International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 03, ISSN No. 2455-2143, Pages 89-93
Published Online July 2023 in IJEAST (http://www.ijeast.com)

FACIAL EXPRESSION RECOGNITION USING CONVOLUTIONAL NEURAL NETWORKS [CNN]

Akash Raghav, Nitin Tewari
Computer Science and Engg.
SRMIST, Chennai, Tamil Nadu, India

Abstract— **While humans have traditionally excelled at discerning emotions through facial expressions, achieving the same capability with a computer program has proven challenging. However, recent advancements in computer vision and machine learning have made it feasible to recognize emotions accurately.**
This paper introduces a novel facial emotion identification technique known as Convolutional Neural Networks for Facial Emotion Recognition (FERC). FERC utilizes a two-part convolutional neural network (CNN) architecture:
The paper consists of two main sections. The first section is dedicated to removing the background from the image, while the second section focuses on extracting the facial feature vector. The FERC model utilizes an expressional vector (EV) to identify five different regular facial expressions. With its single-level CNN approach, FERC offers improved accuracy compared to commonly used techniques. Moreover, prior to constructing the EV, a novel background removal strategy is implemented to address various potential challenges, such as distance from the camera. The application of FERC in emotion detection holds promise for a wide range of applications, including student predictive learning and lie detection.

Keywords— **Convolution Neural Network(CNN), Facial Emotion Recognition (FERC), expressional vector (EV)**

I. INTRODUCTION

Facial expressions play a crucial role in human social communication, encompassing both verbal and non-verbal aspects. Among non-verbal communication cues, facial expressions, including eye contact, convey significant signals. Alongside gestures and body language, they contribute to the overall non-verbal communication process. Humans naturally excel at perceiving and comprehending facial expressions, yet developing an automated system with the same level of understanding remains challenging. Various issues arise in this context, including detecting a segmented image as a valid face despite occlusions or lighting variations, handling diverse head poses, extracting facial expression information, detecting facial landmarks, and classifying expressions.

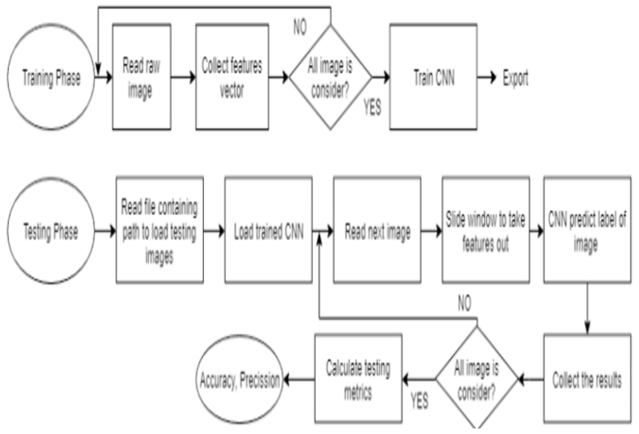
Facial Expression Recognition (FER) has emerged as a prominent area of research in artificial intelligence (AI) with wide-ranging applications. These applications span various fields such as security, surveillance, law enforcement, marketing, entertainment, e-learning, medical, emotionally intelligent robotic interfaces, and social humanoid robots. The ability to automatically recognize facial expressions holds valuable potential across domains like data analytics, psychological research, social gaming, and other domains involving human-computer interactions.

Research has confirmed that specific facial expressions hold universal meanings. As early as 1978, Ekman and Friesen developed the Facial Action Coding System (FACS) to formally categorize and describe facial expressions. Their findings revealed that pleasure, sorrow, surprise, fear, anger, and disgust are six expressions that appear to be universally recognized across cultures. In challenges such as Kaggle's Facial Expression Recognition challenge, these same emotions are used for classification, often including a seventh category for neutral expressions. However, computers face challenges when attempting to identify these common human expressions in natural settings due to variations in lighting conditions.

The objective of this research is to develop a facial expression identification system capable of classifying images into seven distinct emotion categories. The primary focus is to enhance the validation accuracy compared to existing systems and maintain consistent recognition rates across all classes. Previous experiments using convolutional neural networks (CNN) for facial expression recognition have shown inconsistent results, particularly with lower recognition rates for emotions like disgust and fear. To address this concern, our approach involves utilizing CNN along with data augmentation techniques and creating a composite dataset from various existing datasets. These modifications have led to improved validation accuracy and achieved nearly equal recognition rates for each emotion category, surpassing the outcomes of previous studies.

