ENSEMBLE OF CONVOLUTIONAL NEURAL NETWORKS BASED FEATURE DESCRIPTORS

Sneha D P[1] and Dr. Vasudev T[2]
Research Scholar, Department of Computer Science, University of Mysore, Mysuru, Karnataka, India

Professor, Department of MCA, Maharaja Institute of Technology Mysore, Karnataka, India

Abstract

In many face recognition applications, due to the small sample size (SSS) problem, it is difficult to construct a single strong classifier. However, ensemble classifiers are gaining significance mainly in face recognition systems due to its ability to overcome the SSS problem. Hence this chapter proposed a novel Ensemble of Convolutional Neural Networks based feature descriptors for face recognition in uncontrolled environments.

Keywords: SSS, CNN, Inception V3, VGG16, VGG19

1. INTRODUCTION

In recent years, machine learning has reached its pinnacle in automation and deep learning started achieving success in numerous research areas of Computer Vision. As Data became Big Data, traditional CPUs are getting replaced by powerful GPUs for computationally intensive applications. The need to use Deep Learning systems is of utmost importance in various domains such as Medical Image Analysis, Face Recognition, Robotics, Self-driving Cars etc. to achieve better results. Before a decade, traditional feature extraction methods such as Local Binary Pattern (LBP) by Ojala et al. (2015), Scale Invariant Feature Transform (SIFT) by David Lowe, Histogram of Oriented Gradients (HoG) by Dalal and Triggs achieved good results in publicly available datasets. It was mainly due to the use of Bag of Visual Words approach along with a Machine Learning classifier such as Linear SVM as shown by Yang et al. Recent research works in the Machine Learning community showed that automatic learning of these features in raw images is possible if multiple layers of nonlinear activation functions are used. This led to the introduction of a Neural Networks, namely the Convolutional Neural Networks (CNN) which was first applied on a larger dataset. CNN has attracted the Computer Vision (CV) research by significantly improving the state of the art application domains. The key success of using CNN in CV applications is due to its scalable quantities of processing speed, power in accuracy and the massive training dataset. However, deep learning techniques are far better than the conventional machine learning techniques. Technology giants like Google, DeepMind and Facebook, etc., are already making a huge stride in the CV space. In olden days Facebook asked you to tag your friends in your photograph, but the advancement of deep learning techniques leads the ability to recognize your friends from your group photograph. Facebook algorithms are efficient enough to recognize the faces with 98% accuracy from the posted group/individual photographs. The recent face recognition method by Google also uses nearly about 200 million face images which are larger than any other publicly available dataset. Hence, processing these large datasets is a complex task and requires high computational devices like Graphics Processing Units (GPU). Face recognition in an uncontrolled environment is still a challenging task. The recent study on Face recognition under uncontrolled environment reveals the practical difficulties of implementing. The conventional face recognition algorithms use the low-level features and shallow models to represent the faces and facial features. In this decade, many authors have proved the effectiveness of the deep learning models in effective face recognition. Recent deep learning models such as Alexnet, ZFNet, VGG, GoogleNet, Inception etc., are effectively utilized to extract high level visual features. In contrast to the above stated models this research work proposes a novel ensemble of deep learning models for an efficient face recognition. An ensemble is a finite collection of models that can be used to obtain better average predictive accuracy than using a single model in the ensemble collection. This research work also addresses the existing CNN model to extract the features for face matching. The existing handcrafted feature descriptors such as Local Binary Patterns (LBP), Speeded Up Robust Features (SURF) or Histogram of Oriented Gradients (HoG), which requires domain level understanding of the face recognition problem. In this research work an Ensemble of Convolutional Neural Network (ECNN) based feature descriptors for face recognition is proposed to overcome the challenges namely, facial expression, aging and pose, illumination and low resolution.

2. Related works

Face detection algorithms aim to locate the main face area in input images or video frames. Furthermore, they help robots discriminate between humans and other objects in the scene. Before the deep learning era, the cascade-based methods and deformable part models (DPM) dominated the face detection field with limitations in unconstrained face images due to considerable variations in resolutions, illumination, expression, skin color, pose, and occlusions [1]. In recent years, deep learning methods have shown their power in computer vision and pattern recognition. As a result, many deep convolutional neural networks (CNN or DCNN)-based face detection methods have been proposed to overcome the limitations mentioned above [3,2,3,4,5,6]. The CNN-based face detection approaches generally have two stages: a feature extraction stage by utilizing a CNN-backbone network to generate the feature map, and a stage for predicting the bounding box locations [**15**]. They can be divided into two categories: (1) multi-stage; and (2) single-stage detection algorithms.

Two-stage algorithms: Most two-stage algorithms are typically based on Faster R-CNN [12] and generate several candidate boxes and then refine the candidates with a subsequent stage. The first stage utilizes a sliding window to propose the candidate bounding boxes at a given scale, and the second stage rejects the false positives and refines the remaining boxes [16,17,18]. The advantage of this type of model is that they reach the highest accuracy rates, on the other hand they are typically slower.

Single-stage algorithms: Most single-stage algorithms are typically based on the single shot multi-box detector (SSD) [11]. These algorithms treat object detection as a simple regression problem by performing the candidate classification and bounding box regression from the feature maps directly in only one stage, without the dependence on an extra proposal stage [3,13]. The advantage of this type of model is that they are much faster than two-stage algorithms, but they have lower accuracy rates. Among the many variants using the single-stage structure, state-of-the-art face detection performance was achieved by RetinaFace [3]. RetinaFace is the latest one-stage face detection model, which is based on the structure of RetinaNet [19] and uses deformable convolution and dense regression loss. We utilized the lightweight version of RetinaFace based on the mobilenet backbone to enhance the detection speed to achieve real-time performance. Face alignment plays a vital role in many computer vision applications. It is necessary to improve the robustness of face recognition against in-plane rotations and pose variations [20]. Meanwhile, facial landmarks are essential for most existing face alignment algorithms because they are involved in the similarity transformation for finding the closest shape of the face. So, facial landmark localization is a prerequisite for face alignment.

