Real Time Face Expression Recognition along with Balanced FER2013 Dataset using CycleGAN

Fatma Mazen Ali Mazen[1*], Ahmed Aly Nashat[2], Rania Ahmed Abdel Azeem Abul Seoud[3]
Electronics and Communication Engineering Department
Faculty of Engineering, Fayoum University
Fayoum 63514, Egypt

Abstract—**Human face expression recognition is an active research area that has massive applications in medical field, crime investigation, marketing, online learning, automobile safety and video games. The first part of this research defines a deep neural network model-based framework for recognizing the seven main types of facial expression, which are found in all cultures. The proposed methodology involves four stages: (a) pre-processing the FER2013 dataset through relabeling to avoid misleading results and getting rid of non-face and non-frontal faces; (b) design of an efficient stable Cycle Generative Adversarial Network (CycleGAN), which provides unsupervised expression-to-expression translation. The CycleGAN has been designed and trained with a new cycle consistency loss. (c) Generating new images to overcome the class imbalance and finally (d) building the DNN architecture for recognizing the face sign expression, using the pretrained VGG-Face model with vggface weights. The second part encompasses the design of a GPU-accelerated face expression recognition system for real time video sequences using NVIDIAs Compute Unified Device Architecture (CUDA). OpenCV library has been compiled from scratch with CUDA and NVIDIA CUDA Deep Neural Network library, cuDNN. For face detection stage Haar Cascaded and deep learning were used and tested using both CPU and GPU as backend. Results show that the designed model run time to recognize a face sign is 0.44 seconds. Besides, the average test accuracy has been increased from 64% for the original FER2013 dataset to 91.76% for the modified balanced version using the same transfer learning model.**

Keywords—Facial expressions detection and recognition; multi-task cascaded convolutional networks; transfer learning; residual neural network; CycleGAN; FER2013; GPU and CUDA; HAAR

I. INTRODUCTION

Face sign expressions are one of the most active research areas in computer vision since they are a form of nonverbal communication for people in the deaf community. It is, also, used for understanding human behavior, mental disorder detection, cognition of human emotions, and lie detection. They convey non-verbal cues, which play an important role in interpersonal relations. Automatic recognition of facial expressions can be a vital component of natural human-machine interfaces; it may also be used in different areas such as artificial intelligence, computer vision, psychology, physiology, behavioral science and in clinical practice. Some robots can also benefit from the ability to recognize expressions [1]. Automated analysis of facial expressions for behavioral science or medicine is another possible application domain. It can be used to detect the state of the learner in E-learning, help doctors to understand and analyze the behavior of children with Autism Spectrum Disorder (ASD) who are known to have difficulty in producing and perceiving emotional facial expression [2]. The system of facial expression recognition is divided into three steps: optimal preprocessing, feature extraction or selection, and classification, particularly under conditions of input data variability to attain successful recognition performance.

Dataset preprocessing is an elementary step in machine learning systems especially when the dataset contains wrongly labeled images due to source crowding and suffers from class imbalance, which leads to biased learning toward majority classes. In this paper, an innovative strategy has been adopted to overcome this shortage. CycleGAN is a promising data augmentation scheme recently adopted to solve the class imbalance problem by generating new samples of the minority class. An efficient stable CycleGAN has been designed and trained to perform style transfer from a reference domain, which is the neutral class, to a target domain, which could be any of the other face expression classes. After creating a clean balanced dataset, the transfer learning approach has been adopted for model design. Designing a model from scratch is a time-consuming process and will never outperform efficiently, since the pretrained models have been trained on datasets containing millions of images which is a huge number compared to our dataset.

Many machine learning algorithms have been proposed for successful recognition performance, but they have not reached the optimal performance due to lack of the optimal set of features required for classification. Feature extraction is the process of using domain knowledge of the data to create features that make machine learning algorithms work. Coming up with features is difficult, time-consuming, requires expert knowledge. Deep learning aims at learning feature hierarchies where features from higher levels of the hierarchy are formed by lower-level features, so it eliminates the bad need to feature extraction and as a result achieving better performance in less time than traditional machine learning algorithms. Convolutional Neural Networks, CNN, are currently one of the most prominent algorithms for deep learning with image data. Whereas for traditional machine learning relevant features must be extracted manually. In computer vision machines, storing the knowledge learned by solving one problem and applying it to another similar problem is challenging. The cross-domain transfer learning models have received enormous

[*] Corresponding Author

attention in face expression classification and generally artificial intelligence applications, where the training and test data are drawn from different feature space and different distribution. The knowledge learned from the learning dataset can help improve prediction accuracy in the testing domain, especially when the testing data are target scant. Also, knowledge from a labeled domain can help generate labels for an unlabeled domain, which may avoid a costly human labeling process.

In recent years, massive computer visions algorithms and techniques have been developed and employed for designing real time face expression recognition system due to its significance not only in human-to-human interaction but also in Human Machine Interaction (HMI). HMI is an extruded field in computer science that aims at making the computer intelligent such that it interacts with human the way human and human interact. This task can be decomposed into two stages, face detection for face localization and face expression classification. Recently, two main approaches have been adopted for face detection, the traditional machine learning approach, and the deep learning approach.

In this paper, a new framework for real time face expression recognition in video sequences has been designed and tested. It comprises two phases. For face detection phase, two versions of OpenCV library have been employed. The first version is standard OpenCV library that was used for machine learning approach. With the rapid development of GPU, deep learning became more powerful in classification tasks. The second version is compiled from scratch with CUDA and cuDNN support to achieve optimal hardware resources employment through interaction between our Graphical Processing Unit (GPU) and deep learning module of OpenCV which undoubtedly has a great effect in improving the speed of video by increasing number of frames that can be processed per second (FPS) making our system more convenient for real time applications.

The rest of the paper is organized as follows: Section 2 discusses the work related to the face expression classification problem. Section 3 illustrates the dataset and the proposed methodology for the facial expression recognition. Experimental results and discussion are presented in Section 4. Finally, Section 5 concludes the paper and outlines directions for future work.

II. Related Work

In the last decade, face expression recognition has become a hot research area. Several papers have been published using classical machine learning approaches like Support Vector Machine (SVM) and Random Forest, whereas others use modern schemes like deep learning, convolutional neural networks (CNNs), transfer learning and ensemble models.

Ying Zilu and G. Zhang [3] adopted a framework for facial expression recognition based on the combination of non-negative matrix factorization (NMF) and support vector machine (SVM). They achieved a recognition rate of 66.19% with their algorithm applied to Japanese female facial expression database (JAFEE database). In [4], face expression recognition system was designed by Y. Luo, C. M. Wu and Y. Zhang based on a hybrid scheme of principal component analysis (PCA) and local binary pattern (LBP) for feature extraction and support vector machine (SVM) for facial expression classification. They achieved a recognition rate of 93.75% using a small dataset of 350 face sign images. C. Shan, S. Gong and P. W. McOwan [5] proposed a novel scheme based on local binary pattern (LBP) for facial expression recognition. The Cohn-Kanade Database was used to demonstrate the efficiency of their proposed approach. Their model attains a recognition rate of 79.1%. In [6], Ahmed A. Nashat proposed a new approach for extracting features for facial expression classification. Firstly, the discrete wavelet packet best tree (DWPBT) decomposition is used to remove the spatial and spectral redundancies for each block of the desired face regions. Then, the radial difference LGP (RD-LGP) in radial directions of a circular grid is used as a descriptor for facial expression recognition. The average recognition rate was 83.4%.

Y. Wang, Y. Li, Y. Song and X. Rong [7] have proposed an effective hybrid approach of random forest and convolutional neural networks. Their approach achieved an accuracy of 98.9% on JAFFE dataset, 99.9% on CK+ dataset, 84.3% on FER2013 dataset and 92.3% on RAF-DB dataset. E. Barsoum, C. Zhang, C. C. Ferrer and Z. Zhang [8] worked on using a deep CNN on noisy labels acquired via crowdsourcing for ground truth images. They used 10 taggers to relabel each image in the dataset, and used various cost functions for their DCNN, achieving decent accuracy. They reached accuracy of 84.986% on FER+ dataset. C. Pramerdorfer and M. Kampel [9] formed an ensemble of recent deep convolutional neural networks. They investigated the approaches adopted by six current state-of-the-art papers and ensembles their networks to reach 75.2% test accuracy on FER2013 outperforming previous works without needing auxiliary training data.

