Comparative Study of Deep Learning Models for Autism Diagnosis in Children Using Image Analysis

***Abstract*—Autism Spectrum Disorder (ASD) is a complex neurological disorder related to an individual's psychological difficulties which eventually impact their behaviour or reactions to the outside world. Identifying autism at a younger age offers several advantages, including the opportunity for the individual to lead a better life, enabling preparation for their future and that of their close family members and contributing to increased awareness and understanding of various medical conditions.In this paper we suggests a deep learning-based method that makes use of image datasets to identify ASD in children. The databases include facial patterns and additional visual clues that can be deduced from images. Deep learning models like VGG16, VGG19, EfficientNetB4 and MobileNet are used. These architectures are pre-trained on large-scale image datasets and refined to extract discriminative features on the ASD-specific dataset. We have acquired facial images datasets from a publicly available platform called Kaggle. Our primary goal is to compare the deep learning models which better fits the dataset and improve the accuracy of autism detection than any other work before. This paper aims to facilitate model comparisons and streamline the autism detection process using advanced deep-learning techniques available today.**

***Index Terms*—Autism Detection, ASD, Deep Learning, VGG16, VGG19. EfficientNet B4, MobileNet V1**

I. Introduction

Autism Spectrum Disorder (ASD) represents a lifelong condition that is associated with the development of the human brain. It's a rapidly growing category of disabilities characterized by repetitive behaviour patterns, specific interests and challenges in social interactions. Autistic individuals often encounter difficulty in expressing themselves, whether through verbal communication or non-verbal means such as gestures, facial expressions and physical touch. Individuals with autism may experience difficulties in the learning process and their skill development may occur unevenly, with some areas progressing at a different pace than others. This variability is a characteristic feature of autism. Now, individuals with autism can also exhibit remarkable abilities in various memory-intensive activities, excelling in areas such as mathematics, art, and more. These exceptional talents can be a unique aspect of their cognitive profile. The classification of autism as high-functioning or low functioning is contingent upon the individual's level of severity. Our paper aims to integrate machine-learning techniques with facial features data and utilize deep learning models which have been proven valuable in the early diagnosis of (ASD) to make more improvements in the autism detection sphere. We have collected facial feature images from both autistic patients and neurotypical individuals from the available online source Kaggle, conducted preprocessing and testing of the data, and then applied deep learning models like CNN which involves VGG16, VGG19, EfficientNet B4, and MobileNet V1 to analyze and extract valuable insights from the dataset. We will obtain our predictions from the deep learning model, which will enable us to identify and ascertain symptoms of autism based on the analysis of facial feature data. Early detection of Autism Spectrum Disorder (ASD) can benefit not only the individual with ASD but also their close family and support network, as it allows for better preparation and planning for the future. Advancements in modern technology and the application of deep learning models have indeed made early detection of Autism Spectrum Disorder (ASD) more accessible and streamlined.

In this paper, we have divided the proceedings into different sections. Section 2 describes the literature review, Section 3 is our proposed methodology having subsection of data preprocessing, model description and implementation, result analysis is in section 4, result comparison in section 5. Finally, section 6 is the future work part and section 7 for the conclusion and summarizing the full workings.

II. Literature Review

A. Early detection of autism by extracting features: A case study in Bangladesh:

In this paper [1], the differences between kids in Bangladesh who had been diagnosed with autism spectrum disorder (ASD) and those who didn't are examined. With the help of the Autism Barta App and fieldwork in Savar, researchers used a variety of approaches to figure out the characteristics of autism. They discovered that a number of variables including birth time and gender can also greatly affect the chance of developing autism. By examining more than 600 records, scientists were able to show how certain traits could be used to identify autism early on. The study also identified regional differences in autism frequency throughout Bangladesh providing insight into a variety of reasons. These discoveries offer insightful information to researchers and medical experts. On top of that, it also promotes better comprehension and treatment of autism spectrum illnesses.