Face alignment aims to identify the geometric structure of the detected face and calibrate it to the canonical pose, i.e., determining the location and shape of the face elements, such as the mouth, nose, eyes, and eyebrows. From an overall perspective, face alignment methods can be divided into model-based and regression-based methods [21]. However, the regression methods show superior accuracy, speed, and robustness when compared to model-based methods [22]. Furthermore, model-based methods show difficulties to express the very complex individual landmark appearance. Trigeorgis et al. [23] further optimize regression-based methods by introducing a single convolutional recurrent neural network architecture that combines all stages' training through facilitating a memory unit that shares information across all levels. The importance of the initialization strategies for face alignment is demonstrated in [24]. Despite that, Valle et al. [25] handled the sensitivity problem of initialization strategies by introducing the Deeply-initialized Coarse-to-Fine Ensemble (DCFE) approach. DCFE refines a CNN-based initialization stage with Ensemble of Regression Trees (ERT) to estimate probability maps of landmarks' locations. Cascade of experts is used by Feng et al. in [26] to improve the face alignment accuracy versus the different face shape poses. Feng et al. proposed Random Cascaded Regression Copse (R-CR-C) method that utilizes three parallel cascaded regressions. Furthermore, Zhu et al. [27] used a probabilistic approach to adopt coarse-to-fine shape searching. There have been significant improvements in face alignment using deep learning methods. As in [28], Kumar and Chellapa introduced a single dendritic CNN, termed the Pose Conditioned Dendritic Convolution Neural Network (PCD-CNN). Furthermore, they combine a classification network with a second and modular classification network to predict landmark points accurately. In addition, Wu et al. [28] proposed a boundary-aware face alignment algorithm that interpolates the geometric structure of a human face as boundary lines to improve landmark localization. In a later work, a more efficient compact model has been recently proposed by Guo et al. named practical facial landmark detector (PFLD) [29]. They used a branch of the network to estimate the geometric information for each face sample to make the model more robust. PFLD achieved a size of 2.1 Mb and over 140 fps per face on a mobile phone with high accuracy against complex faces, including unconstrained poses, expressions, lighting, and occlusions, which makes it more suitable for HRI applications. A face recognition system is a system that can identify or verify a person in an input image or a video frame. With the current advances in machine learning, the deep face recognition systems based on the CNN models have been the most common due to their remarkable results, and several deep face recognition models have been proposed [4,30,31,32,33,34]. These models work by localizing the face in the input image, extracting the face embeddings, and comparing them to other face embeddings pre-extracted and stored in a database. Every embedding creates a unique face signature and the identity of a specific human face. Taigman et al. proposed a multi-stage approach called DeepFace [30] based on AlexNet architecture [35]. The faces are first aligned to a generic 3D shape model, and then facial representation is derived from a nine-layer deep neural network. In addition, the authors used a Siamese network trained by standard cross-entropy loss for face verification. Inspired by the work of DeepFace, Sun et al. introduced a high-performance deep convolutional neural network called DeepID2+ [36] for face recognition. DeepID2+ achieved a better performance by adding supervision to early convolutional layers and increasing the dimension of hidden representations. Schroff et al. proposed FaceNet [31] based on the GoogleNet architecture [37]. FaceNet directly optimizes the face embedding by a deep convolutional network trained using a triplet loss function at the final layer. He et al. proposed a Wasserstein convolutional neural network (WCNN) approach [38] that optimizes face recognition by learning invariant features between near-infrared and visual face images. Recently, different loss functions for face recognition have been proposed [4,32,33,39,40] to enhance discriminative feature learning and representation. Sphereface presents the importance of the angular margin and its advantage in feature separation, but the training is unstable and hard to converge. CosFace defines the decision margin in the cosine space by directly adding the cosine margin penalty to the target logit, which results in better performance than SphereFace with easier implementation and stable training. The ArcFace or Additive Angular Margin Loss [4] is one of the most potent loss functions designed for deep face recognition [41,42,43]. It enhances discriminative learning by introducing an additive angular margin. In contrast with SphereFace and CosFace which have a nonlinear angular margin, ArcFace has a constant linear angular margin. The evaluation of single face recognition requires high computational power. Furthermore, multiple faces in a single scene need to be recognized and identified in practice. This makes recognizing multiple faces another challenge, as it requires more computing power to process multiple faces per scene. The accuracy and processing time are the main criteria for any face recognition system. Nevertheless, especially for the HRI, accuracy and real-time recognition are a challenge in scenes with subjects that do not co-operate with the recognition system. Visual object tracking has always been a research hotspot in computer vision, and face tracking is a special case. Face tracking is primarily a process of determining the position of the human face in a digital video or frame based on the detected face. This is challenging as the face is not the same during the time (video frames), but it may vary in pose and view. Moreover, other factors affected the face tracking in the actual scene and made it more complex, such as illumination, occlusion, and posture changes. On the other hand, face tracking has many advantages, such as counting the number of human faces in a digital video or camera feed and following a particular face as it moves in a video stream to predict the person’s path or direction.

3. CONVOLUTIONAL NEURAL NETWORKS (CNN)

CNN consists of three layers, namely convolution, max-pooling and fully connected layers. The CNN architecture is formed by stacking these layers sequentially. CNN applies multiple filters to the raw input image to extract the high level features. CNN transform the original image layer by layer from the original pixel values to the final class scores. Figure 1. shows the convolution layer with the set of filters.

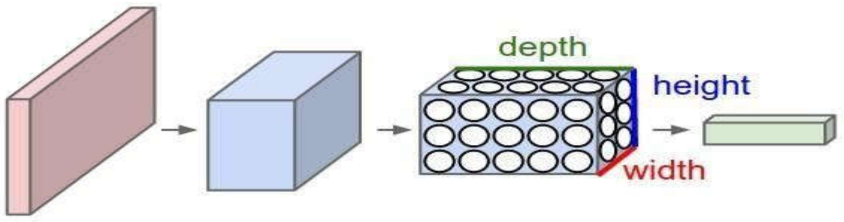

Figure 1: Convolution Layer

3.1 Convolution (Conv) Layer

The convolution layer’s parameters consist of a set of learnable filters. The size of every filter is smaller than the input width and height of volume. Slide each filter across the width and height of the input volume and compute the dot products between entries of the filter and the input at any position. It will produce a 2-dimensional activation map that gives the responses of that filter at every spatial position. The network will learn the filters that activate when they see some type of visual feature. In the first convolution layer it detects the edges from raw pixel data. In the second convolution layers it detects shapes or blobs from the previous layer output. In the subsequent convolution layer, it detects the high level features such as facial structures from the last layer. The last layer in CNN is a classifier that uses these higher-level features to make predictions.