In [10], X. Wang, J. Huang, J. Zhu, M. Yang and F. Yang have utilized transfer learning using Keras VGG-Face library and each of ResNet50, SeNet50 and VGG16 as pre-trained models. They managed to achieve test accuracy of 75.8% which surpassed all existing publications at this time. A. Chen, H. Xing, and F. Wang [11] proposed an efficient and secure facial expression recognition method based on the edge cloud framework combined with the improved CycleGAN to solve class imbalance of FER2013 dataset by generation 4000 new sample for disgust class and 800 new samples for surprised class achieving higher recognition rate not only on the data augmented classes but also on the other classes. They achieved average recognition rate of 78.61% on the enhanced dataset. Z. Zhang, P. Luo, C. C. Loy, and X. Tang [12] also achieved 75.1% test accuracy by adding auxiliary data and additional features: a vector of HoG features was computed from face patches and processed by the first FC layer of the CNN. Facial landmark registration has been also employed.

In this work, a neoteric relabeled balanced FER2013 dataset has been introduced. A balanced CycleGAN has been designed and trained using new cycle consistency loss to preserve luminance conditions of reference class to overcome class imbalance problem through generating new images for minority class (Disgust). State of the art results have been achieved through dataset cleaning and pre-processing. A new deep neural network architecture for recognizing the face sign expression, using the pretrained VGG-Face model with vggface weights and the modified balanced version of the FER2013 dataset is designed and implemented. To enhance our work, A GPU-accelerated system for face expression recognition in real time video stream is introduced achieving 312.01% faster inference for feature-based approach and inference speed improvement by up to 169.74% for deep learning-based approach.

III. Methodology

A. The Dataset

The proposed methodology was trained and tested using the open-source FER2013 dataset [13-14], which is created for an ongoing project by Pierre-Luc Carrier and Aaron Courville from university of Montreal, then shared publicly for a Kaggle competition, shortly before ICML 2013. The dataset consists of 35.887 labelled 48x48 grayscale human facial expressions. These are afraid (11.42%), angry (11.13%), disgusted (1.21%), happy (20.1%), neutral (13.84%), sad (13.46%) and surprised (8.84%). To train and test the performance of the proposed classification model, we selected 80% of the set for training and the rest for testing. Fig. 1 shows some sample images from the FER2013 dataset.

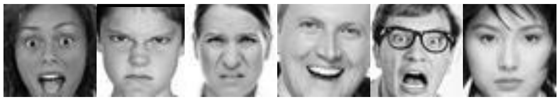


Fig. 1. Sample Images from the FER2013 Dataset.

B. DataSet Preprocessing Phase

Crowdsourcing has become a widely used approach to gather ground truth labels to form a dataset. However, these labels can be very noisy which causes model misleading and as a result low recognition rate [8]. FER2013 suffers from severe crowdsourcing as it contains non-face images, text images, sleepy faces, profile images which are indistinguishable by humans, and large number of wrongly labelled images. According to [15], the overall accuracy of the facial expression classification using this dataset does not exceed 65%. In this research, three steps have been taken to resolve this problem. First, we deleted non-face images, text images and profile images. Second, we re-labelled wrongly labelled images based on class distribution using convolutional neural networks (CNN). Finally, the class imbalance problem has been solved by designing a CycleGAN used for generating new face expressions for the minority classes. CycleGAN phase will be discussed in detail in the next section. Fig. 2 displays some bad images, which have been deleted.

As shown in Fig. 3, a few examples of wrongly labelled misleading images due to crowd sourcing are displayed with the wrong expression in the red color and the right expression in the blue color.

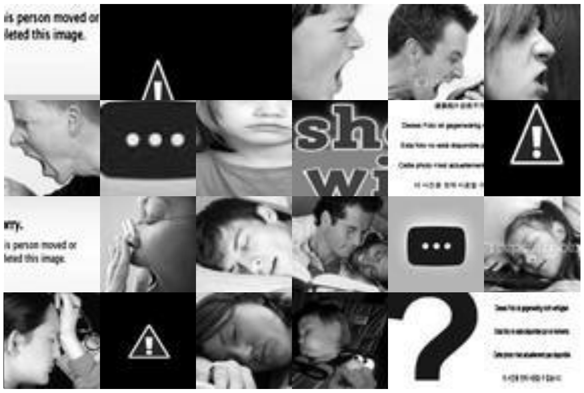


Fig. 2. Samples of Bad Images that have been deleted from the Dataset.

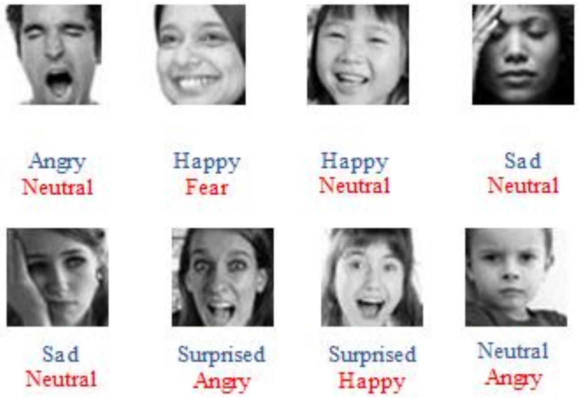


Fig. 3. Samples of Mislabeled Images.

Besides crowd resourcing, the dataset suffers from class imbalance problem which results in model biasing. As shown in Fig. 4, class1 contains only 436 images which are too low compared to other classes which contain thousands of images. The class imbalance problem has been solved by designing a CycleGAN used for generating new samples for the minority class. CycleGAN phase will be discussed in detail in the next section.

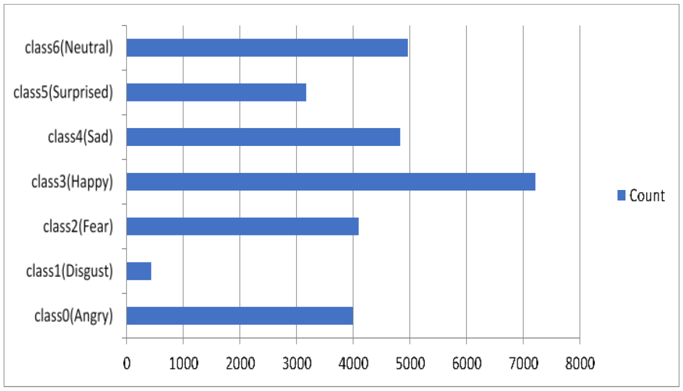


Fig. 4. A Severe Class Imbalance with the Highest Number of Images at Class 3 and the Lowest at Class 1.

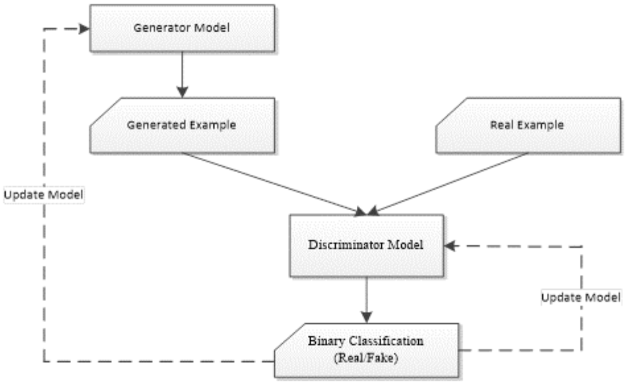


Fig. 5. Flowchart of the Generative Adversarial Network Model Architecture.

C. CycleGAN Phase

It is a challenging task to classify images with multiple class labels using only a small number of labelled examples, especially when the label (class) distribution is imbalanced. Generative Adversarial Networks (GANs) can be used to generate images from an adversarial training. The generator attempts to produce a realistic image to fool the discriminator, which tries to distinguish whether its input image is from the training set or the generated set. The flow chart of the GAN architecture is shown in Fig. 5. J. Y. Zhu, T. Park, P. Isola, and A. Efros [16] proposed the CycleGAN model, which can do image-to-image transition between two unpaired image domains. This model has been employed to build our framework.

The architecture of the discriminator, the generator and the composite model are like those used in [17] except for employing some training tricks like using dropout layers to avoid GAN failure modes, labels flipping (True label was used for generated images while false label was used for Real images) and using Batch Normalization layers to produce sharper generated images. Unlike other GAN models for image translation, the CycleGAN does not require a dataset of paired images.

The discriminator and the composite model architecture of the CycleGAN are shown in Fig. 6 and Fig. 7, respectively.

