Detecting Macro-Facial Expressions Using Convolutional Neural Network(CNN)

Dolapo K. Adebo*
*Department of Computing and Mathematics
†Faculty of Science and Engineering, Manchester Metropolitan University, Manchester, UK
‡dolapo.k.adebo@stu.mmu.ac.uk

***Abstract*—This paper presents a deep learning-based method for macro-facial expression detection. The study compares the performance of two existing convolutional neural network (CNN) models, VGG11 and ResNet50, with a proposed model using a dataset consisting of three emotion classes from FER2013 (Happy, Neutral, and Sad). The evaluation of the proposed macro-facial expression detection system is conducted based on validation accuracy, validation loss, optimizers, and learning rate. The models are trained on the relevant dataset, and their performance is analyzed to determine their effectiveness in recognizing facial emotions. The results show that ResNet50 and VGG11, when trained with the Adam optimizer, achieved validation accuracy scores of 72.6% and 71.4%, respectively. The proposed model, utilizing the SGD optimizer, achieved a validation accuracy of 67.8%.**

Link to the Code.

***Index Terms*—Macro-Facial Expressions, Convolutional Neural Networks, Deep Learning, Artificial Intelligence.**

I. Introduction

Emotions are complex psychological and physiological experiences that play an essential role in human life [1]. Facial expressions serve as non-verbal cues to communicate emotions and play a vital role in distinguishing different emotional states in humans. They are evident and easily noticeable, making them a prominent means of expressing and recognizing emotions. There are seven basic emotions: Happy, Sad, Disgust, Fear, Neutral, Surprise, and Angry. In the interactions between people and computers, facial emotions are crucial due to their significant impact on communication, user experience, and the overall effectiveness of interactive systems [2] [3].

Among the deep learning models, Convolutional Neural Networks (CNNs) have indeed demonstrated exceptional performance in image classification tasks and have become widely used in the field. CNNs are specifically designed to analyze visual data, making them highly effective for tasks such as object recognition, image segmentation, and pattern detection. In this work, we aim to build a CNN model on the FER2013 dataset using only three basic emotions: Happy, Neutral, and Sad, that identify facial expressions We intend to design a range of experiments to investigate diverse optimization algorithms. on three CNN models(ResNet50, VGG11, and a Self-built architecture).

II. Related Work

Convolutional Neural Networks (CNNs) are widely used in image processing and consist of key components such as convolutional layers, pooling layers, and fully connected layers. These components enable efficient manipulation of static images. In the past decade, the advancement in computational capabilities and the accessibility of larger datasets brought about a significant transformation in the practicality of CNNs for feature extraction and image classification. This growth in computing power facilitated the computational demands of CNN models, allowing for more complex and accurate image analysis. Additionally, the availability of larger datasets provided more prosperous and more diverse training samples, leading to improved performance and generalization capabilities of CNNs [4].

Numerous methodologies have been suggested to enhance the performance of CNNs. One notable approach involves replacing the Sigmoid activation function with the Rectified Linear Unit (ReLU) activation function. This substitution aims to address issues related to gradient dispersion, leading to accelerated training [5]. The integration of ReLU activation into CNN architectures has proven beneficial in various image classification tasks, where it has demonstrated improved accuracy and reduced training time compared to the traditional Sigmoid activation [6]. Various pooling techniques, such as average pooling and max pooling, are employed to downsample the inputs and assist in enhancing generalization. [7].

Akriti et al. in 2020, presented a paper that addresses the challenging task of human emotion detection from images in social communication. It highlights the superiority of deep learning (DL) methods over traditional image processing techniques. The proposed AI system utilizes a convolutional neural network (CNN) architecture for emotion detection. The performance of the proposed method is evaluated on two datasets, FERC-2013 and JAFFE, achieving accuracies of 70.14% and 98.65% respectively [8].