Convolution Layer accepts a volume size of W1*H1*D1 and requires four hyper parameters such as number of filters K, filter size F, Stride S and amount of zero padding P. The output volume size of the single convolution layer is W2*H2*D2.

Where,

$W_2 = ((W_1 - F + 2P)/S) + 1$ $H_2 =$

$((H_1 - F + 2P)/S) + 1$ $P = (F-1)/2$

$D_2 = K$

3.2 Max-pooling Layer

The Max-pooling layer is inserted in between the Convolution layers in CNN architecture. The Max-pooling layer is used to reduce the spatial size of Convolution image. The Max-pooling layer operates independently on every depth slice of the input and it resizes the image using MAX operation. The pooling layer accepts input volume size of W1*H1*D1 and it requires two hyper parameters such as Filter Size F and Stride S. It produces the output volume size of W2*H2*D2. Figure 2 shows the max pooling operation with stride 1 and 2.

where,

$W_2 = (W_1 - F)/S + 1$ $H_2 =$

$(H_1 - F)/S + 1$ $D_2 = D_1$

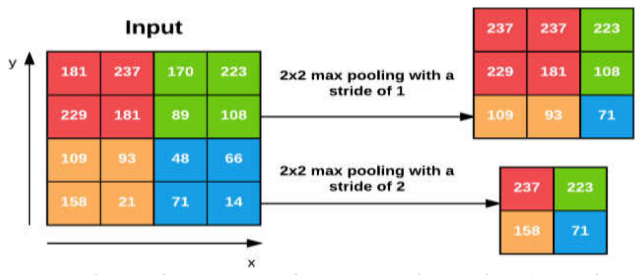

Figure 2: Max pooling layer with stride 1 and 2

3.3 Fully Connected Layer

Fully connected layers are used at the end of the network after feature extraction and consolidation has been performed by the convolution and pooling layers. They are used to create final nonlinear combinations of features and for making predictions by the network.

3.4 Transfer learning

Transfer learning is the process of taking a pre-trained model (the weights and parameters of a network that has been trained on a large dataset) and “fine-tuning” the model with own data set. The idea is that this pre-trained model will act as a feature extractor by removing the last layer of the network and replace it with your own classifier**.**

3.5 Ensemble Methods

Ensemble methods generally refer to training a “large” number of models and then combining their output predictions via voting or averaging to yield an increase in classification accuracy. Ensemble methods are specific to deep learning and Convolutional Neural Networks. Multiple networks are trained and each network returns the probability for each class label. These probabilities are averaged and the final classification is obtained.

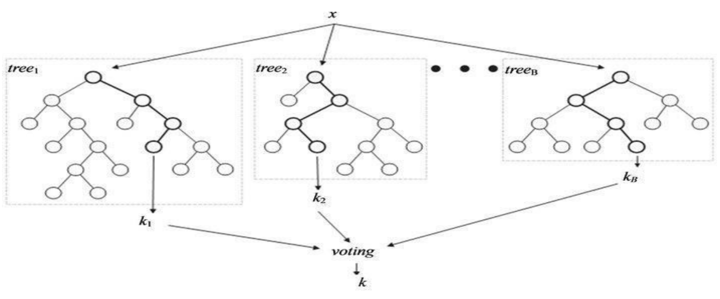

Figure 3: Random Forest consists of multiple decision trees

Figure 3 shows the Random Forest consists of multiple decision trees. The outputs of each decision tree are averaged together to obtain the final classification.

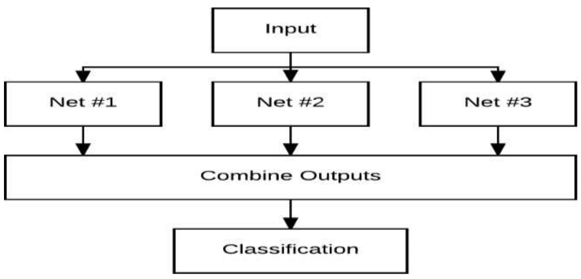

Figure 4: Ensemble of neural networks

Figure 4 shows the ensemble of neural networks consists of multiple networks. When classifying an input image, the data point is passed to each network where it classifies the image independently of all other networks. The classifications across networks are then averaged to obtain the final prediction.

4. Convolution Neural Network for Feature Extraction

Convolution Neural Networks applies multiple filters on the raw input image for extracting high level features. Figure 5 shows the extraction of features, where x represents input face image and f(x) shows the extracted features.

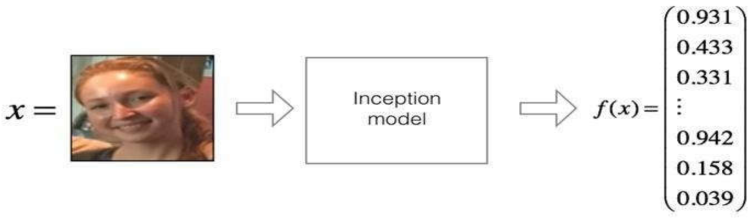

Figure 5: ConvNet feature extraction

The similarity of two face images is computed with the extracted features using Euclidean distance metric algorithm. Therefore, the encodings of two images of the same person are similar to each other and the encodings of two images of different persons are different. Figure 6.6 shows the comparison of two face images of the same person.