B. Detecting autism from facial image:

In this paper [2], Convolutional Neural Network (CNN) classifier with transfer learning is used to identify autistic children. CNN is a core deep learning algorithm that extracts key characteristics from images for classification using pooling and convolution processes. Besides, transfer learning imitates how people apply previously acquired information to new

tasks by employing a pre-trained model for a secondary task. It also gives better accuracy. To categorize photos into different categories including autistic and non-autistic the study used a pre-trained VGG19 model, which is a 19-layer deep convolutional neural network trained on over a million images from the ImageNet database. Confusion matrices and classification reports were used to calculate metrics like specificity, sensitivity, and accuracy. The study acquired an accuracy of 84.67 percent on the validation data.

C. Facial features detection system to identify children with autism spectrum disorder: Deep learning models:

Another research [3], developed a web app using MobileNet which is a CNN architecture-based deep learning model. They used image data from online sources to detect facial features that resemble those of ASD patients. First, they fine-tuned the layers and optimized the models (using the RMSprop optimizer for output error reduction) according to their dataset. Three deep learning models, namely MobileNet, Xception, and InceptionV3 were applied to the dataset and the accuracies in detecting ASD patients were pretty high. Although the accuracy for MobileNet was the highest in their research, a more complex model might have been a better choice for building an ASD detection application, as it would reduce the chance of missing the incredibly intricate facial features.

D. Autism spectrum disorder detection using face features based on deep neural network:

In another research paper [4], we can see the implementation of pre-trained deep learning models like Xception and VGG16 (based on Transfer Learning) to diagnose autism. They used facial images of children for their dataset. After preprocessing the data and training and testing the models, they found that the Xception model performed the best, topping VGG16. It had better accuracy whereas VGG16 had 78% in successfully detecting ASD in children. Xception, as a DL model, has a high model complexity, which can lead to finding better results since intricate facial features cannot be easily missed by it. However, Xception, due to its complex design, can require more computational resources to train and infer, making it less suitable to run on applications with constrained hardware capabilities. There is also a risk of overfitting the data while training the model, due to the larger capacity and complex structure of Xception. This can lead to decreased accuracy for unseen data.

E. Deep learning approach for screening autism spectrum disorder in children with facial images and analysis of ethnoracial factors in model development and application:

In [5], the authors used VGG16 as their model for facial recognition of autism in children. Thus, they applied VGG16 for transfer learning as a pretrained model. In the case of training and testing, they divided the dataset into 80 percent and 20 percent respectively. Moreover, the dataset was composed of East Asian children's facial features. As a result, after training they got a good result.

F. Classification and detection of autism spectrum disorder based on deep learning algorithms:

In another work [6], VGG19, Xception, and NASNETMobile were enforced for classifying autism and non-autism in children.These three models were trained and tested for extracting traits from facial images of children for classification purposes. As a result, the Xception model had the highest accuracy among the three models which was 91%. On the other hand, VGG19 and NASNETMobile had an accuracy rate of 80% and 78% respectively. So, the accuracy rate of NASNETMobile and VGG19 was too low compared to the Xception model.