Yousif et al. in 2021, adopted the VGGNet architecture and meticulously fine-tune its hyperparameters while experimenting with various optimization methods. The model achieved a state-of-the-art single-network accuracy of 73.28% on the FER2013 dataset, without resorting to the utilization of additional training data. The state-of-art- performance was achieved by using the SGD Nesterov optimizer with a Decaying Learning Rate on VGG34 architecture [9]. However, in the scope of this paper, we are only going to work with three classes, i.e Happy, Neutral, and Sad.

III. Methodology

A. Datasets Description and Analysis

The FER2013 dataset, publicly available on Kaggle, is introduced in this section, along with the recommended approach of splitting it into training, validation, and test sets. This dataset comprises 35,888 images depicting seven distinct emotions: Happy, Angry, Sad, Neutral, Fear, Disgust, and Surprise. However, for the purpose of the study discussed earlier, the focus was narrowed down to three specific classes: Happy, Neutral, and Sad.

1) Fuzzy Algorithm: The fuzzy algorithm; dupeGuru was used to remove similar images. A predefined similarity threshold of 95% was established, indicating that the application would identify and classify images exhibiting a similarity level of at least 95%. The application was specifically applied to the selected classes, namely Happy, Neutral, and Sad.

Remarkably, the analysis conducted through the application resulted in the identification of a total of 794 images that displayed a similarity level exceeding the established threshold. These identified images were subsequently removed from the dataset, ensuring that only distinct and dissimilar images were retained for further analysis and evaluation.

This methodological approach, involving image processing in RGB bitmap mode and the application of a stringent similarity threshold, contributes to the robustness and accuracy of the subsequent analysis. By identifying and eliminating highly similar images, potential redundancies and biases within the dataset are mitigated, enhancing the reliability and generalizability of the findings.

2) Dataset Description: The training set contains 11,020 images, with 4,762 categorized as HAPPY, 3,142 as NEUTRAL, and 3,116 as SAD. For the purpose of validation, a separate folder was created, containing 3,672 images. Among these, 1,587 were classified as HAPPY, 1,047 as NEUTRAL, and 1,038 as SAD.

In the test set, a folder named ”Unknown” was created, housing 3,677 images. This folder encompasses all three classes, with 1,588 images classified as HAPPY, 1,049 as NEUTRAL, and 1,040 as SAD. It is worth noting that the dataset's three-class composition allows for the exploration of multiclassification problems using deep learning techniques.

By adhering to this data-splitting methodology, researchers can ensure a rigorous evaluation of their models, enabling them to gauge their performance accurately. This standardized approach facilitates meaningful comparisons and enhances the reproducibility of results in the field of facial and emotion recognition. Figure 1 shows some samples of the images.

3) Ground Truth Annotation: To generate the ground truth for evaluation, the test images were processed using a specific procedure. Initially, each image in the test folder was split based on the occurrence of the ”.jpg” extension. Subsequently, the image was renamed by prefixing it with the name of the folder it belonged to.

To assign class labels to the renamed images, a lambda function was employed. This function excluded the ”.jpg” extension from the image name and checked if the folder name was present within it. If the folder name was found, the class column was appended with a value of 1. Conversely, if the folder name was absent, a value of 0 was appended to the class column.

This process ensured the creation of a ground truth dataset that associated each image with its corresponding class label. By incorporating the folder name information into the image names and utilizing the lambda function for a class assignment, a concise and effective approach was implemented to facilitate subsequent evaluation and analysis.

Fig. 1. Samples Representing the 3 Classes: Happy, Neutral, and Sad

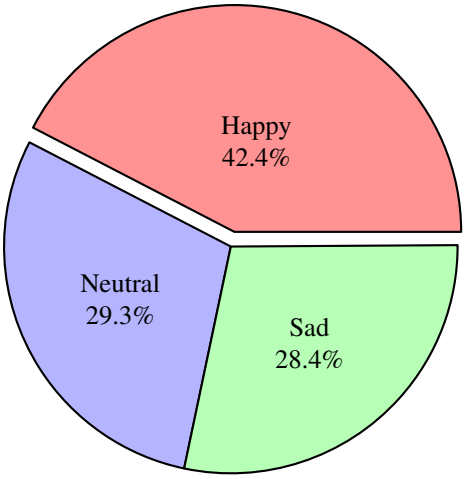


Fig. 2. Distribution of Image Classes

IV. Experiments

A. Data Augmentation

To account for the variability in facial expression recognition, we apply a significant amount of data augmentation in training. This augmentation includes rescaling the images up to 48 x 48 pixels

