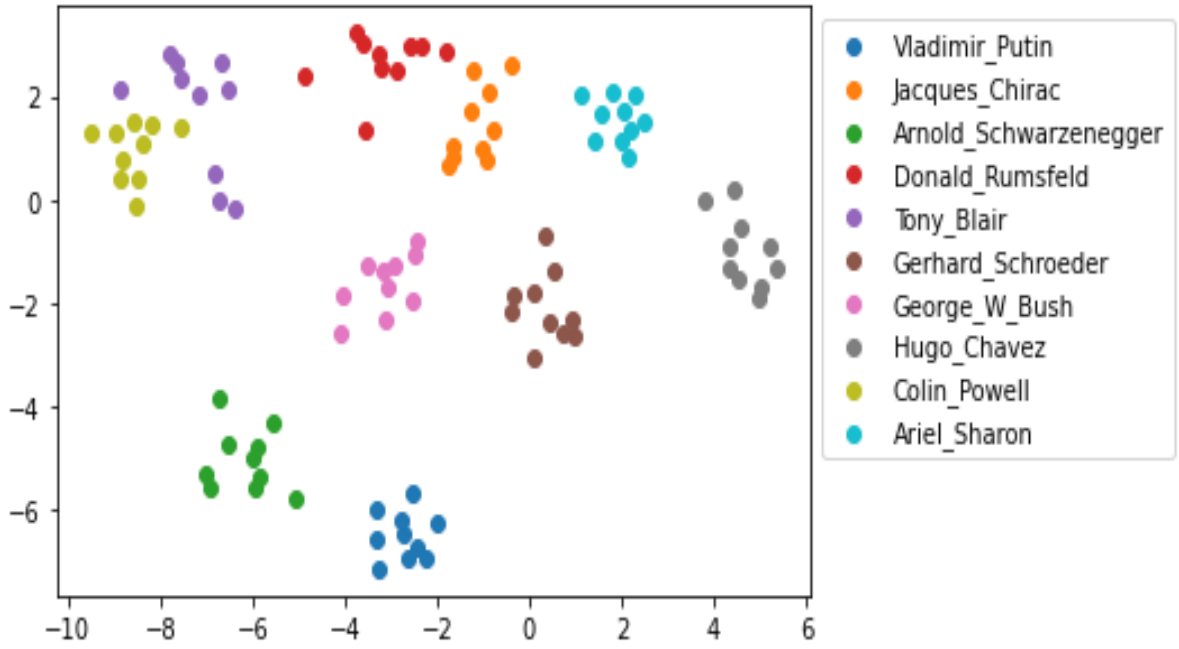

Fig. 7 Data Visualization

Deep Domain Adaptation Network for Face Recognition

In this, we adopt unsupervised transfer learning methods to address this issue. To alleviate the discrepancy between source and target face database and ensure the generalization ability of the model, we constrain the maximum mean discrepancy (MMD) between source database and target database and utilize the massive amount of labeled facial images of source database to training the deep neural network at the same time. We evaluate our method on two face recognition benchmarks and significantly enhance the performance without utilizing the target label.

All the top performing methods for face recognition were all based on DCNN architectures. Facenet presented a triplet embedding loss to learn a mapping from face images to a compact Euclidean space. The loss aimed to separate the positive pair from the negative by a distance margin. VGG-face model is a typical application based on VGG architectures. It was trained on a large scale dataset of 2.6M images from 2622 subjects. There was a proposal of an angular softmax loss to learn angularly discriminative features on a hypersphere manifold. These methods were all focused on utilizing a massive amount of labeled facial images to train a DCNN of strong generalization ability and testing on common benchmarks. All these approaches obtained excellent performance on many benchmarks e.g. LFW, YTF, MegaFace, which were also collected from the Internet like the training data.