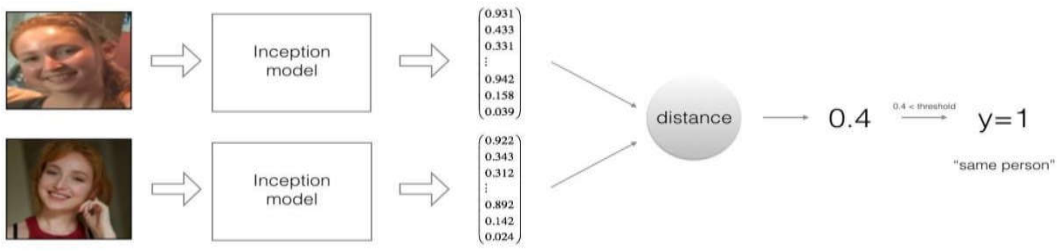

Figure 6: Comparison of two face images

4.1 Feature Extraction using a Pre-trained CNN

The VGG16 network architecture produces the probabilities for the input 1,000 ImageNet class labels. When treating networks as a feature extractor, the network at an arbitrary point is chopped off. Now the last layer in the network is a max pooling layer which will have the dimension of 7*7*512. Thus achieving 25088 feature values for each face image. The above discussed process is repeated for an entire dataset resulting the total size of N images, each with 25,088 columns. The extracted feature vectors are trained using Logistic Regression classifier. Figure 7 shows the architecture of VGG 16 CNN model. The inception model in GoogleNet proposed a new way to perform convolution on the previous layer by performing multiple convolutions (which are computed in parallel over the input volume) which gets concatenated at the output volume. Instead of using [3x3] and [5x5] filter sizes directly on the input volume (which greatly increases the depth of output volume), dimensionality reduction (which could also be thought of as pooling of features) is performed using [1x1] convolution filters, so that the overall depth at the output volume does not get increased at a higher rate. Thus, the inception module holds a smaller filter convolution, a medium filter convolution, a large filter convolution and a pooling operation performed in parallel on the input volume which learns extremely fine grain details, higher level details and combats overfitting (due to the presence of pooling operation). In addition to this, the presence of Rectifier Linear Unit (ReLu) non-linear activation after each convolutional layer enhances the performance.

The incepton-v3 architecture does not have fully connected layers at the top, instead it uses “average pool” operation which greatly reduces the learnable parameters involved. Thus, instead of stacking layers in a CNN sequentially (in the case of VGGNet and OverFeat network), GoogLeNet showed a different type of deep architecture such as the “Inception” module (network in a network) which highly contributes to achieve better results. Figure 8 shows the inception module of GoogleNet architecture.

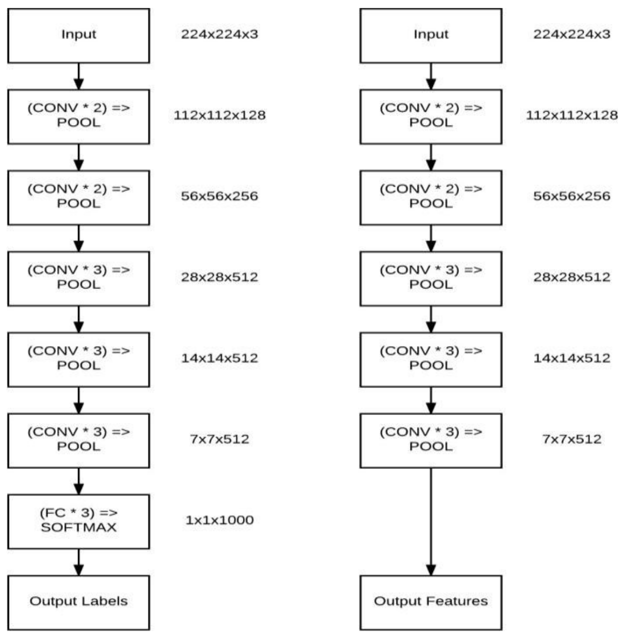

Figure 7: VGG 16 network architecture

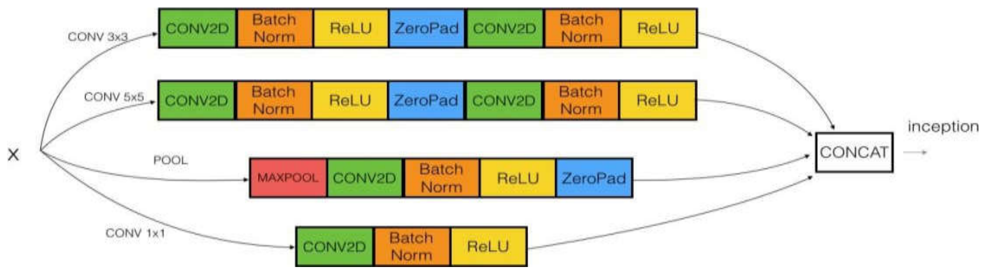

Figure 8: Inception module of GoogleNet

4.2 Ensemble of Convolutional Neural Networks

The proposed ECNN method extracts the features from three different ConvNets models, namely VGG16, Incpetion-v3 and Xception. The pre-trained Convets models are used for extracting features because of weights and architectures of these models are publicly made available for Computer Vision research. The model weights and architecture of the considered models are loaded locally prior to the training phase. The architecture of the proposed ECNN method is presented in Figure 9 and the images in training database are resized to a fixed dimension as shown in Table 1.

After pre-processing, each of the images in the database is given to VGG16, Inception-v3 and Xception model architecture for extracting features by removing the top fully-connected layers of the ConvNets model. The extracted features of the three different pre-trained models of a single image is concatenated and stored in a list. Its corresponding label is also stored in another list. This process is repeated for all the images in the training database. The extracted image features and labels are stored locally in an HDF5 file format as NumPy arrays. This feature extraction process is carried out for two different benchmark Face datasets such as YouTube and WebFace. VGG16 ConvNets model accepts the input image size of 224*224 and it has 13 convolution layers with different number of filter sizes such as 64,128, 256, 512. The size of each filter is 3*3. Along with this the VGG16 network poses five max pooling and three fully connected layers. The size of the feature vector extracted for VGG16 before the last fully connected network is 4096. Inception-V3 ConvNets model accepts the input image size of 299*299, inception module extracts multi-level features by performing convolutions with different filter size such as 1×1, 3×3, and 5×5 convolution. The weights for Inception V3 are smaller than VGG and ResNet, of memory size 96MB. Xception is an extension of the Inception architecture which replaces the standard Inception modules with depth wise separable convolutions and it has a small weight memory size of 91MB.