1) CycleGAN loss functions: Each generator model is optimized via the combination of four outputs with four loss functions and are defined by (1), (2), (3), and (4) as follows:

- Adversarial loss (L_2 or mean squared error). It is calculated as the L2 distance between the model output and the target values of 1.0 for real and 0.0 for fake.

$$\text{Adversarial loss} = L_2 \tag{1}$$

- Identity loss (L_1 or mean absolute error). It is calculated as the L_1 distance between the input and output image for each sequence of translations.

$$\text{Identity loss} = L_1 \tag{2}$$

- Forward cycle loss

$$\text{Forward cycle loss} = \frac{1}{3}(L_1 + L_2 + \text{SSIM loss}) \tag{3}$$

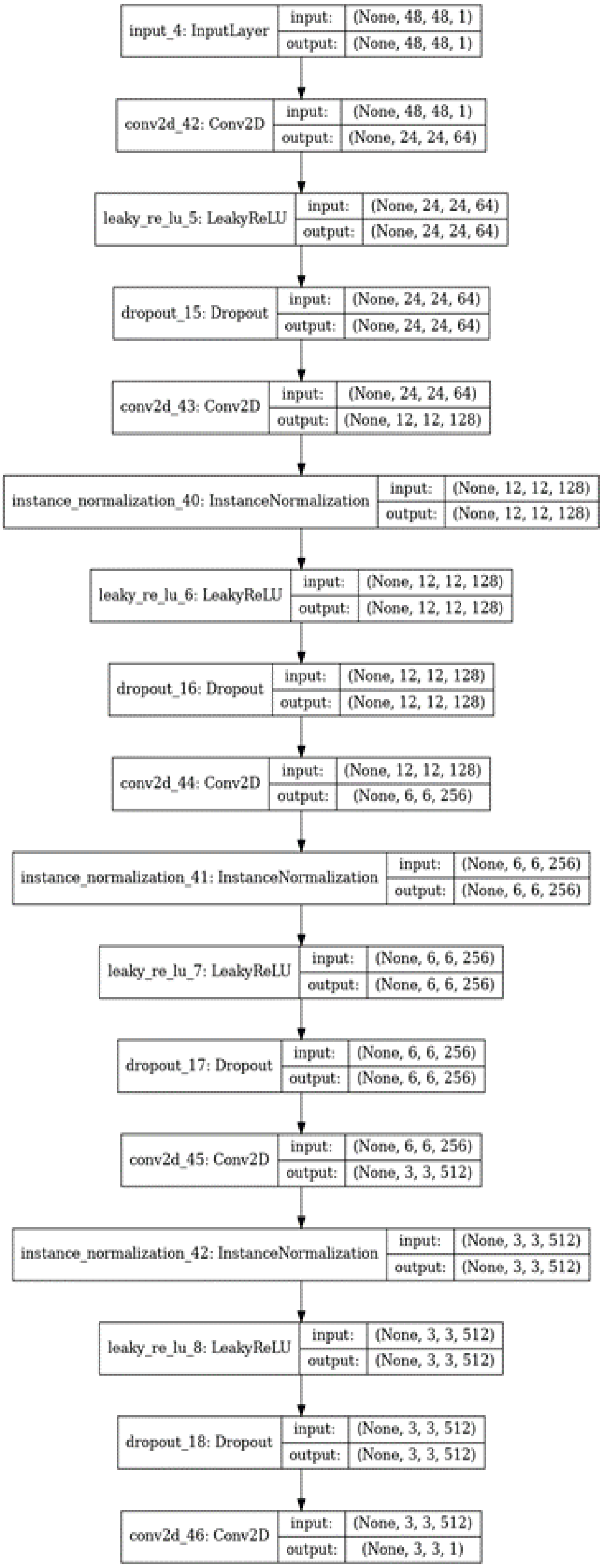

input_4: InputLayer | input: (None, 48, 48, 1) | output: (None, 48, 48, 1)
conv2d_42: Conv2D | input: (None, 48, 48, 1) | output: (None, 24, 24, 64)
leaky_re_lu_5: LeakyReLU | input: (None, 24, 24, 64) | output: (None, 24, 24, 64)
dropout_15: Dropout | input: (None, 24, 24, 64) | output: (None, 24, 24, 64)
conv2d_43: Conv2D | input: (None, 24, 24, 64) | output: (None, 12, 12, 128)
instance_normalization_40: InstanceNormalization | input: (None, 12, 12, 128) | output: (None, 12, 12, 128)
leaky_re_lu_6: LeakyReLU | input: (None, 12, 12, 128) | output: (None, 12, 12, 128)
dropout_16: Dropout | input: (None, 12, 12, 128) | output: (None, 12, 12, 128)
conv2d_44: Conv2D | input: (None, 12, 12, 128) | output: (None, 6, 6, 256)
instance_normalization_41: InstanceNormalization | input: (None, 6, 6, 256) | output: (None, 6, 6, 256)
leaky_re_lu_7: LeakyReLU | input: (None, 6, 6, 256) | output: (None, 6, 6, 256)
dropout_17: Dropout | input: (None, 6, 6, 256) | output: (None, 6, 6, 256)
conv2d_45: Conv2D | input: (None, 6, 6, 256) | output: (None, 3, 3, 512)
instance_normalization_42: InstanceNormalization | input: (None, 3, 3, 512) | output: (None, 3, 3, 512)
leaky_re_lu_8: LeakyReLU | input: (None, 3, 3, 512) | output: (None, 3, 3, 512)
dropout_18: Dropout | input: (None, 3, 3, 512) | output: (None, 3, 3, 512)
conv2d_46: Conv2D | input: (None, 3, 3, 512) | output: (None, 3, 3, 1)

Fig. 6. The Discriminator Architecture.

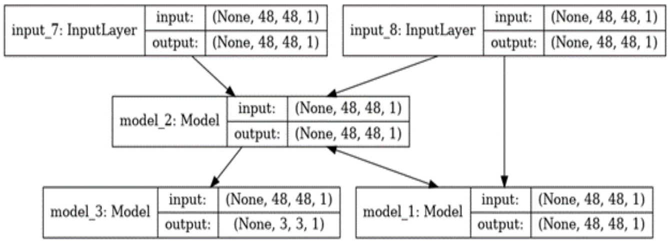


Fig. 7. The Composite Model of the CycleGAN.

- Backward cycle loss

$$\text{Backward cycle loss} = \frac{1}{3}(\mathrm{L}_1 + \mathrm{L}_2 + \text{SSIM loss}) \quad (4)$$

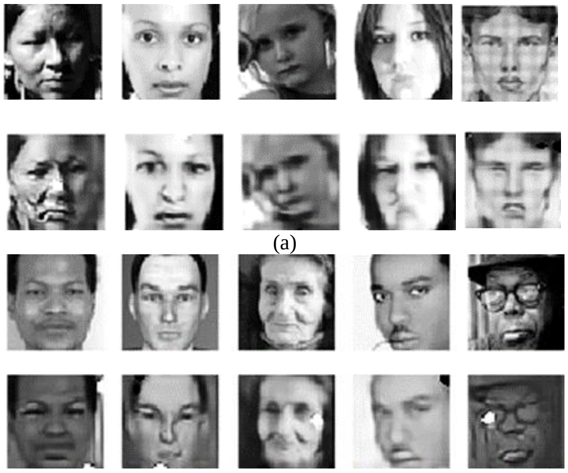
In this paper, the cycle consistency loss, for a better-quality image, is defined as the average of the L_1 distance and the L_2 distance. The structural similarity index measure (SSIM) loss is defined by (5) as:

$$\text{SSIM loss} = 1 - \text{SSIM index} \quad (5)$$

Where SSIM Index is computed by considering the luminance, contrast, and structural similarity. Since the image pixels are correlated and not independent, L_2 loss is unsuitable choice to improve the quality of generated images. SSIM loss term is added to the cycle consistency loss to compensate for this disadvantage by reducing structural distortions between the input and output images. It is worthy mentioned that SSIM was designed for gray scale images.

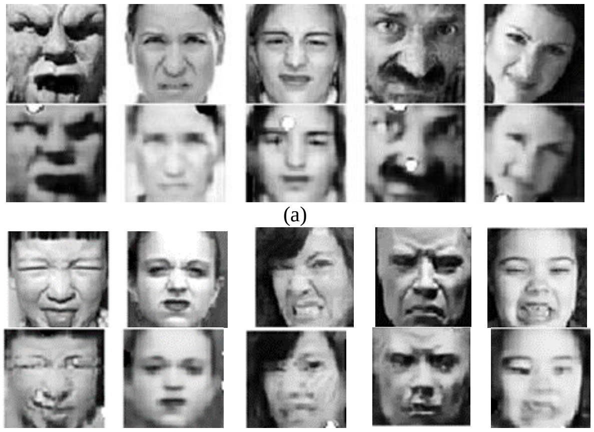
2) Training details: In our design, after five trials, the forward and the backward cycle loss are given more weight than the adversarial loss whereas the identity loss weight has been set to two. In other words, λ_{cyc} is set to five and λ_{id} equals to two.