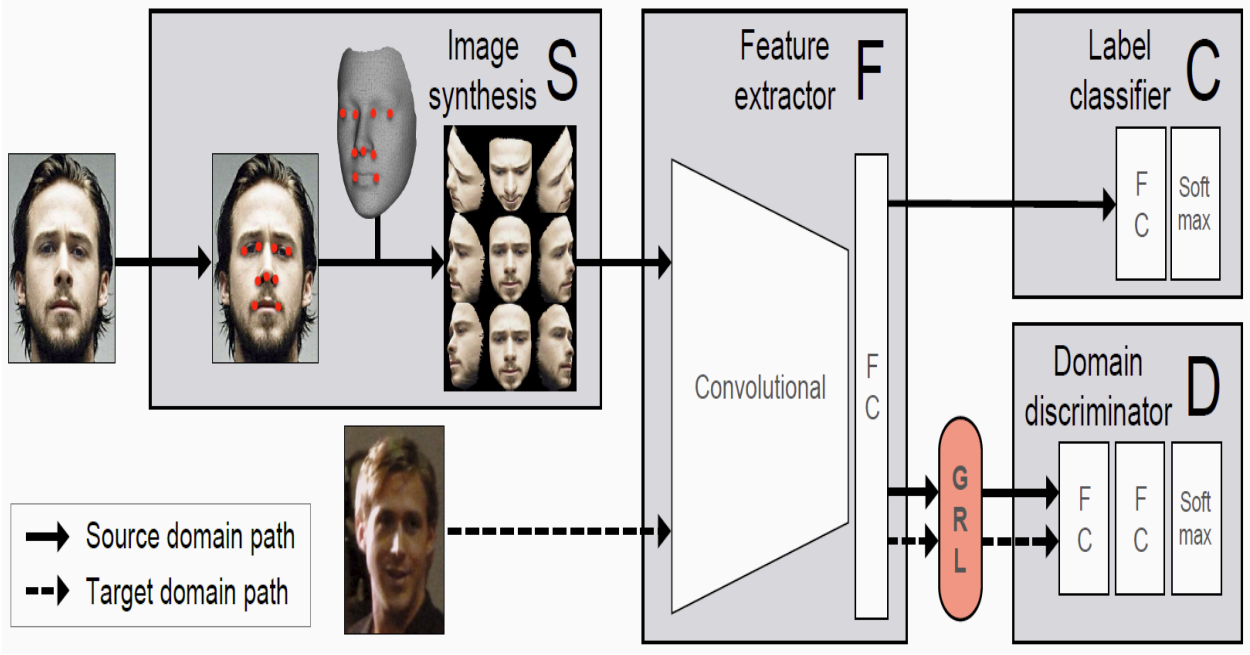

Fig. 8 Deep Domain Adaptation Network for Face Recognition with Single Sample Per Person

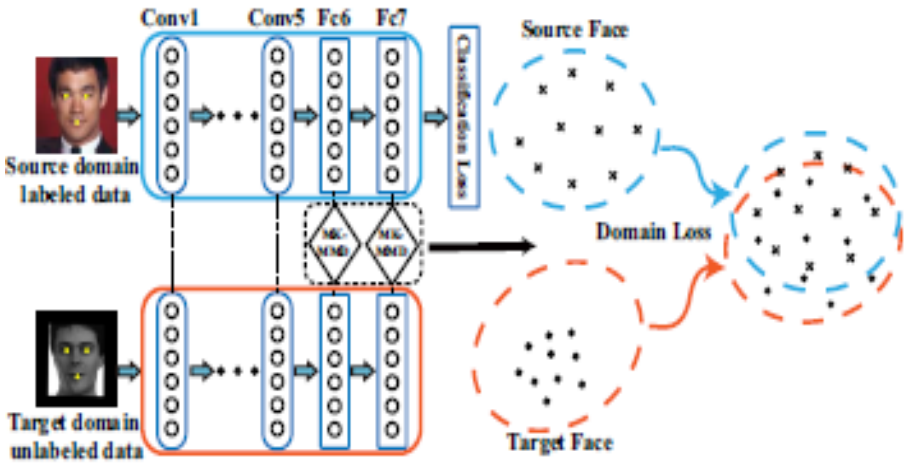

Fig. 9 The joint deep neural network architecture for unsupervised domain adaptation. The inputs of the upper network are source labeled images while the lower are target unlabeled data. All the source face and target face are aligned to the same reference point(the yellow dots in the facial images). The blue dotted oval represents the source domain distribution and the crosses inside donate source face. The orange dotted oval represents the target domain distribution and the dots inside donate the target face. Domain loss aims at minimizing the distribution discrepancy of two domains.