II. PROPOSED ALGORITHM

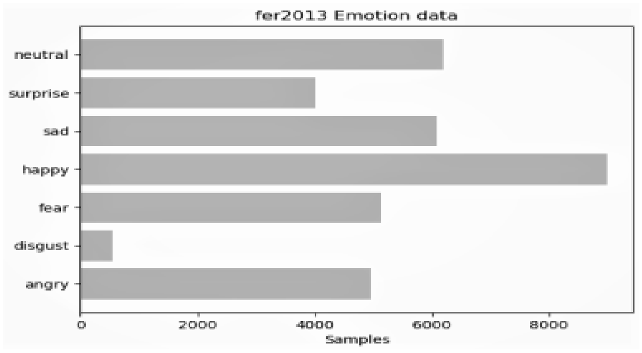
A. Proposed System Architecture



During the training process, the system was provided with grayscale face images accompanied by their corresponding labels for representation. The system then learned a set of network weights. To train the convolutional network, an input image with a face was used, followed by intensity normalization of the image. In order to ensure that the training performance remains unaffected by the order in which samples are presented, validation datasets were employed to determine the optimal weights derived from multiple training iterations with samples presented in different orders. The outcome of a training step is a set of weights that yield the best results based on the provided training data. During testing, the system employed grayscale face images from the test dataset and utilized the final network weights acquired during training to predict the corresponding expression. The output is a single numerical representation representing one of the seven basic expressions.

B. Dataset

Numerous datasets are available for emotion detection research, ranging from a few hundred high-resolution photos to tens of thousands of smaller images. Among these datasets, FERC-2013 holds significant prominence. FERC-2013 dataset comprises grayscale face images with a resolution of 48x48 pixels. The images have been automatically aligned, ensuring that the faces are approximately centered and occupy a consistent amount of space within each image. The objective is to classify each face into one of seven emotion categories based on the expressed facial expression (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral). The training set consists of 28,709 examples, while the public test set, utilized for the leaderboard, comprises 3,589 cases. The final test set, which determines the competition winner, includes an additional 3,589 examples.



C. Data preprocessing and augmentation

To detect and isolate the face region in training images for human facial expression detection, preprocessing is necessary. In this study, a Haar feature-based technique is employed to detect and extract the face region [17]. Additionally, regardless of the color information in the input image, facial expression identification using CNN is performed. This approach allows for the detection of facial expressions irrespective of skin color, based on our experience. In this article, techniques such as intensity normalization are not utilized. Instead, the input image is converted into a grayscale image with a single channel.

Figure 2 illustrates the result of identifying and isolating the facial region from the original image, followed by converting it to grayscale.

Insufficient training images in relation to the CNN training parameters can lead to overfitting and subpar classification performance. To address this issue, a data augmentation strategy is implemented to augment the number of training photos. The facial photos in the database typically capture faces with an upward-straightened neck, as depicted in Figure 2. However, in real-life scenarios, facial images can vary based on the camera's position or the individual's posture. Therefore, these factors are considered when selecting a suitable data augmentation technique.

In the first step, the reference image is rotated by 5°, 10°, and 15° in both the clockwise and counterclockwise directions. These rotated images, along with the original reference image, are horizontally flipped, resulting in a total of fourteen images. By generating synthetic images through this straightforward rotation and flipping operations, the size of the database is increased, leading to enhanced generalization of the CNN model during training. Figure 3 displays the outcome of applying this data augmentation technique.

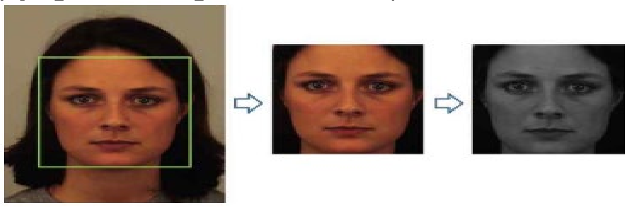


Fig. 2. The result of converting cut-out face region image into gray image

D. Architecture of CNN

A convolutional neural network (CNN) typically consists of an input layer, several convolutional layers, fully-connected layers, and an output layer. The CNN architecture employed in this project is derived from the LeNet Architecture with modifications [10]. It comprises six layers excluding the input and output layers. The specific architecture of the Convolutional Neural Network utilized in the project is illustrated in the following figure.