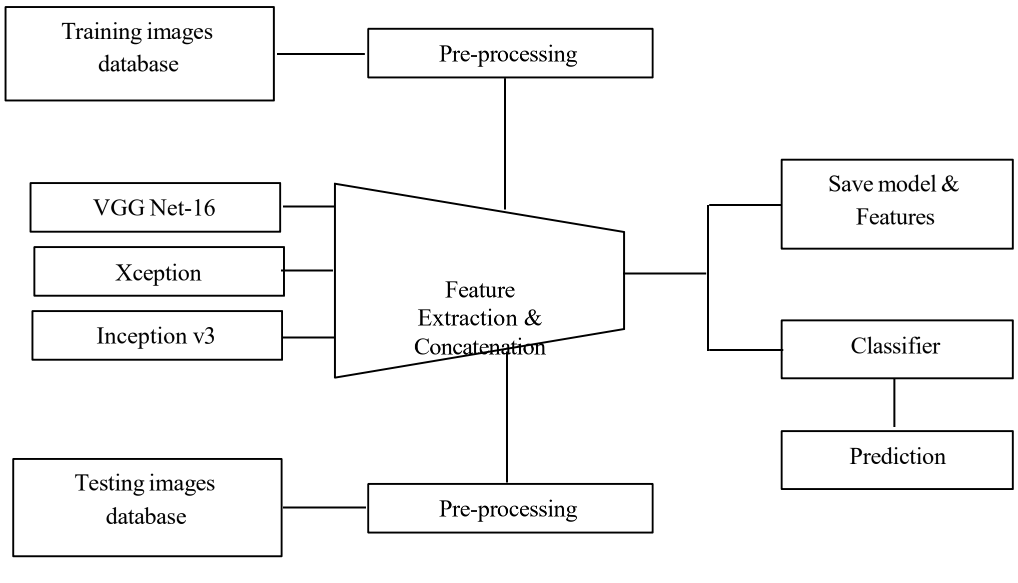

Figure 9: Proposed ECNN architecture for Face Recognition

Table 1: Fixed dimension to resize the images for ECNN method

ConvNets model	Fixed dimension (for pre-processing)
VGG16	224 x 224
VGG19	224 x 224
ResNet50	224 x 224
Inception-v3	299 x 299
Xception`	299 x 299

4.3 Training and Evaluation

After extracting features from the training phase, the stored features and labels are loaded and split into training and testing data based on a parameter 'train_test_split'. If 'train_test_split' is chosen as 0.1, then it means 90% of the overall data is used for training and 10% of the overall data is used to evaluate the trained model. Based on parameter tuning and Grid-search methodology, it is found that Logistic Regression (LR) outperformed all the other machine learning classifiers such as Random Forests (RF), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN). Two performance metrics are chosen to evaluate the trained model, namely Rank-1 and Rank-5 accuracy. Rank-1 accuracy gives the accuracy of the trained model when tested with an unseen test data on the first chance. Rank-5 accuracy gives the accuracy of the trained model when tested it with an unseen data given five chances.

Algorithm

```
Begin
Input: ImageNet data DImageNet = $\{x_i, y_i\}^m$ Output:
model M, Accuracy A, Classifier H xi = $\{x_1, x_2, \dots \dots . x_n\}$
F = $\{f(x_1), f(x_2), \dots . f(x_n)\}$
F- Set of input face image features qi – query face image features.
For feature f=1 to n do
        Learn $F_n$ based on D1..m
        Perform feature matching between database face and query face
        d(p,q)= $\sqrt{\sum_{i=1}^{N} (q_i - x_i)^2}$
        Append all the distance values results = max (d(p,q))
        If results< threshold
            Known person
        Else:
                Unknown person
            End if
    End for End
```

5. Results and Discussion

The experimental setup for the proposed methodology is carried out using Intel Xeon processor with the NVIDIA Titanx GPU and 28GB RAM. Python programming language is used for the overall experiment from data processing, feature extraction, model training till model evaluation. A developed, efficient and modular Deep Learning library for Python called Keras created by François Chollet is used for the overall experiment. The entire experiment is carried out on Windows-7 Operating System (OS) with Theano as backend for Keras. Other python packages are also used for implementing the proposed method are NumPy, SciPy, scikit-image, h5py, scikit-learn and OpenCV 2.4.10.

5.1 Dataset

Two publicly available face datasets are considered for analyzing the performance of the proposed ECNN method. The WebFace dataset contains 494414, color images of 10,575 people with different facial expressions, illumination condition, aging, low resolution, different pose and occlusions. The Figure 10 represents the sample face images of WebFae dataset. The YouTube faces dataset contains 3425 videos of 1595 different people. Figure 11 shows the sample faces images of a YouTube face dataset.