For all the experiments, the Adam optimizer has been used with a batch size of one. All networks were trained from scratch with a learning rate of 0.0002. The learning rate was kept constant for the first 100 epochs and linearly decays to zero over the next 100 epochs. Fig. 8 shows two sets of sample images of translation from class 6, (Neutral), to class 1, (Disgust).


(a)

(b)

Fig. 8. Two Sets, a and b, of Samples of Neutral to Disgust Translation.


(a)

(b)

Fig. 9. Two Sets, a and b, of Samples of Disgust to Neutral Translation.

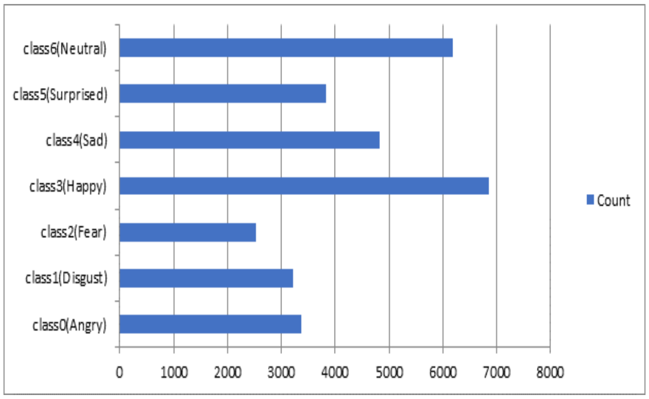


Fig. 10. Data Distribution of the New Balanced FER2013 Dataset.

Fig. 9 displays sample images of class 1 (Disgust) to class 6 (Neutral) translation which are close to real ones.

Fig. 10 shows data distribution of the new balanced FER2013 dataset, after cleaning it from bad images and generating new images using CycleGAN for the minority class.

D. Transfer Learning Phase

In this paper, transfer learning scheme has been adopted as it has a significant role in achieving state of the art accuracy. Keras VGG-Face pre-trained model [18] has been employed with vggface weights. As this network accepts RGB images with size 224x224, the 48x48 gray scale images in FER2013 during training time have been resized and recolored using image data generator.

1) Training details: In our design, the top layers of the VGG-Face model, which include last fully connected layers and softmax layer have been removed and a flatten layer was added to extract the bottleneck feature vector. To reduce overfitting due to large network size, the last three fully connected (dense) layers have been replaced by only one fully connected layer with 128 neurons followed by a dropout layer with dropout rate equals to 0.3 and then a softmax layer with seven outputs has been added to match the seven output facial expressions.

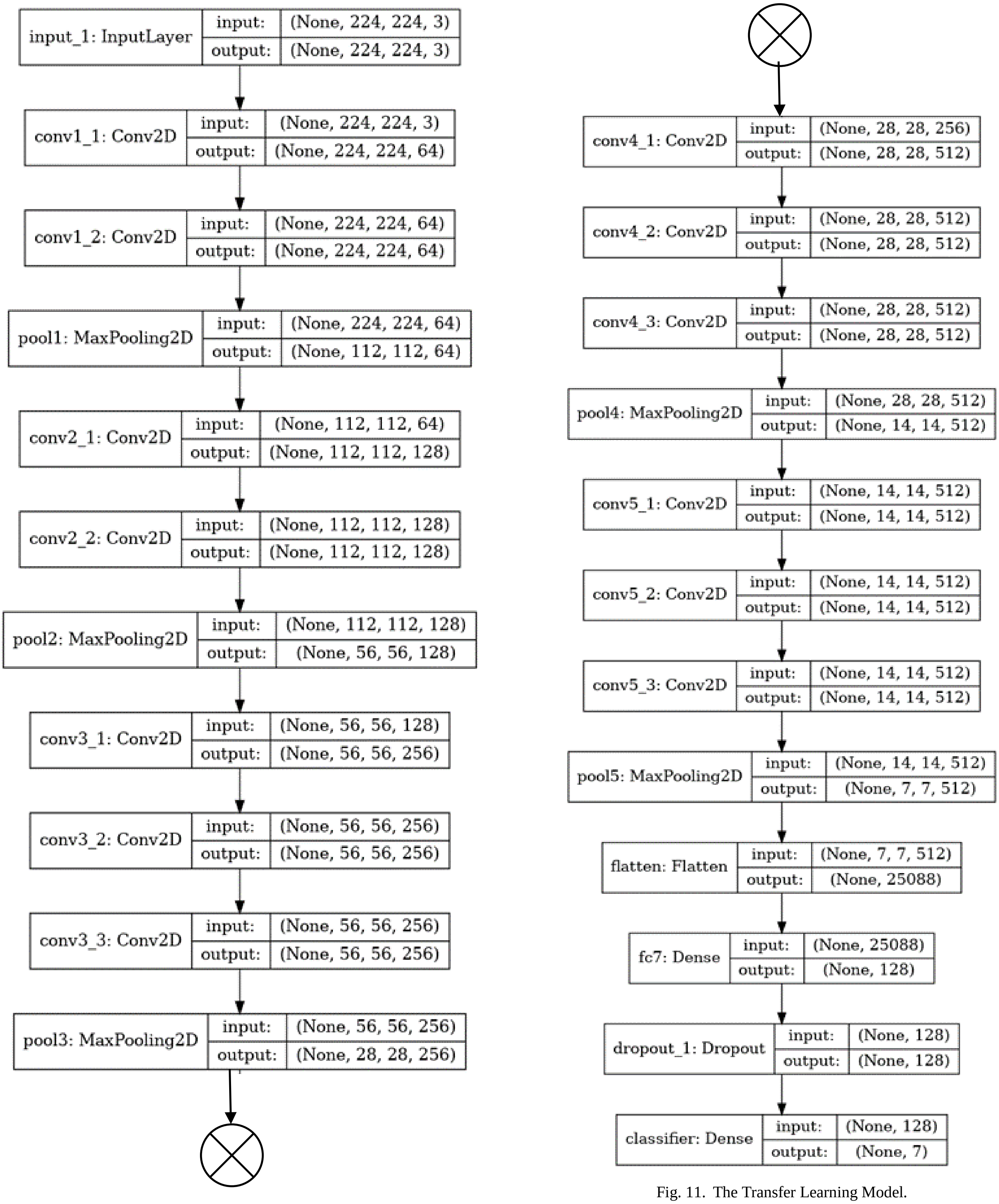

input_1: InputLayer | input: | (None, 224, 224, 3) | output: | (None, 224, 224, 3)
conv1_1: Conv2D | input: | (None, 224, 224, 3) | output: | (None, 224, 224, 64)
conv1_2: Conv2D | input: | (None, 224, 224, 64) | output: | (None, 224, 224, 64)
pool1: MaxPooling2D | input: | (None, 224, 224, 64) | output: | (None, 112, 112, 64)
conv2_1: Conv2D | input: | (None, 112, 112, 64) | output: | (None, 112, 112, 128)
conv2_2: Conv2D | input: | (None, 112, 112, 128) | output: | (None, 112, 112, 128)
pool2: MaxPooling2D | input: | (None, 112, 112, 128) | output: | (None, 56, 56, 128)
conv3_1: Conv2D | input: | (None, 56, 56, 128) | output: | (None, 56, 56, 256)
conv3_2: Conv2D | input: | (None, 56, 56, 256) | output: | (None, 56, 56, 256)
conv3_3: Conv2D | input: | (None, 56, 56, 256) | output: | (None, 56, 56, 256)
pool3: MaxPooling2D | input: | (None, 56, 56, 256) | output: | (None, 28, 28, 256)
conv4_1: Conv2D | input: | (None, 28, 28, 256) | output: | (None, 28, 28, 512)
conv4_2: Conv2D | input: | (None, 28, 28, 512) | output: | (None, 28, 28, 512)
conv4_3: Conv2D | input: | (None, 28, 28, 512) | output: | (None, 28, 28, 512)
pool4: MaxPooling2D | input: | (None, 28, 28, 512) | output: | (None, 14, 14, 512)
conv5_1: Conv2D | input: | (None, 14, 14, 512) | output: | (None, 14, 14, 512)
conv5_2: Conv2D | input: | (None, 14, 14, 512) | output: | (None, 14, 14, 512)
conv5_3: Conv2D | input: | (None, 14, 14, 512) | output: | (None, 14, 14, 512)
pool5: MaxPooling2D | input: | (None, 14, 14, 512) | output: | (None, 7, 7, 512)
flatten: Flatten | input: | (None, 7, 7, 512) | output: | (None, 25088)
fc7: Dense | input: | (None, 25088) | output: | (None, 128)
dropout_1: Dropout | input: | (None, 128) | output: | (None, 128)
classifier: Dense | input: | (None, 128) | output: | (None, 7)

Fig. 11. The Transfer Learning Model.