III. METHODOLOGY

A. Step by Step workflow

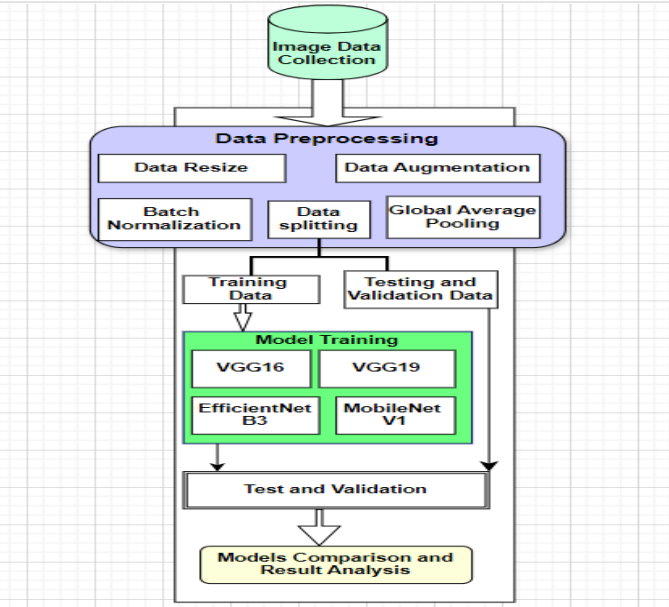

Fig. 1: Step by Step Workflow of the Research

B. Data Collection

The image dataset we used in this study is from kaggle and these photos were taken from Facebook and websites related to autism. The dataset has 2940 photos of autistic and non-autistic children. These photos are 2D and JPG-formatted images of different sizes, showing people between the ages of 2 and 14. The male and female ratio in this dataset is 3:1. The autism of children with autism and those without is the same. This dataset consists of 2940 photos. There are 2536 photos in the training set, 100 in the validation set, and 300 in the test set and it is divided evenly among the subfolders.

C. Data Preprocessing

Preprocessing the image dataset is very important as it increases the precision of image features to recognition and deep learning algorithms are highly dependent on image quality. The step-by-step preprocessing is as follows:

Resizing and Labeling the Image:

Resizing is the process of changing the image size to a predefined size. Our deep learning models need constant input size, therefore we have changed our model input size which ensures that all photos have the same dimensions and it is important for batch processing. Every input images is resized to 224*224 pixels. Every image is assigned to a label here. For multiclassification tasks, Label is returned as one hot-coded vector.

Data Augmentation:

Data augmentation is a technique used to improve the model's robustness and reduce overfitting by applying various transformations to existing images, thus artificially increasing the diversity of the training dataset. We instantiate an Image Data Generator. Lastly, apply it to the training dataset, configure the augmentation settings, generate batches of augmented data, and then proceed to train the model. It helps us to specify several augmentation parameters to work with, like rotation range, shifts in width and height, shearing, zooming, and horizontal flipping.

Batch Normalization:

In our code, we used batch normalization techniques to standardize the activations of each layer and speed up the training process. To guarantee numerical stability, the pixel values were standardized to fall between 0 and 1. Batch normalization layers were added to normalize activations after the dense layers. Then, dropout layers are added to avoid overfitting, and using activation functions ReLU, we introduce non-linearity. This process helps to maximize learning efficiency, accelerates training convergence, and enhances the model's generalization capacity.

Fig. 2: Labeled Images

Data Splitting :

After collecting data from the Kaggle website, a total of 2940 images were obtained. Among these, 2540 were chosen for training purposes and 300 images were chosen for testing purposes. The input data undergoes a sequence of steps, starting with the organization into a directory structure, followed by division into training and testing Datasets. So, For our consolidated dataset, we have used 0.8 percent for training,10 percent for testing, and .10 percent for validation.

Additional Layers:

Global average pooling is a technique used in conventional CNNs to substitute fully connected layers By creating a feature map for every classification job category in the final MLP-CONV layer. After using it, the feature maps' spatial dimensions are reduced to 1 x 1 and it preserved important spatial information and channelwise feature representations. In our Code, we added Global Average Pooling 2D then other layers were added to further process the features extracted by the pretrained CNN. A fully Connected Dense layer is also added which has 128 neurons. It also adds Relu Activation to understand complex features by introducing Nonlinearity. To learn strong features dropout layer was also added and the dropout rate was 0.5. We used softmax activation in output layers to generate a distribution of probabilities for all the classes in the dataset. It ensures that the sum of all probabilities is 1.

Early Stopping:

Early Stopping is a regularization method that is used to prevent the overfitting of the model. We used early stopping because the training was needed to stop when the validation accuracy did not improve after a predetermined number of epochs. We used early stopping here to better our model performance.