The augmentation process applied to the dataset involved the utilization of specific transformations to enhance the training and testing procedures. The transformations were applied to the images in a standardized manner to ensure consistency and improve the model's generalization capability. The images were resized to a dimension of 48x48 pixels, maintaining the aspect ratio of the original images, a random horizontal flip was applied to the images, introducing variations in orientation to enhance the model's ability to handle different orientations, each channel of the image was standardized using precomputed mean and standard deviation values of [0.4907, 0.4907, 0.4907] and [0.2099, 0.2099, 0.2099] respectively. This normalization step aided in reducing the effect of varying pixel intensities and ensured consistent data distribution. Similarly, for the testing dataset, the same transformations were applied, excluding the RandomHorizontalFlip operation. This ensured that the testing images underwent a consistent resizing, tensor conversion, and normalization process, aligning with the training data.

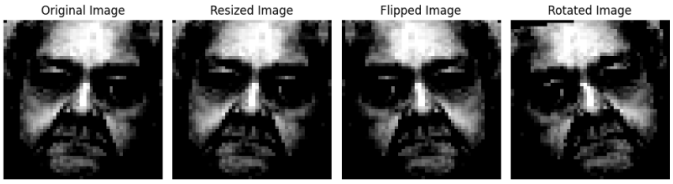


Fig. 3. Examples of three different data augmentation techniques: resize, horizontal flip, and rotation.

B. ResNet50

Residual Networks (ResNet) are a type of convolutional neural network architecture that tackle the challenge of training very deep networks. Unlike traditional networks that aim to learn unreferenced functions, ResNet learns residual functions by making predictions relative to the input of each layer. The residual refers to the discrepancy or difference between the predicted output and the actual value. Rather than directly skipping layers in the network, ResNet allows these layers to learn residual mappings, enabling them to fit the underlying desired mapping more effectively. Residual blocks, which consist of stacked layers, form the building blocks of ResNet models.

One specific ResNet model, ResNet-50, exemplifies this architecture with fifty layers comprising these residual blocks. The classification process of ResNet-50 involves initial convolution operations on the input, followed by the application of residual blocks, and concluded with fully connected operations. In convolutional neural networks, it is common to end with fully connected layers that summarize the characteristics learned from the preceding connected layers. These fully connected layers can be seen as a way to assign weights to features learned by earlier convolutional and pooling layers, thereby performing feature engineering, amplification, and extraction at a local level.

The structure of the fully connected layer enables the model to quickly learn non-linear combinations of advanced attributes generated by the preceding convolutional layers. This layer learns a non-linear function that processes the flattened image, which has been appropriately formatted into column vectors. The flattened data is then passed through a feed-forward neural network in each training iteration. This approach allows the model to distinguish between low-level aspects and the main features of the image, ultimately classifying them using methods such as SoftMax. The output of the categorization process provides the classification results for the three expressions [10] [11].

C. VGG11

VGG, which stands for Visual Geometry Group, is a convolutional neural network architecture that was developed to address the depth of CNNs. Pre-trained models like VGG have been trained on specific datasets and contain weights that represent the learned features of the dataset they were trained on. Utilizing a pre-trained model saves time and computational resources, as a considerable amount of effort has already been dedicated to learning numerous features, which can benefit subsequent tasks [12].