During training the network, the model has been compiled using Adam optimizer with learning rate equals to 0.001 and Beta_1 equals to 0.5. The input to the network has been divided into minibatches of 128 images and the total number of epochs equals to twenty epochs. All experiments have been conducted using Keras with TensorFlow backend on Alien ware laptop with Nvidia GeForce GTX 1070 GPU. To prevent model overfitting and performance deterioration, early stopping callback function is used to halt training the model at the right time. Model Check Point callback has been used to save the model only when better results are obtained according to predefined performance measures. Reduce on Plateau callback has been used to reduce learning rate when the validation accuracy decays. The detailed description of our transfer learning model is shown in Fig. 11.

The upcoming sections discuss the procedure of designing a real time facial expression recognition system to detect face and recognize the seven standard face expressions in real time video sequences.

E. Compiling OpenCV from Source

In this stage OpenCV with CUDA and cuDNN support has been compiled from source to achieve the best optimum utilization of hardware resources (Alien ware laptop with NVidia GeForce GTX 1070 GPU). OpenCV has been compiled with eight cores which should be adjusted according to the number of processor cores you use. The block diagram of this process is shown in Fig. 12.

The CMake output for NVIDIA CUDA and cuDNN indicates that CUDA version, is 10.1 and cuDNN version is 7.6.5. Another important CMake output that should be verified before proceeding to compilation step are the proper python paths to Interpreter and Libraries.

F. Face Detection Phase

Face detection is a challenging computer vision problem which aims to identify and localize faces in images. It is a primary step for any facial expression recognition (FER) system. It can be accomplished using the classical machine learning (feature based) Haar cascaded classifier or using recent deep learning-based approaches through using the OpenCV library.

1) Feature-based approach (Haar cascaded classifier): OpenCV provides the cascade classifier class that can be used for face detection. The constructor can take a filename as an argument that specifies the XML file for a pre-trained model. The model can be used for face detection on an image or video by calling the detectMultiScale function whose output is a list of bounding boxes for all faces detected. ScaleFactor and minNeighbors are two parameters that should be carefully tuned for a given dataset as the ScaleFactor controls how the input image is scaled before detection.

2) Deep learning-based approach: After building OpenCV from source with CUDA and cuDNN support, now its Deep Neural Network (DNN) module can be used. It contains a CNN-based face detector which enhances the face detection performance compared to machine learning-based models like Haar Cascade classifier. It utilizes the Single Shot Detector (SSD) framework with ResNet as the base network. The confidence level is extracted after looping over the detections to suppress weak detections if they do not meet the minimum confidence level. In our experiment, the minimum confidence level was set to 0.5.

G. Face Expression Recognition Phase

In this phase, the system classifies each frame of the video into one of the seven universal expressions - Anger, Disgust, Fear, Happiness, Sadness, Surprise and Neutral as labelled in the FER2013 dataset. The flowchart of the proposed deep learning framework and machine learning framework are shown in Fig. 13 and Fig. 14, respectively.

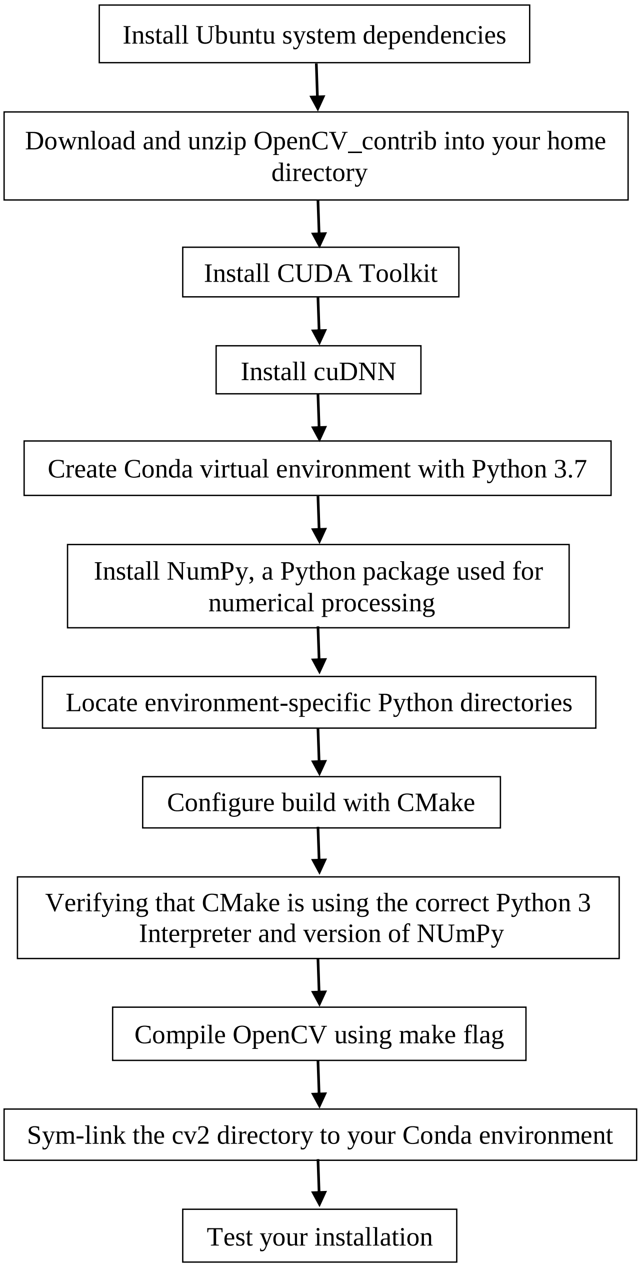

Install Ubuntu system dependencies
Download and unzip OpenCV_contrib into your home directory
Install CUDA Toolkit
Install cuDNN
Create Conda virtual environment with Python 3.7
Install NumPy, a Python package used for numerical processing
Locate environment-specific Python directories
Configure build with CMake
Verifying that CMake is using the correct Python 3 Interpreter and version of NUmPy
Compile OpenCV using make flag
Sym-link the cv2 directory to your Conda environment
Test your installation

Fig. 12. The Block Diagram of OpenCV Compilation Process.

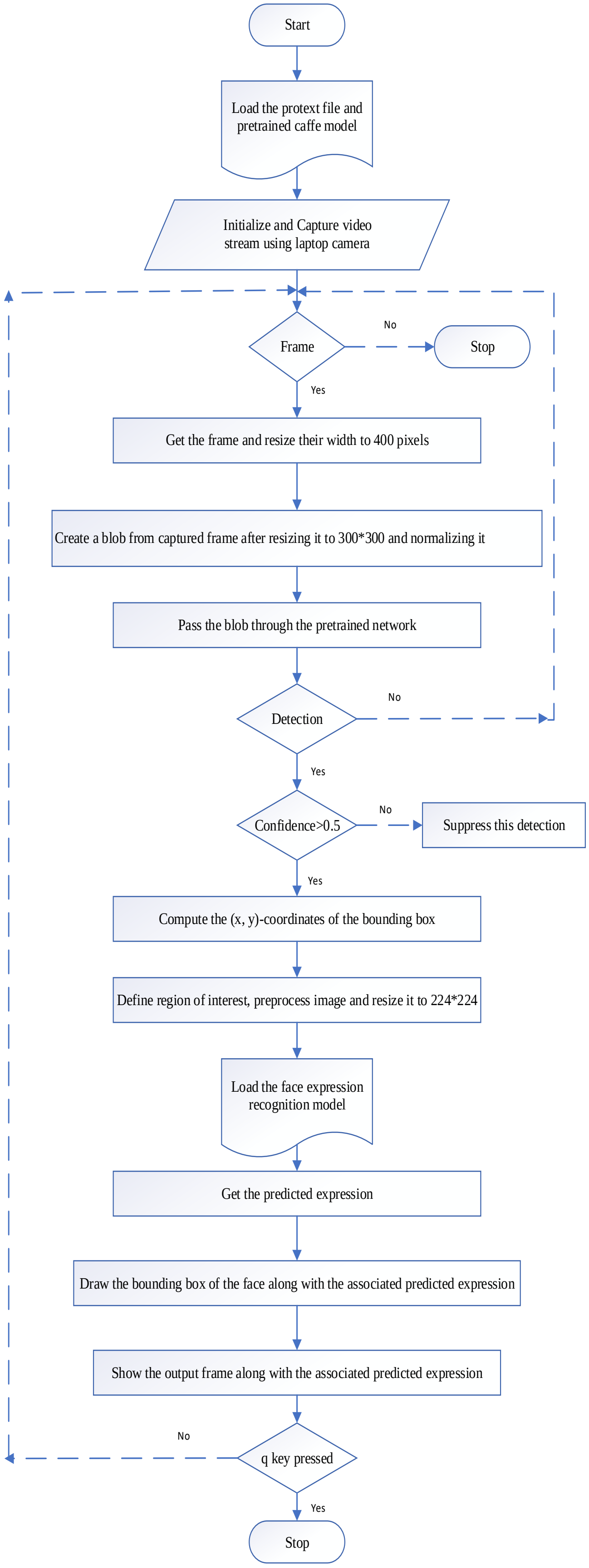

Start
Load the protext file and pretrained caffe model
Initialize and Capture video stream using laptop camera
Frame
No
Stop
Yes
Get the frame and resize their width to 400 pixels
Create a blob from captured frame after resizing it to 300*300 and normalizing it
Pass the blob through the pretrained network
Detection
No
Yes
Confidence>0.5
No
Suppress this detection
Yes
Compute the (x, y)-coordinates of the bounding box
Define region of interest, preprocess image and resize it to 224*224
Load the face expression recognition model
Get the predicted expression
Draw the bounding box of the face along with the associated predicted expression
Show the output frame along with the associated predicted expression
No
q key pressed
Yes
Stop

Fig. 13. Flowchart of Deep Learning-based Approach.