Figure 10: Sample face images of WebFace dataset

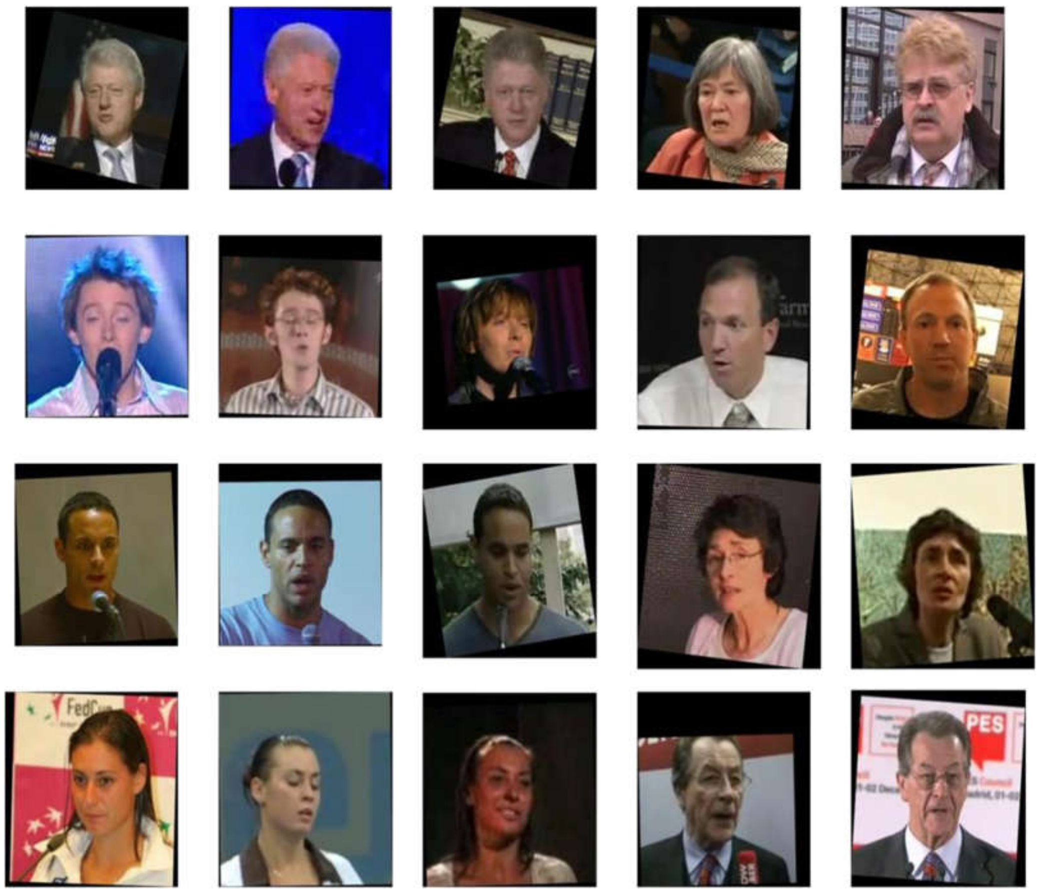

Figure 11: Sample face images of a YouTube face dataset

Table 2: Feature vector dimension of ConvNets model

ConvNets Models	Feature Dimension
VGG16	4096
VGG19	4096
ResNet 50	2048
Inception-v3	2048
Xception	2048

Table 2 represents the feature vector dimension of different pre- trained CNN models. The proposed ECNN method concatenate vgg16, inception-v3 and xception feature vectors which produces the final feature vector of 8192 size. Table 3 presents face recognition accuracy of WebFace dataset. From Table 3 it is observed that the proposed ECNN method performs better face recognition accuracy compared to vgg16, vgg19, inception-v3, resnet50 and xception CNN models. Figure 12 shows the rank 1 and rank 5 face recognition accuracy for webface dataset.

Table 3: Recognition accuracy for Web Face dataset.

ConvNet Models	**Accuracy (%)**	
	Rank-1	**Rank-5**
VGG16 (Simonyan et al) [12]	69.96	88.86
VGG19 (Simonyan et al) [12]	66.25	86.49
ResNet 50 (Kaiming He et al) [16]	29.90	55.25
Inception V3 (Szegedy et al) [15]	78.81	93.91
Xception (François Chollet et al) [17]	65.82	89.02
Inception V3 + VGG16 (proposed)	85.44	96.25
Inception V3 + VGG19 (proposed)	84.01	95.58
Inception V3 + Xception (proposed)	77.30	94.98
Inception V3+ ResNet 50 (proposed)	79.71	94.15
ECNN (proposed)	**87.08**	**97.12**

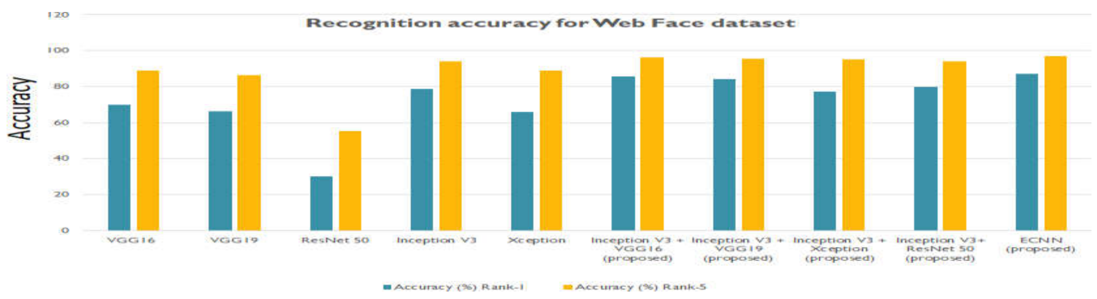

Figure 12: Face recognition accuracy for web face dataset

Table 4: Recognition accuracy of YouTube face dataset.

ConvNet's Models	**Accuracy (%)**	
	Rank-1	**Rank-5**
VGG16 (Simonyan et al) [12]	99.97	99.98
VGG19 (Simonyan et al) [12]	99.97	99.98
ResNet 50 (Kaiming He et al) [16]	99.02	99.79
Inception V3 (Szegedy et al) [15]	99.93	99.97
Xception (François Chollet et al) [17]	99.92	99.98
Inception V3 + VGG16 (proposed)	99.97	100
Inception V3 + VGG19 (proposed)	99.97	100
Inception V3 + Xception (proposed)	99.97	100
Inception V3+ ResNet 50 (proposed)	99.98	100
ECNN (proposed)	**99.99**	**100**

Table 4 shows the recognition accuracy of a YouTube face dataset. From Table 4 it is proved that the proposed ECNN method outperforms vgg16, vgg19, inception-v3, resnet50 and xception CNN models.