1. The input layer of the neural network has a predefined and fixed dimension, requiring pre-processing of the image before it can be inputted. For training, normalized grayscale images of size 48 x 48 pixels from the Kaggle dataset are utilized, while validation and testing terms are used interchangeably. Additionally, proposed laptop webcam photos are employed for testing purposes. These photos undergo face recognition, cropping, and normalization using the OpenCV Haar Cascade Classifier.

2. The Convolution and Pooling layers (ConvPool) employ batch processing, where each batch contains N photos, and the CNN filter weights are adjusted accordingly. Each convolution layer accepts a four-dimensional input batch of images, represented as N x Color-Channel x width x height. Similarly, the convolution feature maps or filters are also four-dimensional, including the number of feature maps in, the number of feature maps out, filter width, and filter height. A four-dimensional convolution operation is performed between the image batch and feature maps within each convolution layer. Following the convolution, the only parameters that undergo change are the image width and height.
New image width = old image width - filter width + 1
New image height = old image height - filter height + 1

To reduce dimensionality, down sampling or subsampling is implemented following each convolution layer. This process, known as pooling, involves dividing the image into a grid of blocks. In this project, maximum pooling is employed after convolution. Each block has a size of 2x2, and the maximum value among the four pixels within the block is selected using a pool size of (2x2). The only parameters that undergo modification after pooling are the height and width of the image.
The architecture consists of two convolution layers and a pooling layer. The first convolution layer takes an input image batch with a size of Nx1x48x48. Here, N represents the batch size, the number of color channels is 1, and both the image height and width are 48 pixels. The convolution operation with a feature map of size 1x20x5x5 yields an image batch of size Nx20x44x44.
After convolution, pooling is applied with a pool size of 2x2, resulting in an image batch of size Nx20x22x22. Subsequently, the second convolution layer employs a feature map of size 20x20x5x5, generating an image batch of size Nx20x18x18. This is followed by a pooling layer with a pool size of 2x2, resulting in an image batch of size Nx20x9x9.

3. Fully Connected Layer

Inspired by the way neurons transmit signals in the brain, this layer processes a large number of input features by utilizing trainable weights and layers. In the fully-connected layer, two hidden layers with 500 and 300 units respectively are employed. Training the weights of these layers involves forward propagation of training data and backward propagation of errors. The process of backpropagation calculates the necessary weight adjustments for each layer, followed by evaluating the difference between the predicted and true values. By adjusting hyperparameters such as learning rate and network density, the training speed and complexity of the architecture can be controlled. Hyperparameters for this layer include learning rate, momentum, regularization parameter, and decay. The output of the second pooling layer is Nx20x9x9, while the input of the first hidden layer in the fully-connected layer is Nx500. Consequently, the output from the pooling layer is flattened to Nx1620 dimensions and fed into the first hidden layer. The output of the first hidden layer is then passed to the second hidden layer. The final output layer, with a size equal to the number of facial expression classes, receives the output of the second hidden layer, which is Nx300 in size.

4. The Output Layer

The output of the second hidden layer is connected to an output layer with seven classes. To determine the outcome, the Softmax activation function is applied, which provides probabilities for each of the seven classes. The predicted class is the one with the highest probability of success.

E. Rectified Linear Unit

Each convolution operation in the architecture is enhanced with a Rectified Linear Unit (ReLU) operation. ReLU is a neural network cell that calculates its output based on the activation function applied to the input x:
Max R(x) = R(x) = R(x) = R(x (0,x)

Compared to binary units, these cells are more efficient than sigmoid units and transmit more information. In the case of even weight initialization, approximately half of the weights are negative. This characteristic helps in highlighting sparse features.
Another benefit is the relatively low computational cost associated with these cells. There is no need to calculate exponential functions, which can be computationally expensive. Moreover, this function helps prevent the issue of vanishing gradients by ensuring that the gradients do not diminish. It is worth noting that this function behaves linearly or as zero, but it does not exhibit non-linear behavior.