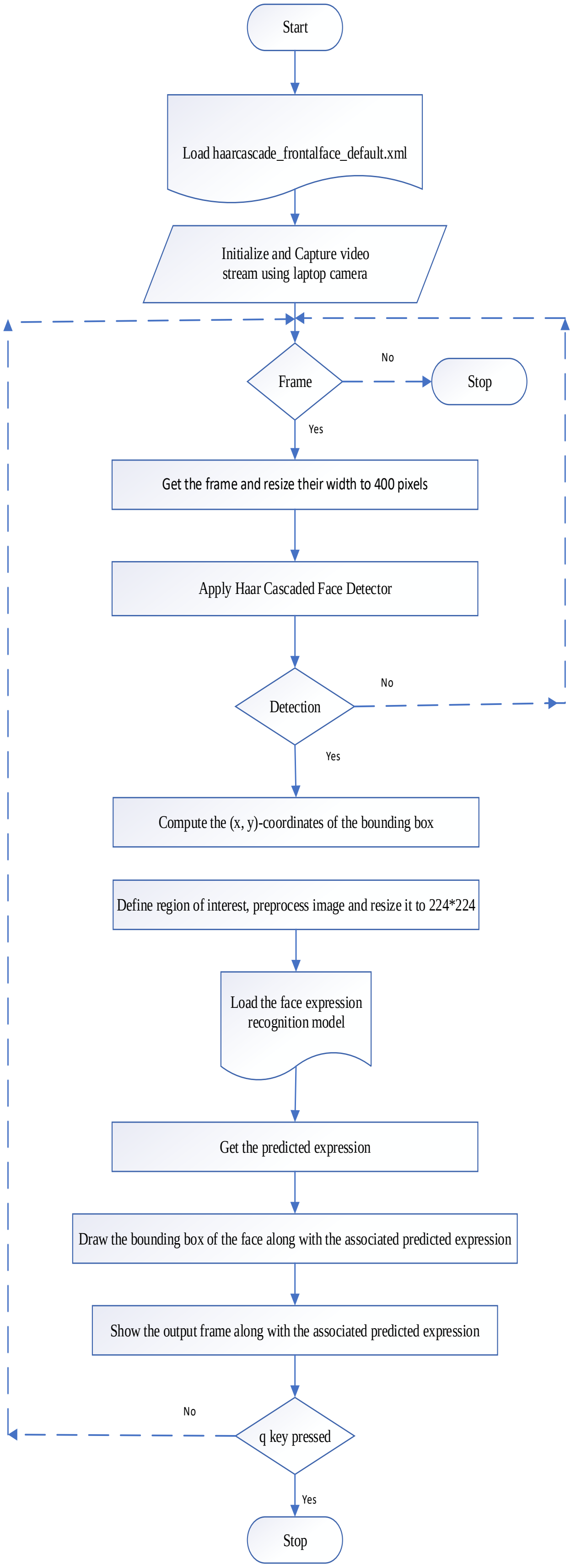

Start
Load haarcascade_frontalface_default.xml
Initialize and Capture video stream using laptop camera
Frame
No
Stop
Yes
Get the frame and resize their width to 400 pixels
Apply Haar Cascaded Face Detector
Detection
No
Yes
Compute the (x, y)-coordinates of the bounding box
Define region of interest, preprocess image and resize it to 224*224
Load the face expression recognition model
Get the predicted expression
Draw the bounding box of the face along with the associated predicted expression
Show the output frame along with the associated predicted expression
No
q key pressed
Yes
Stop

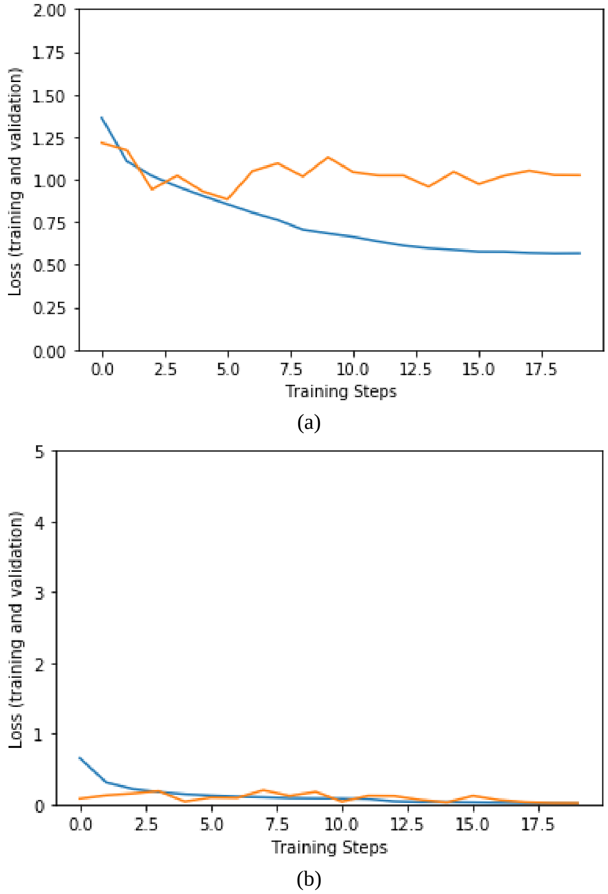
Fig. 14. Flowchart of Machine Learning (Haar Cascaded) based Approach.

IV. Results and Discussion

The designed FER transfer learning model has been tested on both the original FER2013 dataset and the modified balanced version of FER2013 dataset. Fig. 15 and Fig. 16 show the robustness of the model and the effectiveness of the modified balanced dataset. It is seen that the test loss approaches zero and the overall test accuracy reaches 91.76%, which is a state-of-the-art result, for our model using the modified balanced FER2013 dataset. However, the test loss increases, and the average overall test accuracy drops to 64% for our model using the traditional FER2013 dataset despite using dropout layer and a dense layer with small number of neurons to prevent model overfitting.

Results show that our approach starting from dataset cleaning, re-labelling, and solving class imbalance problem by generating new samples of the minority class using CycleGAN is a promising scheme.

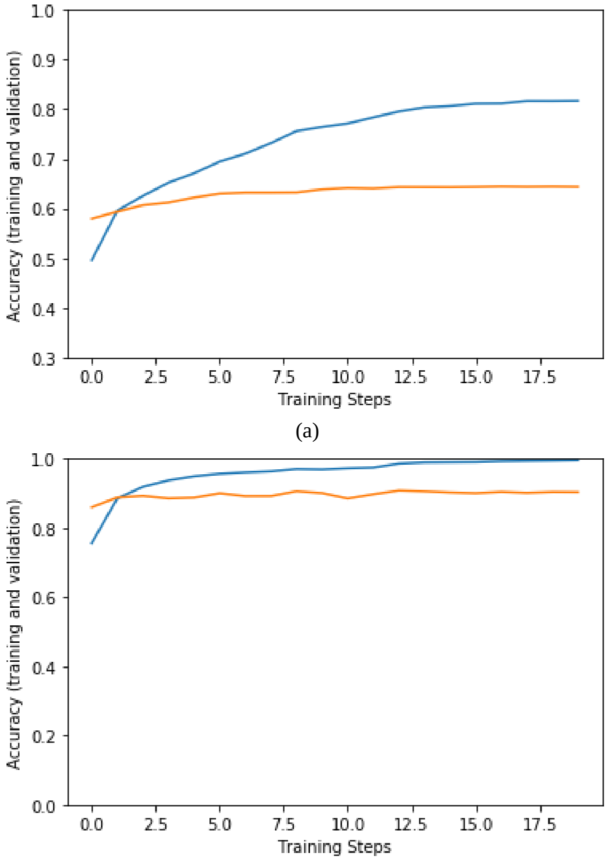
Fig. 17 shows the normalized confusion matrix for our model using the original FER2013 dataset and the modified balanced version of FER2013 dataset. It is seen that the class recognition rate has improved remarkably using the modified balanced FER2013 dataset. The recognition rate is higher by 35% for the angry class, 71% for the disgust class, 50% for the fear class, 14% for the happy class, 34% for the sad class, 21% for the surprised class, and 29% for the neutral class.