D. Model Description and Implementation:

1) VGG16: VGG16 is a widely used and popular Convolutional Neural Network (CNN) model, primarily used for tasks involving image recognition, classification, and segmentation. It consists of 13 convolutional layers and 3 fully connected layers. The convolutional layers are organized into 5 distinct blocks, where each block contains 2-3 convolutional layers followed by a max-pooling layer. Max-pooling, with a stride of 2 and a 2x2 filter, is necessary to reduce the spatial dimensions of the feature maps produced by the convolutional layers while preserving their most essential features. These convolutional layers use small 3x3 filters with a padding of 1 and a stride of 1. The number of filters starts at 64 in the first block and doubles with each subsequent block, reaching up to 512 in the final block. For the fully connected layers, the first two contain 4096 units each, meaning these layers are densely connected, allowing them to learn high-level features from the extracted features of the earlier layers. These two layers are followed by a final Softmax layer, which is used for classifying the categories or classes within the input image. VGG16 uses the Rectified Linear Unit (ReLU) activation function after every convolutional and fully connected layer, enabling the model to capture non-linear patterns in the data. In our code, we froze the learning of VGG16 to leverage its pre-trained feature extraction capabilities from ImageNet, a large dataset. Freezing the layers also reduces the number of parameters required for training, increasing the training speed and making the model less likely to overfit. Additionally, we implemented a custom model on top of the frozen VGG16 layers, which helped us adapt the network for binary classification.

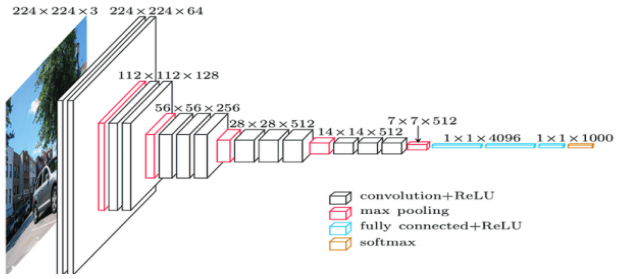

Fig. 3: VGG16 Architecture [7]

VGG16 is notable for its simple architecture and lower model complexity compared to other modern architectures. The use of 3x3 convolutional layers makes the model easier to implement and understand.

2) VGG19: VGG19 is well-known for its depth and efficiency in image classification tasks. The architecture consists of 19 layers, including 16 convolutional layers and 3 fully connected layers, enabling the network to capture high-level features from images. The convolutional layers use small 3x3 filters with a stride of 1 and padding to preserve the spatial resolution of the input image, allowing the network to capture fine details. These layers are followed by max-pooling layers with a 2x2 window and stride of 2, which reduce the spatial dimensions of the feature maps while retaining essential features. The network concludes with three fully connected layers: the first two contain 4096 neurons each, and the final layer has 1000 neurons for classification into 1000 classes. ReLU (Rectified Linear Unit) activations are applied after each convolutional and fully connected layer to introduce non-linearity. The final layer is a Softmax layer that produces a probability distribution over the 1000 classes. Pre-trained on the extensive ImageNet dataset, which contains over a million images across a thousand categories, VGG19 excels in transfer learning tasks, where its learned features can be fine-tuned for specific applications. VGG19 can be easily implemented using popular deep learning frameworks like TensorFlow and PyTorch. In TensorFlow, the Keras API provides a pre-trained VGG19 model that can be easily loaded and customized.

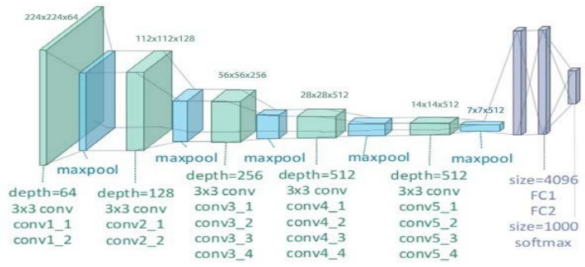

Fig. 4: VGG19 Architecture [8]

VGG-19 has been trained on the ImageNet dataset and is well-known for achieving high accuracy in image classification tasks. Its simple and uniform architecture, combined with the use of 3x3 convolutional layers, is highly effective. The hierarchical feature extraction of VGG-19 makes it particularly powerful. For transfer learning, VGG-19 is an excellent choice due to its ability to generalize well. Another key feature of the VGG-19 model is its versatility, as it can be easily adapted to a wide range of applications.

