American International University-Bangladesh (AIUB)

Brain Tumor Detection and Categorization Using Deep Learning Techniques

Rameen Farhan (20-41981-1)
Raqibul Alam (20-42692-1)
MD Minhajul Hoque Chowdhury (20-43086-1)
MD Shahadat Hossen (20-43083-1)

A Thesis submitted for the degree of Bachelor of Science (BSc) in Computer Science and Engineering (CSE) at American International University Bangladesh in February, 2024

Faculty of Science and Technology (FST)

Abstract

Brain tumor classification and detection plays a pivotal role in medical diagnosis and treatment planning. This study introduces a comprehensive method to enhance the accuracy of brain tumor identification and classification using a diverse dataset comprising three different types of brain tumors, alongside a class representing the absence of a tumor. Our approach integrates various image preprocessing techniques, diverse classifiers, and convolutional neural network (CNN) models to achieve robust classification performance. Initially, Gaussian, Unsharp Mask, and Laplacian filters are applied to improve image quality and emphasize significant features. Subsequently, feature extraction is conducted using three pre-trained CNN models (ResNet50, VGG19, and MobileNetV2) alongside a custom-designed CNN architecture, adept at capturing hierarchical image representations. Feature vectors of size 1x1024 are obtained from these models and then utilized as inputs for three classical classifiers: Random Forest, AdaBoost, and k-Nearest Neighbors (KNN). By amalgamating preprocessing, deep learning, and classical machine learning techniques, our aim is to exploit the complementary strengths of each approach for enhanced classification accuracy. Experimental results on the multi-class brain tumor dataset demonstrate promising outcomes, achieving a classification accuracy of 97.1%. This notable accuracy highlights the efficacy of our proposed methodology in accurately identifying and categorizing brain tumor types, thereby facilitating more informed medical decisions and treatments. Our study contributes to ongoing efforts in leveraging advanced computational techniques for improving medical image analysis and diagnosis.

Approval

The thesis titled **"Brain tumor detection and categorization using deep learning techniques"** has been submitted to the following respected members ofthe board of examiners of the department of computer science in partial fulfilment of the requirementsfor the degree of Bachelor of Science in Computer Science on **(23th February)** and has been acceptedas satisfactory.

DR. S M Hasan Mahmud
Assistant Professor, Computer Science
Faculty of Science and Technology
American International University-Bangladesh

Sazzad Hossain
Assistant Professor, Computer Science
Faculty of Science and Technology
American International University-Bangladesh

DR. Akinul Islam Jony
Associate Professor & Head (UG)
Faculty of Science and Technology
American International University-Bangladesh

Prof. Dr. Dip Nandi
Associate Dean
Faculty of Science and Technology
American International University-Bangladesh

Mashiour Rahman
Sr. Associate Professor & Dean-in-charge
Faculty of Science and Technology
American International University-Bangladesh

Contributions by authors to the thesis

List the significant and substantial inputs made by different authors to this research, work and writing represented and/or reported in the thesis. These could include significant contributions to: the conception and design of the project; non-routine technical work; analysis and interpretation of researchdata; drafting significant parts of the work or critically revising it so as to contribute to the interpretation.

	Rameen Farhan	**Raqibul Alam**	**MD Minhajul Haque Chowdhuary**	**MD Shahadat Hossen**	**Contribution (%)**
	20-41981-1	*20-42692-1*	*20-43086-1*	*20-43083-1*	
Conceptualization					100 %
Data curation					100 %
Formal analysis					100 %
Investigation					100 %
Methodology					100 %
Implementation					100 %
Validation					100 %
Theoretical derivations					100 %
Preparation of figures					100 %
Writing – original draft					100 %
Writing – review & editing					100 %

If your task breakdown requires further clarification, do so here. Do not exceed a single page.

Acknowledgments

First of all, I would like to thank almighty Allah, for his grace in accomplishing our thesis timely and our families for giving us mental and financial support throughout the whole bachelors.

We would like to express our gratitude to the Faculty of Science & Technology to keep thesis credit in the curriculum of the graduation program and giving us a scope of tasting the flavor or research work with our interest.