(a)

(b)

Fig. 15. Loss Curve. (a) Original FER2013 Dataset and (b) Modified Balanced FER2013 Dataset.



(a)

(b)

Fig. 16. Learning Curve. (a) Original FER2013 Dataset and (b) Modified Balanced FER2013 Dataset.

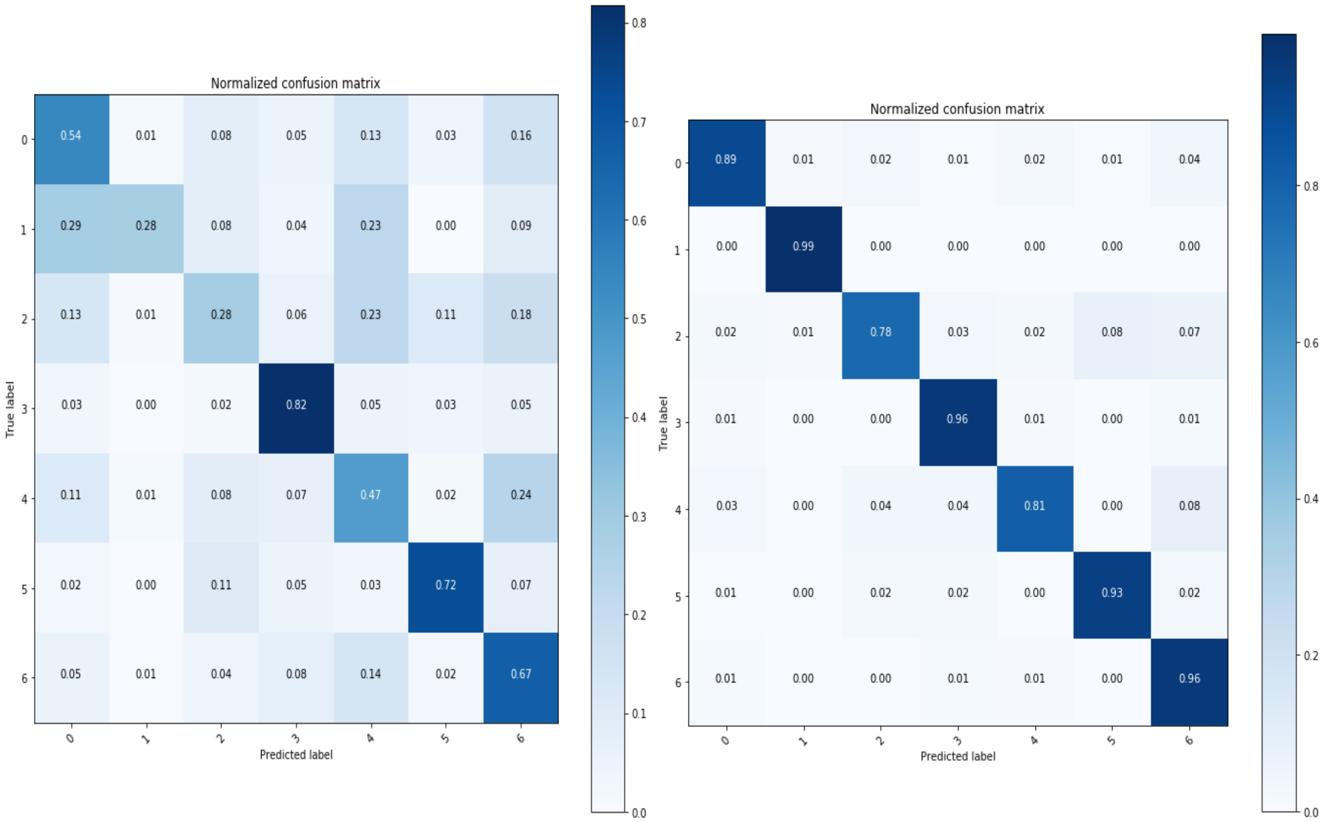
TABLE I COMPARISON OF THE STATE OF THE ART ACCURICES ON FER-2013 DATASET

Methodology	**Test Accuracy**
Zhang et al [12]	75.1%
Christopher Pramerdorfer et al [9]	75.2%
Amil Khanzada et al [10]	75.8%
An Chen et al [11]	78.61%
Barsoum et al [8]	84.986%
Our approach	**91.76%**

Table I summarizes the top test accuracies of recent approaches applied to FER-2013 dataset, where our approach achieves state of the art results.

To test our model in real time, Multi-task Cascaded Convolutional Networks (MTCNN) [19] has been used to detect the face prior to recognizing the type of the expression. Fig. 18 shows results.

In this stage, the compiled OpenCV DNN module is ready to be used. Four experiments have been conducted using Keras with TensorFlow backend on Alien ware laptop with NVidia GeForce GTX 1070 GPU. The first and third experiments used CPU as backend. On the other hand, the second and forth experiments utilized GPU as backend.



(a)

(b)

Fig. 17. The Confusion Matrix. (a) Original FER2013 Dataset and (b) Modified Balanced FER2013 Dataset.