3) EfficientNet B4: EfficientNetB4 is part of Google's EfficientNet family and is a standout convolutional neural network (CNN) designed to optimize both accuracy and efficiency. Unlike traditional models that typically increase only one dimension, such as depth or width, EfficientNetB4 uses a compound scaling method to uniformly scale depth, width, and resolution. This approach starts with a highly optimized baseline network. EfficientNetB4 incorporates mobile inverted bottleneck MBConv layers, which break down computations into smaller, more manageable parts through depthwise separable convolutions. This helps the model run faster, use less memory, and maintain high performance. The model also uses the Swish activation function, improving accuracy. With a resolution of 380x380 pixels, EfficientNetB4 captures detailed image features, making it suitable for a wide range of image classification tasks. Pre-trained on the ImageNet dataset, it has strong feature extraction capabilities that can be fine-tuned for specific applications. EfficientNetB4 is flexible, powerful, and easy to implement using popular frameworks like TensorFlow and PyTorch.

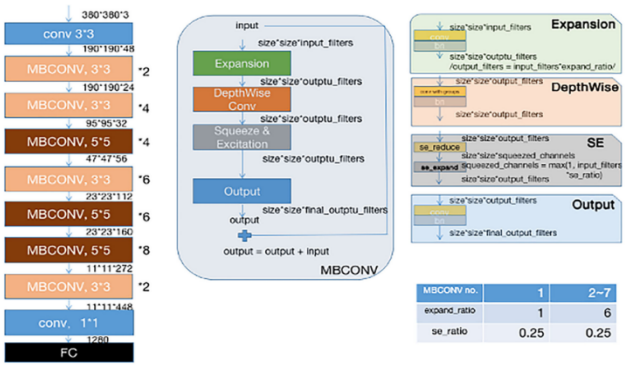

Fig. 5: EfficientNet B4 Architecture [9]

EfficientNetB4 achieves high performance by using a compound scaling method that balances the scaling of depth, width, and resolution. This approach results in fewer computational loads while maintaining efficiency. Pre-trained EfficientNetB4 models are available, offering outstanding results and simplifying the task of transfer learning. Additionally, the model is highly adaptive and easily scalable, delivering strong performance across various tasks.

4) MobileNet V1: MobileNet V1 is a convolutional neural network designed specifically for embedded and mobile vision applications. It is widely used in real-world tasks such as detection, fine-grained classification, localization, and facial feature recognition. MobileNet uses depthwise separable convolutions, where one filter is applied to each input channel.

It also includes a pointwise convolution layer, which uses a 1x1 convolution to create a linear combination of the output from the depthwise convolution. MobileNet is designed to be both complex and computationally lightweight, with two key parameters for adjusting efficiency: the width multiplier and the resolution multiplier. The width multiplier helps reduce the computational cost by adjusting the number of channels in each layer. The resolution multiplier reduces computational cost by adjusting the input image resolution. The architecture starts with a standard 3x3 convolutional layer, followed by depthwise separable convolutions. A pooling layer reduces the spatial dimensions of the feature maps, and the final layer is a fully connected layer that outputs class probabilities.

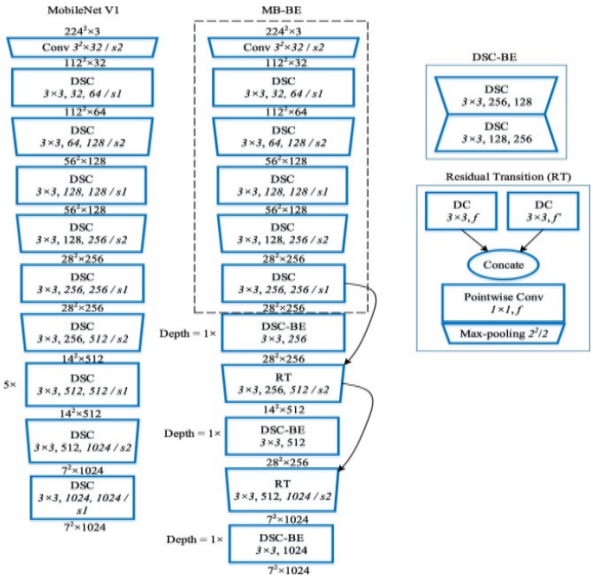

Fig. 6: MobileNet V1 Architecture [10]