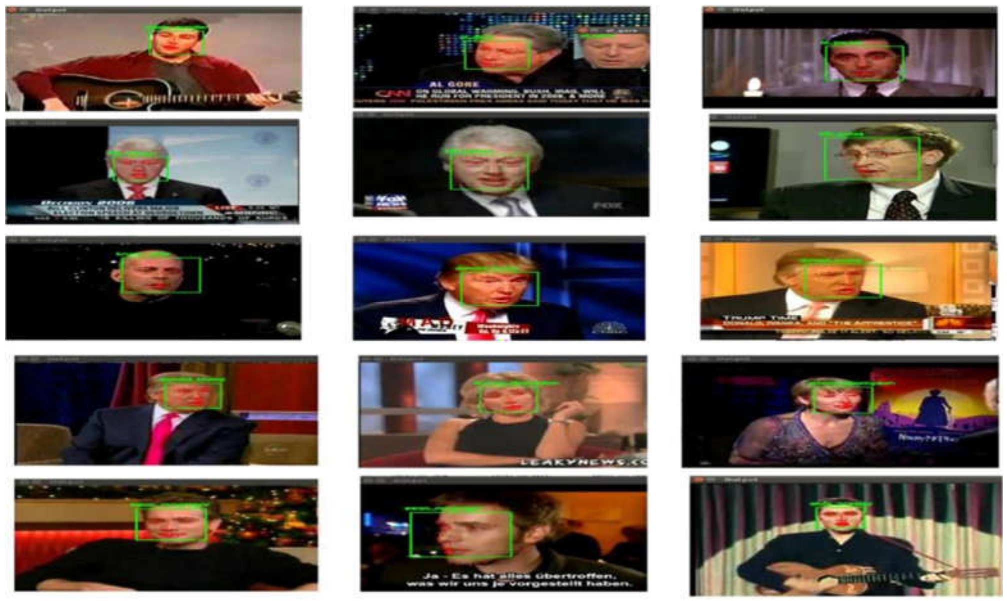

Figure 13 Face Recognition results for YouTube Faces Dataset

Figure 13 shows the face recognition results for YouTube Face dataset and Figure 14 shows face recognition results for MIT-India dataset.

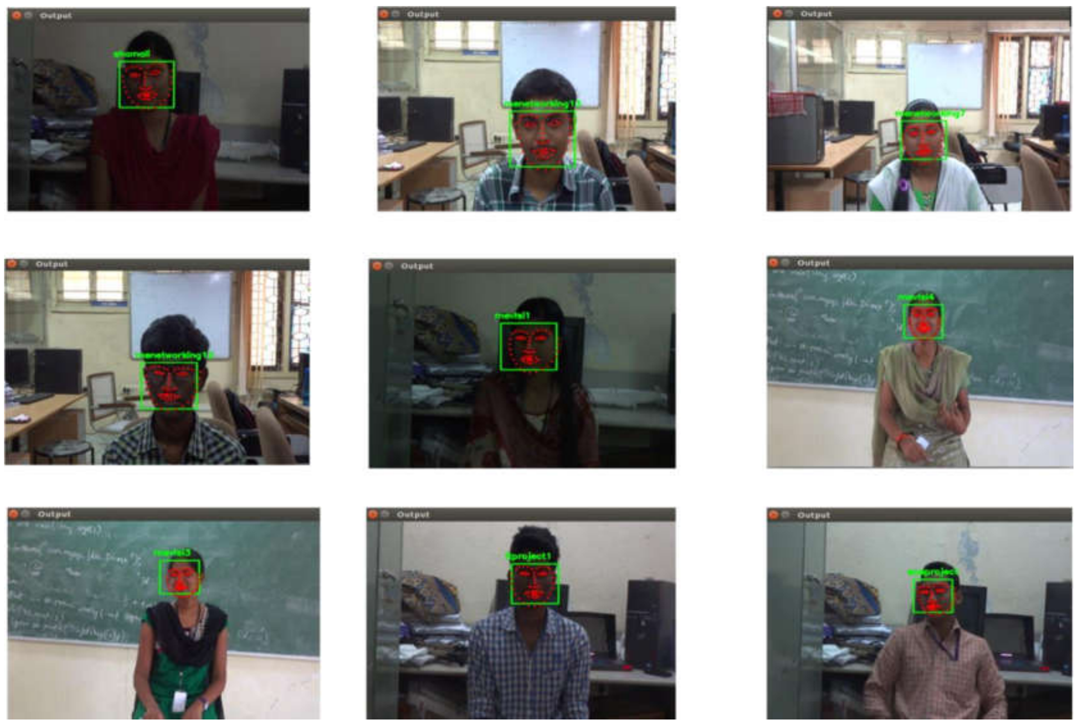

Figure 14 Face Recognition results for MIT-Dataset

5.2 Complexity Analysis

The dataset processed using any deep learning models are huge and complex. Processing such huge data always requires multiple loops to traverse the whole datasets. However, the increased number of loops always consumes more time for obtaining the results. In order to reduce the number of loops and for fast processing, the present day neural model utilizes vectorization. The complexity of the proposed model is found to be in O(mlogn), where m represents the total number of features and n represents the total number of stacked classifiers.

6. **Summary**

This work presented an Ensemble of Convolutional Neural Networks (ECNN) for face recognition. The performance of the proposed ECNN method is analyzed using benchmark datasets namely Webface and YouTube face in terms of recognition accuracy. The proposed ECNN model performs better than Inception- v3, VGG16, VGG19, Xception and ResNet50 CNN models with a Rank-5 accuracy of 97.12% on Webface dataset. It is concluded that the proposed ECNN has outperformed the existing state of art methods in terms of recognition accuracy.

References

1. Minaee, S.; Luo, P.; Lin, Z.; Bowyer, K. Going deeper into face detection: A survey. *arXiv* 2021,
arXiv:2103.14983.
2. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
3. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 21–37. [Google Scholar]
4. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* 2017, *39*, 1137–1149.
5. Najibi, M.; Samangouei, P.; Chellappa, R.; Davis, L.S. Ssh: Single stage headless face detector. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017;
pp. 4875–4884.
6. Werner, P.; Khalifa, A.; Al-Hamadi, A. SFPD: Simultaneous Face and Person Detection in Real-Time for human–robot Interaction. *Sensors* 2021, *21*, 5918.
7. Jiao, L.; Zhang, F.; Liu, F.; Yang, S.; Li, L.; Feng, Z.; Qu, R. A survey of deep learning-based object detection. *IEEE Access* 2019, *7*, 128837–128868.
8. Zhang, K.; Zhang, Z.; Li, Z.; Qiao, Y. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Process. Lett.* 2016, *23*, 1499–1503
9. Zhang, C.; Xu, X.; Tu, D. Face detection using improved faster rcnn. *arXiv* 2018, arXiv:1802.02142
10. Najibi, M.; Singh, B.; Davis, L.S. Fa-rpn: Floating region proposals for face detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 7723–7732.
11. Zhang, H.; Chang, H.; Ma, B.; Shan, S.; Chen, X. Cascade retinanet: Maintaining consistency for single- stage object detection. *arXiv* 2019, arXiv:1907.06881.
12. Huang, G.B.; Mattar, M.; Berg, T.; Learned-Miller, E. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In Proceedings of the Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition, Marseille, France, 17 October