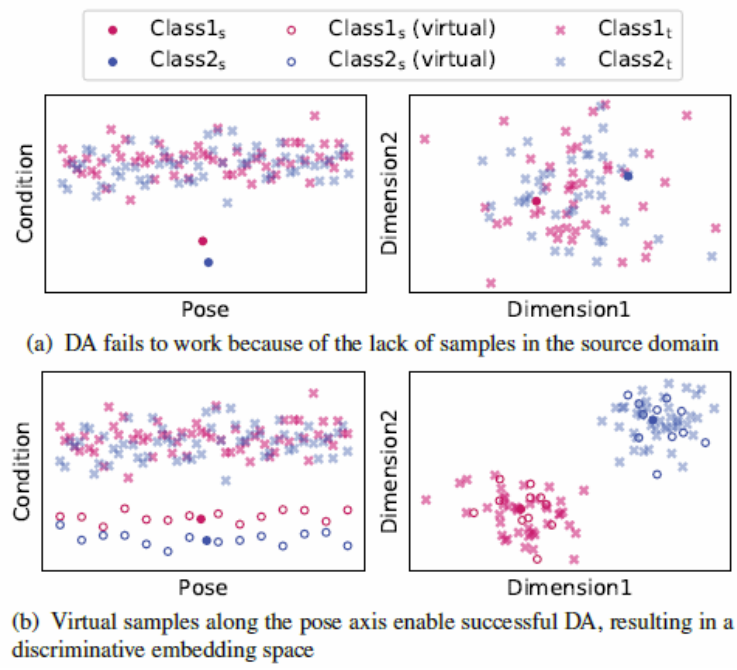

Fig. 10 Facial feature space (left) and its embedding space after applying DA (right). The subscript "s" and "t" in the legend refer to the source and target domains, respectively.

Author	Processing	Classification	Number of Categories	Number of Subjects	Performance
Mase	optical flow	kNN	4	1	86%
Black & Yacoob	parametric model	rule-based	6	40	92%
Yacoob & Davis	optical flow	rule-based	6	32	95%
Rosenblum et al.	optical flow	neural networks	2	32	88%
Essa & Pentland	optical flow	distance-based	5	8	98%
Otsuka & Ohya	2D FT of optical flow	HMM	6	4	93%
Lanitis et al.	appearance model	distance-based	7	-	74%
Chen	appearance model	Winnow	6	5	86%
Cohen et al.	appearance model	Bayesian networks	7	5+53	83%

Fig. 11 Comparison of performances of different algorithms.

5. REFERENCES (Research Papers)

1. Facial recognition using OpenCV https://core.ac.uk/download/pdf/233567034.pdf
2. Facial recognition with OpenCV https://www.bytefish.de/pdf/facerec.pdf
3. G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.
4. M. Kan, J. Wu, S. Shan, and X. Chen. Domain adaptation for face recognition: Targetize source domain bridged by common subspace. International journal of computer vision, 109(1-2):94–109, 2014.
5. O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In British Machine Vision Conference, volume 1, page 6, 2015.

6. Datasets

Data set link -
https://drive.google.com/drive/folders/1uC7vmzYAT1nQc0TKoRQBwB6Oa55DWlR_?usp=sharing

https://drive.google.com/drive/folders/1TfO3NndJtFUGkLweDrmpmCtmjFicQsBx?usp=sharing