VGG-11 specifically refers to a VGG model with 11 weighted layers. VGG models typically take RGB images with a resolution of 224 x 224 pixels as input. The choice of 224 pixels is to ensure a consistent image size throughout the dataset, disregarding the full range of pixel values (0 to 255) in each color channel. VGG11, is composed of convolutional layers with a 3 x 3 kernel size. The ReLU activation function is employed within the convolutional layers. Max pooling layers are utilized to return the pixel with the maximum value from a set of pixels within a kernel. The convolution stride, denoting the step size of the convolutional network, is fixed at 1 pixel. The choice of ReLU activation function in VGG models offers several advantages, including reducing the computational cost for training larger networks, and enabling the incorporation of more parameters at the same computational expense [13]. Fully-connected layers in VGG-11 consist of three nodes. The upper two fully-connected layers have 4096 channels, as visually represented in the corresponding diagram. The third fully-connected layer contains 1000 channels, aligning with the number of classes in the classification task.

D. Self-Built Model

The Self-Built model is a convolutional neural network architecture designed for image classification tasks. The model consists of multiple layers, including convolutional layers, batch normalization layers, pooling layers, and fully connected layers. The input to the network is a 3-channel image, and

the first convolutional layer applies a 5x5 kernel to extract local features. Batch normalization is applied after the first convolutional layer to improve training stability. Max pooling is then performed to reduce the spatial dimensions of the feature maps.

The second convolutional layer further extracts high-level features from the input, followed by another round of batch normalization and max pooling. The resulting feature maps are flattened to a 1-dimensional vector. The flattened features are then passed through two fully connected layers. The first fully connected layer has 4096 units and applies a rectified linear unit (ReLU) activation function. The second fully connected layer has 512 units and also applies ReLU activation. The final layer of the model is a fully connected layer with 3 output units, corresponding to the number of classes in the classification task. This layer produces the predicted class probabilities for the input image.

E. Performance Metrics

The FER2013 dataset exhibits class imbalance, where the distribution of classes is uneven. In order to address this issue effectively, performance evaluation should include metrics such as The evaluation metrics used in this study include per-class F1-Score, micro-average F1-Score, and the area under the Receiver Operating Characteristics Curve (AUC). These metrics are employed to assess the performance of the models and provide a comprehensive analysis of their classification accuracy, overall effectiveness, and discriminative capabilities. Additionally, the macro-average of Precision, Recall, F1-Score, and AUC is recommended to assess the overall performance, particularly in imbalanced multi-class scenarios. This approach is well-suited for situations where there is a significant class imbalance, as it takes into account the performance of individual classes. It should be noted that while the micro-average provides an overall performance measure, the macro-average is more suitable for imbalanced datasets [14]. The micro-average is calculated by considering the total number of true positives, false negatives, and false positives across all classes. It treats the dataset as a whole and computes the metrics by aggregating the counts of true positives, false negatives, and false positives from all classes. It gives equal weight to each sample in the dataset, regardless of its class label. On the other hand, the macro-average is calculated by taking the average of the precision, recall, and F1-score for each class. It treats each class equally, regardless of its representation in the dataset. It computes the metrics separately for each class and then takes the average, giving equal weight to each class. The micro-average and macro-average scores provide different perspectives on the overall performance of the classification model. The micro-average is influenced by the class with the largest number of samples, while the macro-average treats each class equally. F1-Score is a commonly used performance metric in classification tasks that takes into account both precision and recall. It provides a balanced measure of the model's accuracy by considering both the ability to correctly identify positive instances (precision) and the ability to capture all positive instances (recall), as in equation (1).

$$\text{macro-average F1-Score} = \frac{1}{N}\sum_{i}^{N} F1 - Score_i \qquad (1)$$

For each class, denoted by $i = 1, .., N$ where N represents the total number of classes (in this case, $N = 3$), the results will be compared based on the macro-average F1-Score. Additionally, to ensure a comprehensive scientific assessment, other metrics such as macro-average recall (also known as Unweighted Average Recall or UAR) and macro-average AUC will be discussed. These metrics offer a balanced evaluation of an approach's ability to predict all classes equally well, mitigating the risk of an approach being tailored to perform well only on specific classes.