IV. Result Analysis

The VGG16 model achieved an accuracy of 73%. Upon evaluation, its precision was 76.9%, recall was 71.5%, and the F1 score was 74.1%. These metrics provide insights into the model's performance: precision indicates the model's ability to correctly classify positive instances, recall measures its ability to capture all positive instances, and the F1 score represents the balance between precision and recall. We implemented the VGG19 model on our dataset, using pre-trained weights and adjusting its parameters for better learning. The model was trained in stages, called epochs. After each epoch, we evaluated its performance on a separate validation set, ensuring that it was improving without overfitting to the training data. EfficientNetB4 gave the highest accuracy among all the deep learning models we tested. We initially set the training to 60 epochs, but early stopping terminated the process at the 9th epoch due to a lack of significant improvement in validation performance. The model achieved an impressive accuracy of nearly 99% on the training set and a solid validation accuracy of around 88%. Its precision, recall, and F1 scores were 89.68%, 89.45%, and 89.48%, respectively. The AUC score was outstanding at 95%, reflecting the model's strong ability to distinguish between different classes. The MobileNet V1 model performed well but showed some fluctuations. It achieved an overall accuracy of 86%. The AUC score for both the autistic and non-autistic classes was 94%, demonstrating the model's strong ability to differentiate between the two classes.

The confusion matrix we got from VGG16, VGG19. EfficientNet B4 and MobileNet V1 are attached below:

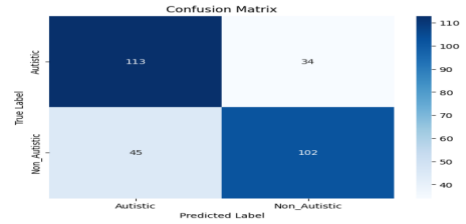

Fig. 7: VGG16

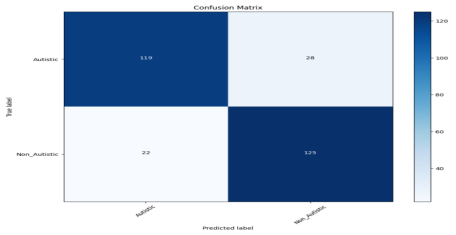

Fig. 8: VGG19

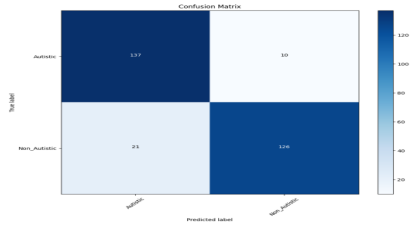

Fig. 9: EfficientNet B4

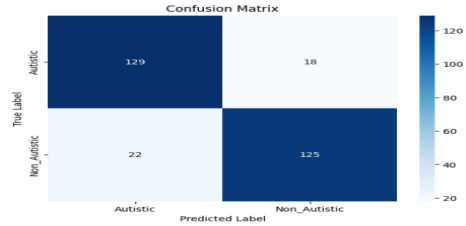

Fig. 10: MobileNetV1

Sample of predicted output we got from VGG19 and EfficientNet B4 are attached below and we also implemented the ROC curve and got the AUC score from that. EfficientNet B4 performed very well with our dataset and outperformed other models. The table and the model comparison is given below:

Fig. 11: Predicted Results with VGG19

Fig. 12: Predicted Results with EfficientNet B4

Models	AUC	Accuracy	Precision	Recall	F1 Score
VGG16	80%	73%	73%	73%	73%
VGG19	84%	84%	84%	84%	84%
EfficientNet B4	95%	90%	90%	89%	89%
MobileNet V1	94%	86%	86%	86%	86%