We also want to express our gratitude to our supervisor, Dr. SM Hasan Mahmud sir, who gave us motivation, guidance, and control for this venture. He has decent knowledge of programming advancement, and his personal circumstances, perpetual assistance made the task do able.

Keywords

CNN, Pre-trained, Feature Extraction, Brain Tumor, MRI, CT, X-Ray, Random Forest, KNN, Filte.

Table of Contents

List of Figures

List of Tables

List of Abbreviations and Symbols

Mention all the abbreviations and the different symbols that is used in this document.

Abbreviations	
ML	Machine Learning
CNN	Convolutional Neural Network
KNN	K-nearest Neighbor
SVM	Support Vector Machine
SVC	Support Vector Classifier
ResNet	Residual Network
RF	Random Forest
FE	Feature Extraction
RN	Residual Network
etc.	*etc.*

Symbols	
σ	variance and hyperparameter
etc.	*etc.*

Chapter 1

Introduction

1.1 Thesis topic

Brain Tumor Detection and Categorization Using Deep Learning Techniques

1.2 Introduction

Brain tumor classification and detection play a crucial role in medical imaging for precise diagnosis and treatment planning. This thesis presents a comprehensive methodology for the classification of brain tumors utilizing a diverse dataset comprising three distinct tumor types along with a tumor-free class. By utilizing advanced image preprocessing filters and cutting-edge deep learning models like ResNet50, VGG19 and MobileNetV2, alongside classical machine learning classifiers such as AdaBoost, Random Forest, and KNN, our approach aims to achieve high accuracy in distinguishing between different tumor types. This research contributes to the advancement of medical image analysis, with significant implications for improving patient care and treatment outcomes.

1.3 Medical Images

Medical imaging plays a pivotal role in modern healthcare, facilitating non-invasive visualization of internal structures and aiding in early detection, diagnosis, and treatment monitoring of various medical conditions. Among the array of modalities utilized in medical imaging, including X-ray, CT, MRI and ultrasound, the analysis of medical images holds significant promise for advancing diagnostic accuracy and patient care. Particularly in neuroimaging, where the complexities of brain structures and abnormalities require sophisticated analysis techniques, our research focuses on medical image analysis, specifically targeting brain tumor classification—a critical task with profound implications for patient management and treatment decision-making. Utilizing a diverse dataset sourced from Kaggle, which includes various types of brain tumors alongside tumor-free samples, our aim is to leverage computational methods and advanced machine learning(ml)

algorithms to enhance the effectiveness and efficiency of medical image analysis. Ultimately, our objective is to contribute to the improvement of healthcare outcomes for individuals affected by brain tumors.

1.4 Motivation

The impetus behind this research originates from the urgent necessity for enhanced methodologies in medical imaging analysis, particularly within the realm of brain tumor classification. Diagnosing and planning treatment for brain tumors present intricate challenges that demand precise identification and characterization for effective patient care. Existing approaches to brain tumor classification are hindered by accuracy and efficiency limitations, underscoring the need for innovative methodologies to overcome these deficiencies.

Moreover, the presence of extensive medical imaging datasets, such as those accessible via platforms like Kaggle, provides an opportunity to utilize advanced computational techniques and machine learning algorithms for improved medical image analysis. By capitalizing on these datasets alongside state-of-the-art methodologies, our objective is to construct a robust framework for brain tumor classification that can significantly impact clinical practice.

The potential ramifications of this research are considerable, as precise and efficient brain tumor classification can facilitate earlier detection, more accurate treatment planning, and ultimately enhance patient outcomes. By contributing to the progression of medical imaging analysis techniques, this research holds promise for making a tangible difference in the lives of individuals grappling with brain tumors, ultimately enhancing their quality of life and prognosis.

1.5 Objective

The primary aim of this research is to formulate a comprehensive methodology for brain tumor classification employing advanced computational techniques and ML algorithms. The specific objectives are outlined as follows:

1. To investigate and assess various image preprocessing filters, such as Unsharp Mask, Gaussian and Laplacian filters, to enhance the quality and emphasize pertinent features in

brain tumor images.