2008.
13. Wu, Y.; Ji, Q. Facial landmark detection: A literature survey. *Int. J. Comput. Vis.* 2019, *127*, 115–142.
14. Gogić, I.; Ahlberg, J.; Pandžić, I.S. Regression-based methods for face alignment: A survey. *Signal Process.* 2021, *178*, 107755.
15. Trigeorgis, G.; Snape, P.; Nicolaou, M.A.; Antonakos, E.; Zafeiriou, S. Mnemonic descent method: A recurrent process applied for end-to-end face alignment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4177–4187.
16. Zhu, H.; Sheng, B.; Shao, Z.; Hao, Y.; Hou, X.; Ma, L. Better initialization for regression-based face alignment. *Comput. Graph.* 2018, *70*, 261–269.
17. Valle, R.; Buenaposada, J.M.; Valdes, A.; Baumela, L. A deeply-initialized coarse-to-fine ensemble of regression trees for face alignment. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 585–601.
18. Feng, Z.H.; Huber, P.; Kittler, J.; Christmas, W.; Wu, X.J. Random cascaded-regression copse for robust facial landmark detection. *IEEE Signal Process. Lett.* 2014, *22*, 76–80.
19. Zhu, S.; Li, C.; Loy, C.C.; Tang, X. Face alignment by coarse-to-fine shape searching. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 4998–5006.
20. Kumar, A.; Chellappa, R. Disentangling 3d pose in a dendritic cnn for unconstrained 2d face alignment. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 430–439
21. Guo, X.; Li, S.; Yu, J.; Zhang, J.; Ma, J.; Ma, L.; Liu, W.; Ling, H. PFLD: A practical facial landmark detector. *arXiv* 2019, arXiv:1902.10859.
22. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.
23. Schroff, F.; Kalenichenko, D.; Philbin, J. Facenet: A unified embedding for face recognition and clustering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7– 12 June 2015; pp. 815–823.
24. Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; Liu, W. Cosface: Large margin cosine loss for deep face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 5265–5274.
25. Zhong, Y.; Deng, W.; Hu, J.; Zhao, D.; Li, X.; Wen, D. SFace: Sigmoid-constrained Hypersphere Loss for Robust Face Recognition. *IEEE Trans. Image Process.* 2021, *30*, 2587–2598.
26. Li, L.; Mu, X.; Li, S.; Peng, H. A Review of Face Recognition Technology. *IEEE Access* 2020, *8*, 139110– 139120.
27. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* 2012, *25*, 1097–1105.
28. Sun, Y.; Wang, X.; Tang, X. Deeply learned face representations are sparse, selective, and robust. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 2892–2900.
29. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich,
A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
30. He, R.; Wu, X.; Sun, Z.; Tan, T. Wasserstein cnn: Learning invariant features for nir-vis face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* 2018, *41*, 1761–1773.
31. Liu, W.; Wen, Y.; Yu, Z.; Li, M.; Raj, B.; Song, L. Sphereface: Deep hypersphere embedding for face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 June 2017; pp. 212–220.
32. Deng, J.; Zhou, Y.; Zafeiriou, S. Marginal loss for deep face recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Honolulu, HI, USA, 21–26 June 2017;
pp. 60–68
33. Khalifa, A.; Al-Hamadi, A. A Survey on Loss Functions for Deep Face Recognition Network. In Proceedings of the 2021 IEEE 2nd International Conference on human–machine Systems (ICHMS), Magdeburg, Germany, 8–10 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–7.
34. Fuad, M.T.H.; Fime, A.A.; Sikder, D.; Iftee, M.A.R.; Rabbi, J.; Al-Rakhami, M.S.; Gumaei, A.; Sen, O.; Fuad, M.; Islam, M.N. Recent Advances in Deep Learning Techniques for Face Recognition. *IEEE Access* 2021, *9*, 99112–99142.
35. Hsu, G.S.J.; Wu, H.Y.; Yap, M.H. A comprehensive study on loss functions for cross-factor face

recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 826–827.

36. Hu, W.C.; Chen, C.H.; Chen, T.Y.; Huang, D.Y.; Wu, Z.C. Moving object detection and tracking from video captured by moving camera. *J. Vis. Commun. Image Represent.* 2015, *30*, 164–180.
37. Liu, F.; Gong, C.; Huang, X.; Zhou, T.; Yang, J.; Tao, D. Robust visual tracking revisited: From correlation filter to template matching. *IEEE Trans. Image Process.* 2018, *27*, 2777–2790.
38. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple online and realtime tracking. In Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 3464–3468.
39. Wojke, N.; Bewley, A.; Paulus, D. Simple online and realtime tracking with a deep association metric. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 3645–3649.
40. Lian, Z.; Shao, S.; Huang, C. A real time face tracking system based on multiple information fusion. *Multimed. Tools Appl.* 2020, *79*, 16751–16769.
41. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* 2017, arXiv:1704.04861.
42. Chen, S.; Liu, Y.; Gao, X.; Han, Z. Mobilefacenets: Efficient cnns for accurate real-time face verification on mobile devices. In Proceedings of the Chinese Conference on Biometric Recognition, Urumchi, China, 11– 12 August 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 428–438.
43. Nguyen, H.V.; Bai, L. Cosine similarity metric learning for face verification. In Proceedings of the Asian Conference on Computer Vision, Queenstown, New Zealand, 8–12 November 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 709–720.