V. Experiments

To benchmark the performance of the networks, we used the input size of 488 pixels, train the model at a learning rate of 0.001 and decrease with a factor of 0.1 if the validation score does not decrease after 8 epochs. For all the models, we set the maximum epochs to 100 and we implemented early stopping if the categorical loss of validation does not increase after 8 epochs. We saved the best model of the weights that maximized the score on the validation set. We used PyTorch on Windows 11 for our implementation, with 4GB NVIDIA GeForce GTX 1650 GPU and 8GB DDR5 RAM. pre-trained model(ResNet50) was first considered 4 data augmentation techniques i.e image resizing, flip horizontal, rotating 10 degrees, and normalization was performed on the images. Figure 3 shows some examples of data augmentation.

VI. Results and Discussion

The performance of the models was evaluated using three different optimizers: Adam, SGD, and SGD with Nesterov. The results indicated that, in general, the Adam optimizer underperformed compared to the ResNet560 and VGG11 models, while the SGD optimizer showed better performance on the Self-built model. These findings are summarized in Table II. ResNet50 emerged as the top-performing model, exhibiting the highest macro-average F1-Score of 0.71 and an AUC of 0.88. Following closely is VGG11, which achieved an F1-Score of 0.68 and an AUC of 0.89. Based on these results, we selected the two best models in terms of macro-average performance, namely ResNet50 with the Adam optimizer and VGG11 with the Adam optimizer. It is worth noting that all three optimizers yielded satisfactory results when applied to the ResNet50 pre-trained model, with F1-Scores of 0.71 (Adam), 0.69 (SGD), and 0.70 (SGD with Nesterov), respectively.

A. Confusion Matrix

Both model's confusion matrix on the FER2013 testing set in Figure 4 shows the best classification on the "happy" emotions, ResNet50 predicts correctly 83% of the "happy" class while VGG11 correctly predicts 92% on the dataset. The VGG11 model makes it most mistakes when classifying "sad"

TABLE I
A COMPARISON OF THE VALIDATION ACCURACY OF THE STATE-OF-THE-ART METHODS ON THEIR BEST EPOCH.

Method	Pretrained	Best epoch	Optimizer	Best Validation Accuracy
ResNet50	✓	11	Adam	0.726
ResNet50	✓	17	SGD	0.706
ResNet50	✓	7	SGD Nestorev	0.705
VGG11	×	14	Adam	0.714
VGG11	×	35	SGD	0.665
VGG11	×	12	SGD Nestorev	0.688
SelfModel	×	11	Adam	0.657
SelfModel	×	91	SGD	0.678
SelfModel	×	24	SGD Nestorev	0.597

TABLE II
A COMPARISON OF THE OVERALL PERFORMANCE OF THE STATE-OF-THE-ART METHODS ON THEIR BEST EPOCH.

Method	Settings			Metrics									
				Per class F1-Score			micro-average		macro-average				
	Pretrained	Best epoch	Optimizer	Happy	Neutral	Sad	All	F1	AUC	Precision	Recall	F1	AUC
ResNet50	✓	11	Adam	**0.87**	0.63	0.62	**0.73**	**0.73**	**0.90**	**0.71**	**0.71**	**0.71**	0.88
ResNet50	✓	17	SGD	0.85	0.63	0.59	0.72	0.72	0.89	0.70	0.69	0.69	0.88
ResNet50	✓	7	SGD Nestorev	0.85	0.61	**0.63**	0.72	0.72	0.88	0.70	0.70	0.70	0.88
VGG11	×	14	Adam	0.85	**0.64**	0.56	0.72	0.71	0.89	**0.71**	0.69	0.68	**0.89**
VGG11	×	35	SGD	0.80	0.57	0.55	0.67	0.66	0.85	0.64	0.64	0.64	0.84
VGG11	×	12	SGD Nestorev	0.72	0.54	0.42	0.45	0.62	0.86	0.53	0.57	0.53	0.84
SelfModel	×	11	Adam	0.80	0.54	0.57	0.66	0.66	0.83	0.64	0.64	0.63	0.82
SelfModel	×	91	SGD	0.82	0.60	0.57	0.69	0.69	0.86	0.67	0.66	0.66	0.85
SelfModel	×	100	SGD Nestorev	0.73	0.53	0.44	0.61	0.59	0.79	0.58	0.57	0.57	0.78

emotions. The misclassification between ”Neutral” and ”Sad” may be due to the images having similarities in the dataset.