F. Classification (Multilayer Perceptron):

The output layer incorporates a Fully Connected layer, which is a conventional Multi-Layer Perceptron with a softmax

International Journal of Engineering Applied Sciences and Technology, 2023
Vol. 8, Issue 03, ISSN No. 2455-2143, Pages 89-93
Published Online July 2023 in IJEAST (http://www.ijeast.com)

activation function. As the term "Fully Connected" suggests, every neuron in the preceding layer is connected to each neuron in the subsequent layer. The output from the convolutional and pooling layers captures the high-level characteristics of the input image. The purpose of the Fully Connected layer is to leverage these features to classify the input image into different categories, guided by the training dataset.

The activation function utilized in this context is Softmax, where the outputs are considered as scores assigned to each class. The Softmax function preserves the mapping and these scores are interpreted as unnormalized log probabilities for each class. The formula for Softmax is as follows:

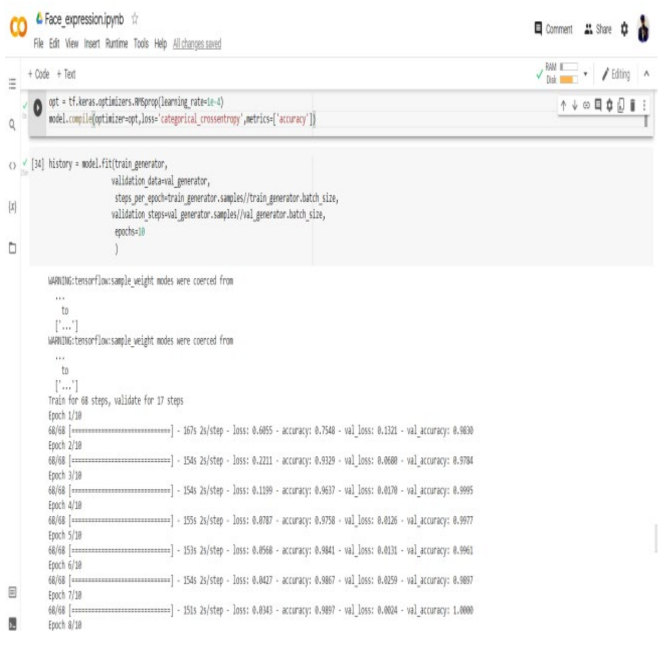
$f(z)j = exp(zj) / \sum expKk{=}1\ (zk)$

where j is the image index and K is the total number of facial expression classes.

In addition to its classification capabilities, the inclusion of a fully-connected layer offers an (typically) cost-effective means to learn non-linear combinations of diverse features. While the information extracted from convolutional and pooling layers is often valuable for classification purposes, combining them can further enhance the effectiveness of feature representation.

G. The output probabilities of the Fully Connected Layer are normalized so that their sum is equal to 1. This normalization is accomplished by utilizing the Softmax function as the activation function in the output layer of the Fully Connected Layer. The Softmax function transforms a vector of arbitrary real-valued scores into a vector of values ranging from zero to one, ensuring that the sum of these values adds up to one

IV.CODING AND TESTING



```
opt = tf.keras.optimizers.RMSprop(learning_rate=1e-4)
model.compile(optimizer=opt,loss='categorical_crossentropy',metrics=['accuracy'])
```

```
[34] history = model.fit(train_generator,
                     validation_data=val_generator,
                      steps_per_epoch=train_generator.samples//train_generator.batch_size,
                     validation_steps=val_generator.samples//val_generator.batch_size,
                      epochs=10
                      )
```