TABLE I: Deep Learning Models Result

V. Future Work

Our data collection process involved acquiring image datasets from Kaggle, predominantly comprising images of children from diverse Western countries. However, the dataset lacked a representation of Asian facial features. In the future, we want to collect image datasets from Asian childrens to increase diversity. Our future plans include gathering more data as primary sources to address this gap and enhance the accuracy of our image-based models. We would like to make a web interface that can detect autism through image or behaviour datasets. We can make that free and people can get highly benefited from this. Besides, we would like to explore more models most likely to implement AI models which have great demand nowadays.

VI. Conclusion

The aim of this research project is to identify autism in young children more accurately by using facial image processing. We tried to create a data-driven framework that will help in classifying the characteristics of the spectrum of autism in children effortlessly. We tried to ensure that there were fewer chances of misdiagnosis by proper training of deep learning algorithms. The confusion matrix assesses the performance of models that we are using in this research which helps in evaluating the negative and positive classes. Application of different deep learning models like VGG16, VGG19, MobileNet V1 and Efficient NetB4 helped us detect the characteristics of autism beneficially from the image datasets. Furthermore, leveraging autism features extracted from individual modalities of data will aid in enhancing the accuracy rate. Our result analysis revealed that EfficientNet B4 performs better than other models. With further data collection, training and testing we believe our research will be able to make a way towards the advancement of ASD research.

References

[1] M. S. Satu, F. F. Sathi, M. S. Arifen, M. H. Ali, and M. A. Moni, "Early detection of autism by extracting features: A case study in bangladesh," in 2019 international conference on robotics, electrical and signal processing techniques (ICREST), IEEE, 2019, pp. 400–405.

[2] S. Jahanara and S. Padmanabhan, "Detecting autism from facial image," International Journal of Advance Research, Ideas and Innovations in Technology, vol. 7, no. 2, pp. 219–225, 2021.

[3] Z. A. Ahmed, T. H. Aldhyani, M. E. Jadhav, et al., "Facial features detection system to identify children with autism spectrum disorder: Deep learning models," Computational and Mathematical Methods in Medicine, 2022.

[4] A. Rashid and S. Shaker, "Autism spectrum disorder detection using face features based on deep neural network," Wasit Journal of Computer and Mathematics Sciences, vol. 2, no. 1, 2023.

[5] A. Lu and M. Perkowski, "Deep learning approach for screening autism spectrum disorder in children with facial images and analysis of ethnoracial factors in model development and application," Brain Sciences, vol. 11, no. 11, p. 1446, 2021.

[6] F. W. Alsaade, M. S. Alzahrani, et al., "Classification and detection of autism spectrum disorder based on deep learning algorithms," Computational Intelligence and Neuroscience, 2022.

[7] T. Sugata and C. Yang, "Leaf app: Leaf recognition with deep convolutional neural networks," IOP Conference Series: Materials Science and Engineering, vol. 273, p. 012 004, Nov. 2017. doi: 10.1088/1757-899X/273/1/012004.

[8] Y. Zheng, C. Yang, and A. Merkulov, "Breast cancer screening using convolutional neural network and follow-up digital mammography," p. 4, May 2018. doi: 10.1117/12.2304564.

[9] C.-Y. Zhu, Y.-K. Wang, H.-P. Chen, et al., "A deep learning based framework for diagnosing multiple skin diseases in a clinical environment," Frontiers in Medicine, vol. 8, Apr. 2021. doi: 10.3389/fmed.2021.626369.

[10] E. Prasetyo, R. Purbaningtyas, R. D. Adityo, N. Suciati, and C. Fatichah, "Combining mobilenetv1 and depthwise separable convolution bottleneck with expansion for classifying the freshness of fish eyes," Information Processing in Agriculture, vol. 9, no. 4, pp. 485–496, 2022.