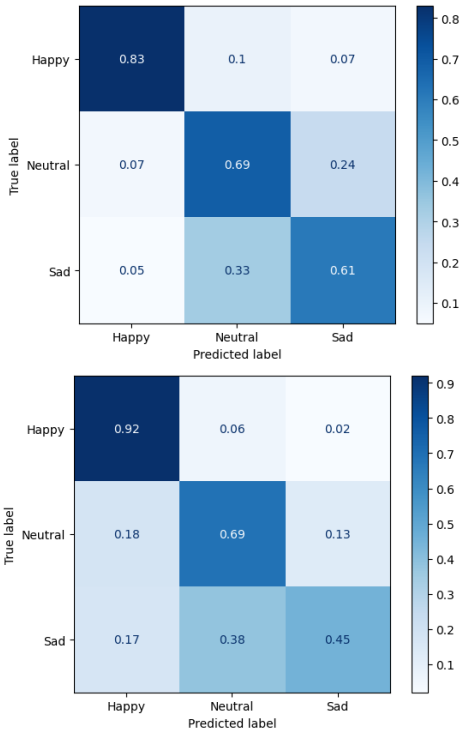


Fig. 4. Confusion Matrices for the top two best performers: ResNet50(top) best result and VGG11(bottom).

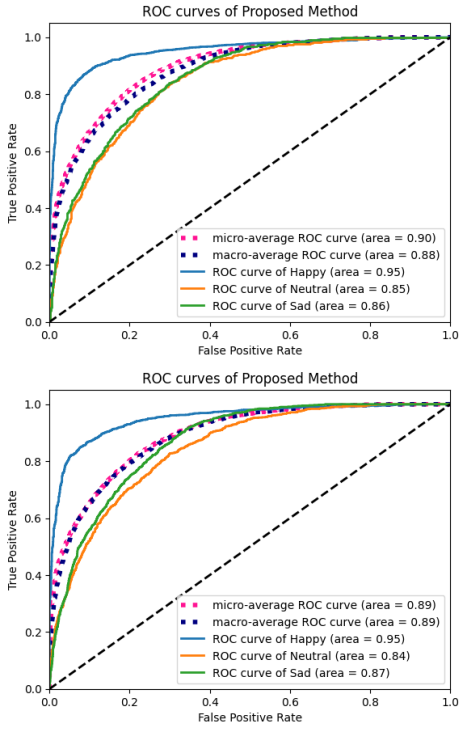
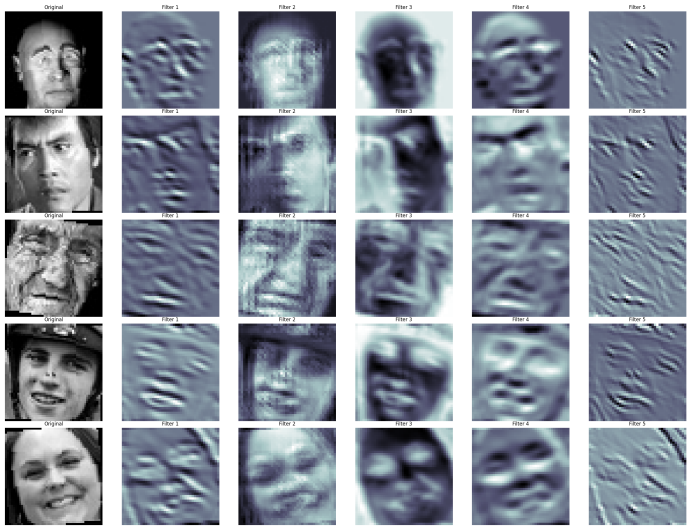


Fig. 5. ROC for the top two best performers: ResNet50 best(top) and VGG11(bottom).