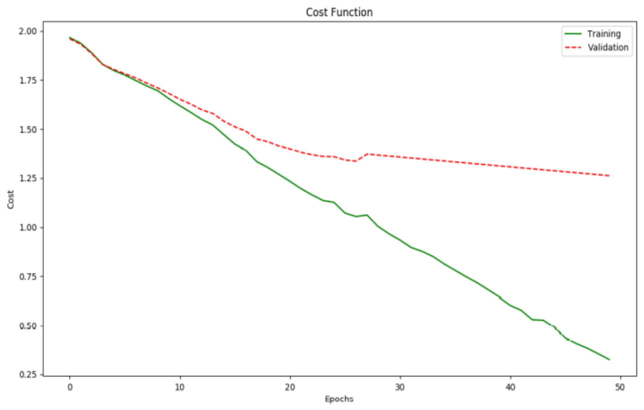
```
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
WARNING:tensorflow:sample_weight modes were coerced from
  ...
    to
  ['...']
Train for 68 steps, validate for 17 steps
Epoch 1/10
68/68 [==============================] - 167s 2s/step - loss: 0.6055 - accuracy: 0.7548 - val_loss: 0.1321 - val_accuracy: 0.9830
Epoch 2/10
68/68 [==============================] - 154s 2s/step - loss: 0.2211 - accuracy: 0.9329 - val_loss: 0.0680 - val_accuracy: 0.9784
Epoch 3/10
68/68 [==============================] - 154s 2s/step - loss: 0.1199 - accuracy: 0.9637 - val_loss: 0.0170 - val_accuracy: 0.9995
Epoch 4/10
68/68 [==============================] - 155s 2s/step - loss: 0.0787 - accuracy: 0.9758 - val_loss: 0.0126 - val_accuracy: 0.9977
Epoch 5/10
68/68 [==============================] - 153s 2s/step - loss: 0.0568 - accuracy: 0.9841 - val_loss: 0.0131 - val_accuracy: 0.9961
Epoch 6/10
68/68 [==============================] - 154s 2s/step - loss: 0.0427 - accuracy: 0.9867 - val_loss: 0.0259 - val_accuracy: 0.9897
Epoch 7/10
68/68 [==============================] - 151s 2s/step - loss: 0.0343 - accuracy: 0.9897 - val_loss: 0.0024 - val_accuracy: 1.0000
Epoch 8/10
```

The CNN architecture for facial expression recognition, as described earlier, was implemented using the Python programming language. The implementation also utilized the Numpy, Theano, and CUDA libraries alongside Python.

For both convolution layers, a training image batch size of 30 was utilized, and the filter map size was set to 20x5x5. The training process incorporated a validation set to verify its effectiveness. During each epoch, the validation cost, validation error, training cost, and training error were computed in the last batch. The training procedure involved providing the image set and output labels as input parameters.

The weights of the feature maps and hidden layers were adjusted during training based on hyperparameters such as learning rate, momentum, regularization, and decay. In this system, a batch-wise learning rate of 10e-5, momentum of 0.99, regularization of 10e-7, and decay of 0.99999 were employed.

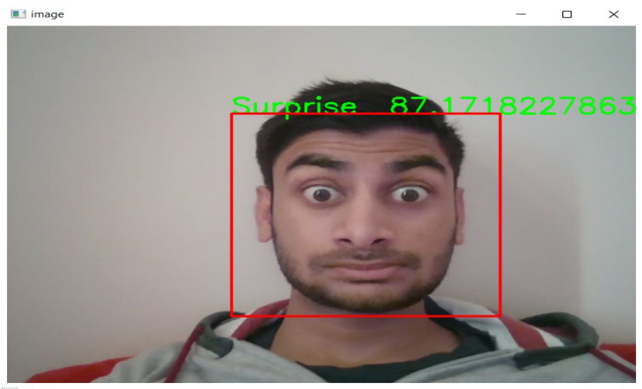
The figures below present the comparison of validation cost, validation error, training cost, and training error.

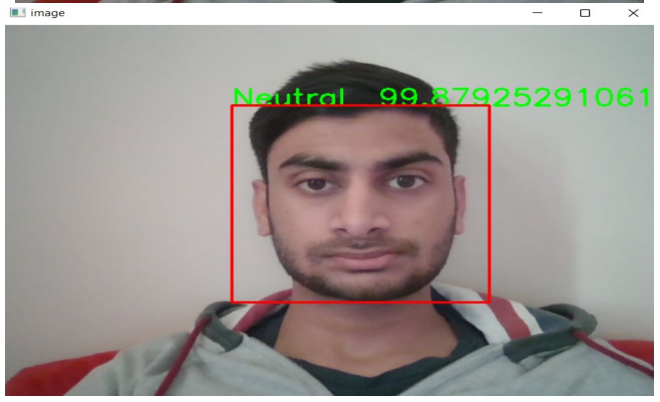


The model was tested using 6000 images, resulting in an accuracy of 56.77% provided by the classifier. The confusion matrix for the seven facial expression classes is displayed below:

Predicted Actual	Anger	Disgust	Fear	Happy	Sad	Surprise	Neutral
Anger	312	16	93	89	97	27	106
Disgust	35	700	0	0	0	0	0
Fear	123	7	300	69	100	67	117
Happy	85	5	74	887	80	29	128
Sad	137	6	122	89	338	27	165
Surprise	27	4	75	45	24	402	46
Neutral	114	4	91	116	123	32	467

The tables above indicate that the model achieved the highest accuracy for the disgust emotion at 95.23%, followed by happiness at 68.86%, surprise at 64.52%, neutral at 49.31%, anger at 42.16%, fear at 38.31%, and the lowest accuracy for the sad emotion at 38.23%.





V.CONCLUSION

This research employed a six-layer convolutional neural network inspired by the LeNet architecture to classify various human facial expressions, including happiness, sadness, surprise, fear, anger, disgust, and neutrality. The system's performance was evaluated using metrics such as accuracy, precision, recall, and F1-score. The classifier achieved an accuracy of 56.77 percent, with precision, recall, and F1-score all measuring 0.57.

In the future, the concept can be extended to include color photos, enabling researchers to assess the effectiveness of pre-trained models for facial emotion identification, such as AlexNet [11] or VGGNet [12]. This expansion would open up new possibilities for evaluating and enhancing the performance of face emotion identification systems

VI. REFERENCE

[1]. Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., and Taylor. (2001) “Emotion recognition in human computer interaction " IEEE Signal Processing magazine, Vol. 18, No. 1: 32-80Jannat, R., Tynes, I., Lime, L. L., Adorno, J., and Canavan, S. Ubiquitous emotion recognition using audio and video data. In Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers (2018), ACM, pp. 956–959.

[2]. T. F. Cootes, C. J. Taylor, et al. Statistical models of appearance for computer vision, 2004.

[3]. Ross Girshick. (2015) “Fast R-CNN.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA, CVPR.15: 1440-1448Logan, B., et al.: Mel frequency cepstral coefficients for music modeling. In ISMIR (2000), vol. 270, pp. 1–11

[4]. A. Gudi. Recognizing semantic features in faces using deep learning. arXiv preprint arXiv:1512.00743, 2015.

[5]. Zhang L, Liu J, Zhang B, et al. 2019 Deep Cascade Model-Based Face Recognition: When Deep-Layered Learning Meets Small Data. IEEE Transactions on Image Processing. 29 1016–1029.

[6]. S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167, 2015.

[7]. Jonathan Long, Evan Shelhamer, and Trevor Darrell. (2015) “Fully Convolutional Networks for Semantic Segmenta.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA. CVPR.15: 3431-3440Toronto emotional speech set (TESS)(https://tspace.library.utoronto.ca/handle/1807/24487

[8]. Gao S, Zhang, Y, Jia K, et al. 2015 Single Sample Face Recognition via Learning Deep Supervised Autoencoders. IEEE Transactions on Information Forensics and Security. 10 2108–2118..

[9]. Bettadapura, Vinay (2012). Face expression recognition and analysis: the state of the art. arXiv preprint arXiv:1203.6722A. Milton, S. Sharmy Roy, and S. Tamil Selvi, “SVM scheme forspeech emotion recognition using MFCC feature,” International JournalMof Computer Applications (0975–8887), vol. 69, no. 9, pp. 34–39, May 2013.

[10]. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[11]. Bettadapura, Vinay (2012). Face expression recognition and analysis: the state of the art. arXiv preprint arXiv:1203.6722A. Milton, S. Sharmy Roy, and S. Tamil Selvi, “SVM scheme forspeech emotion recognition using MFCC feature,” International JournalMof Computer Applications (0975–8887), vol. 69, no. 9, pp. 34–39, May 2013

[12]. Samson C, Blanc-Féraud L, Aubert G, et al. 2000 A Level Set Model for Image Classification. International Journal of Computer Vision. 40 187-197.