2. To examine the efficacy of diverse pre-trained convolutional neural network (CNN) architectures, encompassing ResNet50, VGG19 and MobileNetV2, for extracting distinctive features from brain tumor images.
3. To propose a feature extraction approach based on pre-trained CNN models, followed by classification utilizing classical machine learning algorithms like AdaBoost, Random Forest and k-Nearest Neighbors (KNN).
4. To gauge the performance of the proposed methodology on a multi-class brain tumor dataset by assessing classification accuracy and comparing it with existing methodologies.
5. To evaluate the clinical significance and potential impact of the developed methodology in enhancing medical image analysis and facilitating more precise diagnosis and treatment planning for individuals afflicted with brain tumors.

1.6 Orientation

The essay's second chapter covers the foundations of ML, image processing, and brain tumor identification in addition to previous academics' study on the same topic. The implementation, data preprocessing, augmentation, CNN model train, classification, and a quick summary of the methods used with this model are all covered in the third chapter. The next section contains a representation of the train test portion and the tools and libraries used to build this model. Chapter 4 summarizes the specifics by outlining all of the challenges and contrasting each model's output to determine which accuracy is optimal. We covered the paper and our model's shortcomings in chapter 5. Finally, a few closing thoughts and potential directions for further research are covered in chapter 6.

Chapter 2

Literature review

Due to the intricate nature of human brains, the detection of brain tumors stands out as one of the most formidable and demanding challenges in medical science. Through the utilization of deep learning and image processing methodologies, medical experts and researchers continuously endeavor to augment the precision of brain tumor identification. The application of Convolutional Neural Networks (CNN) and its models in brain tumor detection is increasingly expanding, primarily due to their high detection accuracy. Researchers are continuously introducing new techniques and algorithms aimed at enhancing the accuracy of brain tumor identification.

In recent years, deep learning methods like CNN have shown significant potential in diagnosing brain cancers. By extracting complex patterns and features from MRI data, CNN models can accurately identify the locations of brain tumors. For instance, Dunsheng Liu et al. proposed a novel model named Global Average Pooling Residual Network (G-ResNet) for automatic brain tumor identification using CNN classification. Achieving a classification accuracy of 95.00%, this model surpasses earlier models by a significant margin. [6]

Ouiza Nait Belaid et al. utilized grey level co-occurrence matrix (GLCM) features images and original photographs as inputs to propose a deep learning method relying on pre-trained VGG-16 CNNs for the classification of three types of brain cancers. The experimental findings indicate that original images exhibit more prominent and distinguishable features when energy is employed as input, achieving an average accuracy of 96.5% compared to alternative input combinations. [7]

To differentiate between images of normal brain tissue and brain tumors, Ramya Mohan and colleagues introduced the MIDNet18 CNN architecture as an alternative to the VGG16 CNN architecture. The MIDNet18 model achieved an accuracy of 98.7%, while the VGG16 model attained 50% accuracy, indicating a significant difference in performance. [8]

Moreover, several CNN models, including ResNet50, Mobile Net V2, and VGG16, have been employed by researchers to identify brain cancers. An algorithm presented by Zahid Rasheed and his colleagues, for example, was tested against the pre-trained VGG16, VGG19, ResNet50, MobileNet V2, and

Inception V3 algorithms using benchmarked data. The results of the trial showed a noteworthy 98.04% categorization accuracy. [9]

Moreover, deep learning models' computational complexity has decreased because to feature extraction, allowing for quicker processing and improved accuracy. For example, Arpit Kumar and colleagues introduced a novel method using an ensemble classifier based on machine learning that combines augmentation and feature extraction techniques. To precisely identify malignancies, the suggested approach makes use of an optimized fusion vector. This hybrid technique performed exceptionally well, utilizing a modified Resnett50 model and HOG (Histogram of Oriented Gradients) to achieve an 88% detection accuracy. The superiority of the suggested strategy was demonstrated by a comparison of the outcomes with the current methods. [10]