B. Model Interpretability

The ”black box” nature of deep learning models, where it’s difficult to understand how the model processes input

images to make predictions. To address this issue and gain insights into the model's behavior, visualizing the information captured inside deep neural networks becomes crucial. Figure **??** provide a visual representation of the effects of different filters on input images. It helps in understanding how the filters modify the appearance of the images and can be useful for analyzing the behavior of convolutional filters in image processing tasks. This helps in understanding the visual features that the CNN captures from the input image and their importance in the final classification decision.



VII. Conclusion

This paper proposed a deep learning-based facial emotion detection method from images. we discussed two existing CNN models and a proposed model using just three classes from the FER2013 [Happy, Neutral, and Sad]. The performance evaluation of the proposed facial emotion detection is carried out in terms of validation accuracy, validation loss, optimizers, and learning rate. The models VGG11, ResNet50, and the proposed model were trained on the relevant dataset. After training the models, it is found that the ResNet50 and VGG11 with the Adam optimizer had a validation accuracy score of 72.6% and 71.4% respectively. The proposed model with SGD optimizer has a validation accuracy of 67.8%. The performance evaluation of the proposed facial emotion detection is carried out in terms of validation accuracy, validation loss, optimizers, and learning rate.

To further advance the research, we propose an investigation into various image processing techniques applied to the FER2013 dataset. This exploration aims to enhance the performance of existing models for facial emotion detection. By exploring different preprocessing methods and transformations, such as image augmentation and feature extraction, we anticipate improvements in accuracy and robustness. These image-processing techniques offer potential benefits and relevance within the domain of deep learning-based emotion recognition. Additionally, with the implementation of an ensemble of different transfer learning techniques, combining multiple models and leveraging their diverse capabilities, a more robust and accurate facial emotion detection system can be developed. Moreover, the investigation of alternative optimization algorithms and learning rate schedules may also yield improved results

References

[1] R. S. Lazarus, *Emotion and adaptation*. Oxford University Press, 1991.

[2] M. S. Bartlett, G. Littlewort, I. Fasel, and J. R. Movellan, "Real time face detection and facial expression recognition: Development and applications to human computer interaction." in *2003 Conference on computer vision and pattern recognition workshop*, vol. 5. IEEE, 2003, pp. 53–53.

[3] M. S. Juergen, "Automatic facial expression analysis: a survey," *Pattern recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[5] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for lvcsr using rectified linear units and dropout," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8609–8613.

[6] S. Ali, J. Li, Y. Pei, M. S. Aslam, Z. Shaukat, and M. Azeem, "An effective and improved cnn-elm classifier for handwritten digits recognition and classification," *Symmetry*, vol. 12, no. 10, p. 1742, 2020.

[7] R. Zhang, "Making convolutional networks shift-invariant again," in *International conference on machine learning*. PMLR, 2019, pp. 7324–7334.

[8] A. Jaiswal, A. K. Raju, and S. Deb, "Facial emotion detection using deep learning," in *2020 international conference for emerging technology (INCET)*. IEEE, 2020, pp. 1–5.

[9] Y. Khaireddin and Z. Chen, "Facial emotion recognition: State of the art performance on fer2013," *arXiv preprint arXiv:2105.03588*, 2021.

[10] K. Teoh, R. Ismail, S. Naziri, R. Hussin, M. Isa, and M. Basir, "Face recognition and identification using deep learning approach," in *Journal of Physics: Conference Series*, vol. 1755, no. 1. IOP Publishing, 2021, p. 012006.

[11] P. Bagane, S. Vishal, R. Raj, T. Ganorkar *et al.*, "Facial emotion detection using convolutional neural network," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 11, 2022.

[12] K. Donuk and D. Hanbay, "Sınıflandırma algoritmalarına dayalı vgg-11 ile yüzde duygu tanıma," *Computer Science*, no. Special, pp. 359–365.

[13] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, pp. 1–74, 2021.

[14] G. Forman and M. Scholz, "Apples-to-apples in cross-validation studies: pitfalls in classifier performance measurement," *Acm Sigkdd Explorations Newsletter*, vol. 12, no. 1, pp. 49–57, 2010.