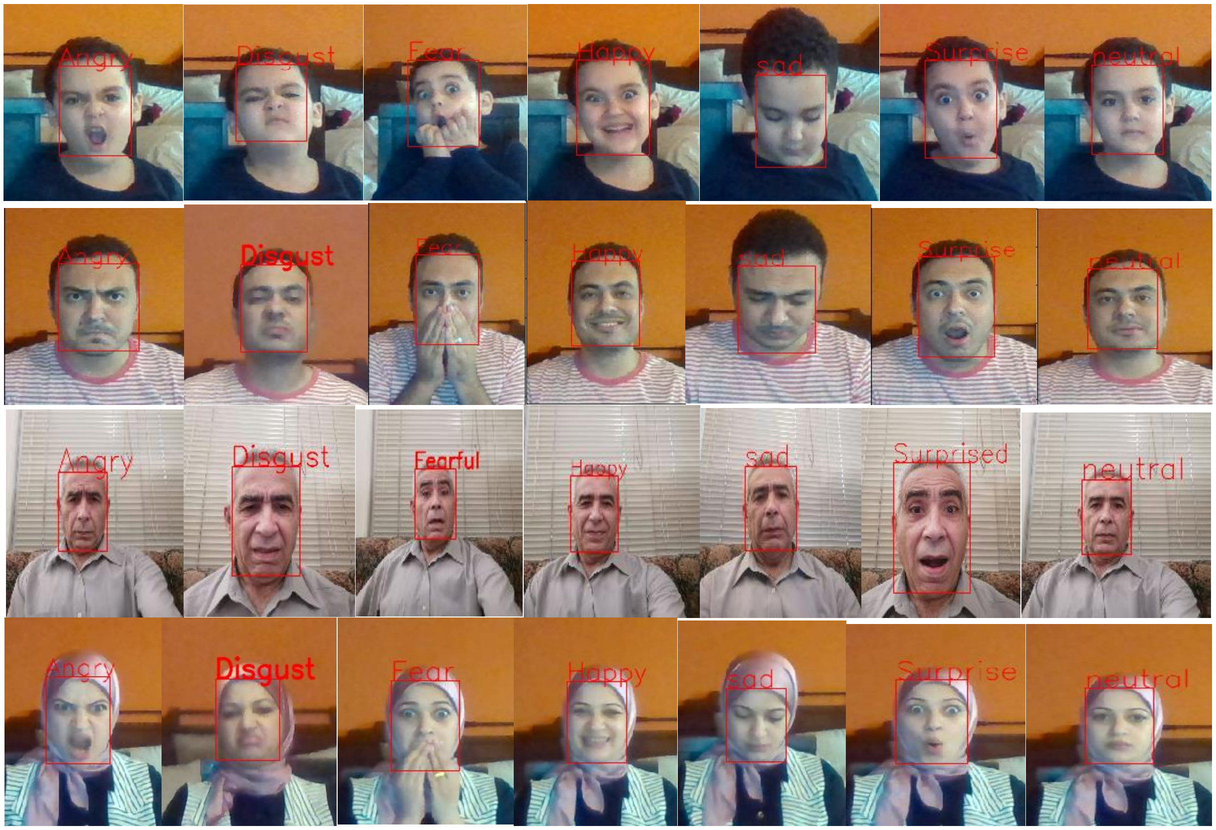


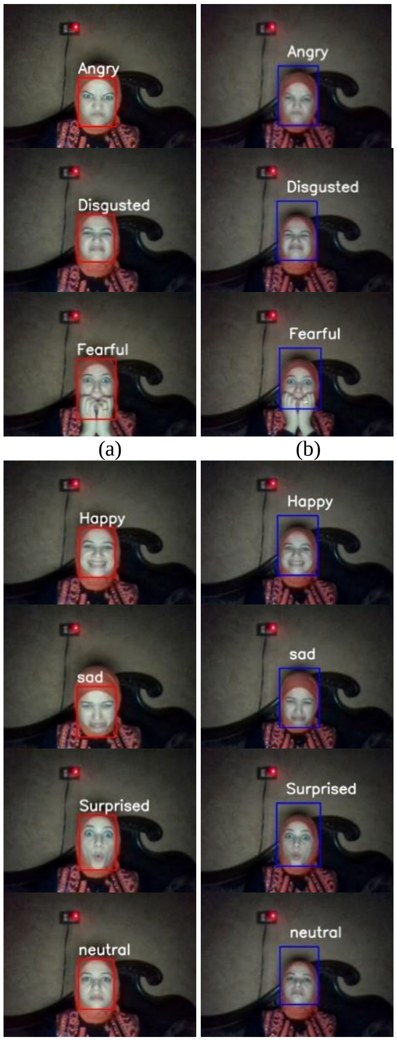
Fig. 18. Real Time Face Expression Recognition for different Subjects Showing the Seven Principal Expressions.

TABLE II FRAME PER SECOND OF THE PROPOSED METHODOLOGIES USING CPU/GPU BACKEND

The Proposed Methodology	Face Detection	FPS
1	Haar cascade classifier	7.41
2	Haar cascade classifier	23.12
3	deep learning-based approach	30.30
4	deep learning-based approach	51.43

As shown in Table II, machine learning-based methodology which used Haar cascaded classifier in face detection phase has been tested on CPU and GPU and the frame per second (FPS) has been compared. Using NVIDIA GPU has improved inference speed by up to 312.01%.

Deep learning-based methodology which used deep learning in face detection phase has also been tested on CPU and GPU and FPS has been compared. FPS throughput rate is improved by over 169.74% with OpenCV's DNN module and an NVIDIA GPU.



(a) (b)

(a) (b)

Fig. 19. The Output of Real Time FER System. (a) Deep Learning-based Approach and (b) Feature-based Approach.

Fig. 19 displays the output of FER system for real time video sequence using deep learning and Haar cascaded classifier in face detection phase, respectively.

V. CONCLUSION AND FUTURE WORK

A deep neural network model based upon VGG-Face pre-trained framework for face expression recognition has been implemented. An efficient stable Cycle Generative Adversarial Network, (CycleGAN), is designed and used to produce a balance relabeled FER2013 dataset. The designed model produced an optimal performance and a state-of-the-art overall recognition rate of 91.76% in only 0.44 second using Alien ware laptop with Nvidia GeForce GTX 1070 GPU. A GPU-based framework has been designed for face expression recognition in real time video stream. OpenCV library has been compiled from source to achieve the best exploitation of hardware resources (GPU). The framework involves two main stages, face detection and face expression recognition. The face detection phase has been realized through machine learning-based approach (Haar cascaded classifier) and deep learning-based approach. Face expression recognition has been accomplished using transfer learning approach. Both Methodologies have been tested with CPU and GPU as backend. The performance is evaluated through FPS of the whole process. Deep learning has been assessed to be faster and more accurate.

Future directions will focus on applying the model for real time face expression recognition in video sequences or online learning platform. The simplicity, the high recognition rate and the speed of the facial expression classification model make it appropriate for implementing a productive and profitable computer vision machine.

REFERENCES

[1] S. Yousefi, M. P. Nguyen, N. Kehtarnavaz, and Y. Cao, "Facial expression recognition based on diffeomorphic matching," Proc. - Int. Conf. Image Process, 2010, ICIP pp 4549–4552. https://doi.org/10.1109/ICIP.2010.5650670.

[2] C. Tsangouri, W. Li, Z. Zhu, F. Abtahi, and T. Ro, "An Interactive Facial-Expression Training Platform for Individuals with Autism Spectrum Disorder," in: 2016 IEEE MIT Undergraduate Research Technology Conference (URTC), 2016, pp. 1–4, https://doi.org/10.1109/URTC.2016.8284067.

[3] Z. Ying and G. Zhang, "Facial expression recognition based on NMF and SVM," Proc. - 2009 Int. Forum Inf. Technol. Appl. IFITA 2009, 3, 2009, pp 612–615. https://doi.org/10.1109/IFITA.2009.279.

[4] Y. Luo, C. M. Wu, and Y. Zhang, "Facial expression recognition based on fusion feature of PCA and LBP with SVM," Optik (Stuttg), 124, 2013, pp 2767–2770. https://doi.org/10.1016/j.ijleo.2012.08.040.

[5] C. Shan, S. Gong, and P. W. McOwan, "Robust facial expression recognition using local binary patterns," Proc. - Int. Conf. Image Process, ICIP 2, 2005, pp 370–373. https://doi.org/10.1109/ICIP.2005.1530069.

[6] A. A. Nashat, "Facial expression recognition using best tree RD-LGP encoded features and HMM," Int. J. Wavelets, Multiresolution Inf. Process, 16. 2018, https://doi.org/10.1142/S0219691318500479.

[7] Y. Wang, Y. Li, Y. Song, and X. Rong, "Facial expression recognition based on random forest and convolutional neural network," Inf. 10, 2029, pp 1–16. https://doi.org/10.3390/info10120375.

[8] E. Barsoum, C. Zhang, C. C. Ferrer, and Z. Zhang, "Training deep networks for facial expression recognition with crowd-sourced label distribution," ICMI 2016 - Proc. 18th ACM Int. Conf. Multimodal Interact, 2016, pp 279–283. https://doi.org/10.1145/2993148.2993165.

[9] C. Pramerdorfer and M. Kampel, “Facial Expression Recognition using Convolutional Neural Networks,” State of the Art., 2016.

[10] X. Wang, J. Huang, J. Zhu, M. Yang, and F. Yang, “Facial expression recognition with deep learning,” ACM Int. Conf. Proceeding Ser, 2018. https://doi.org/10.1145/3240876.3240908.

[11] A. Chen, H. Xing, and F. Wang, “A Facial Expression Recognition Method Using Deep Convolutional Neural Networks Based on Edge Computing,” IEEE Access, 8, 2020, pp 49741–49751. https://doi.org/10.1109/ACCESS.2020.2980060.

[12] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, “Learning social relation traits from face images,” Proc. IEEE Int. Conf. Comput. Vis., 2015 International Conference on Computer Vision, ICCV 2015, pp 3631–3639. https://doi.org/10.1109/ICCV.2015.414.

[13] fer2013 | Kaggle [WWW Document], n.d. URL https://www.kaggle.com/deadskull7/fer2013 (accessed 11.3.20).

[14] Goodfellow et al., “Challenges in representation learning: A report on three machine learning contests,” Neural Networks, 64, 2015, pp 59–63. https://doi.org/10.1016/j.neunet.2014.09.005.

[15] G. C. Porusniuc, F. Leon, R. Timofte, and C. Miron, “Convolutional neural networks architectures for facial expression recognition,” 2019 7th E-Health Bioeng. Conf. EHB 2019. https://doi.org/10.1109/EHB47216.2019.8969930.

[16] J. Y. Zhu, T. Park, P. Isola, and A. Efros, “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” Proc. IEEE Int. Conf. Comput. Vis., October 2017, pp 2242–2251. https://doi.org/10.1109/ICCV.2017.244.

[17] J. Brownlee, “Generative Adversarial Networks with Python,” Mach. Learn. Mastery 2019, pp 1–654.

[18] A. M. Bukar and H. Ugail, “Convnet features for age estimation,” Proc. Int. Conf. Comput. Graph. Vis. Comput. Vis. Image Process, Big Data Anal. Data Min. Comput. Intell. - Part Multi Conf. Comput. Sci. Info 2017, pp 94–102.

[19] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks,” IEEE Signal Process. Lett., 23, 2016, pp 1499–1503. https://doi.org/10.1109/LSP.2016.2603342.