Furthermore, Researchers have explored a range of classifiers for extracting features and classifying them into tumor and non-tumor regions. For instance, Tonmoy Hossain et al. employed the Sci-kit-learn implementation to utilize six standard classifiers: Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Multilayer Perceptron (MLP), Logistic Regression, Naive Bayes, and Random Forest. Subsequently, they experimented with CNN and achieved an impressive accuracy of 97.87%, showcasing its exceptional capabilities. [11]

A method introduced by Aryan Sagar Methil involved conducting an experimental study on a dataset comprising tumors with diverse characteristics such as sizes, shapes, textures, and locations. The classification task was accomplished using Convolutional Neural Network (CNN) technology. The CNN model exhibited highly impressive performance, achieving a recall rate of 98.55% on the training set and an astounding 99.73% on the validation set. [12]

A technique proposed by A. Ari et al. consists of three stages: preprocessing, extracting tumor regions based on image processing, and tumor classification using the Extreme Learning Machine Local Receptive Fields (ELM-LRF). The experimental studies achieved a 97.18% accuracy in classifying cranial MR images. Evaluation results indicated that the suggested method's effectiveness surpassed that of other recent studies in the literature. [13]

D. Fabbri et al. studied 989 axial photos from 191 patients in order to avoid the confusion of three different planes with the same diagnostic in neural networks. Neural networks, both convolutional and totally linked, were used for classification. Within these two groups, more tests were calculated by

enhancing the initial 512 512 axial photos. An average five-fold cross-validation of 91.43% for the most highly trained neural network shows the classification accuracy of neural networks trained on axial data. [14]

A. Rehman introduced a novel deep learning-based approach for the identification and categorization of small brain cancers. Initially, brain tumors are extracted using a 3D CNN architecture, followed by feeding the tumors into a CNN model pretrained for feature extraction. For testing and validation purposes, three BraTS datasets from 2015, 2017, and 2018 are utilized, yielding corresponding accuracy rates of 98.32%, 96.97%, and 92.67%. Comparative analysis against existing techniques demonstrates that the proposed design achieves a comparable level of accuracy. [15]

Liya Zhao et al. devised a three-stream architecture called multiscale CNNs, aimed at automatically selecting the optimal image sizes and integrating data from different scales of the surrounding areas. They utilized the MICCAI 2013 Multimodal Brain Tumor Image Segmentation Benchmark (BRATS) datasets for both training and assessment. The developed multiscale CNNs framework not only incorporates multimodal features from T1, T1-enhanced, T2, and FLAIR MRI images but also integrates multimodal features from T1, T1-enhanced, T2, and FLAIR images. Their approach enhances the robustness and accuracy of brain tumor segmentation compared to the top two BRATS 2012 and 2013 algorithms and conventional CNNs. [16]

A brain tumor categorization framework was introduced by A. Deshpande et al., utilizing CNN algorithms and artificial intelligence. The effectiveness of the framework was evaluated with and without the utilization of super-resolution techniques. A notable accuracy of 98.14% was achieved by integrating the ResNet50 architecture with super-resolution. The proposed super-resolution framework, incorporating CNN, ResNet50, and the discrete cosine transform (DCT), significantly enhances the accuracy of tumor classification, as demonstrated by experimental results using MRI images. [17]

ResNet-50, CNN, and DNN are three different neural networks that Imran Javaid et al. presented. Ultimately, every deconstructed neural network is assigned to a separate dataset. OTSU segmentation is used to isolate a tumor when an image has been accurately confirmed to be one. The experimental findings show that the ResNet-50 algorithm has a minimal test loss of 0.0269. It also has the greatest F1 score of 1.0, precision of 1.00, and high classification accuracy of 0.996. Numerous tests show that the suggested segmentation for tumor identification is accurate and efficient. [18]

H. Khan and colleagues propose a novel method that integrates data augmentation, image processing, and a convolutional neural network (CNN) to classify brain MRI scan images as malignant or noncancerous. Employing transfer learning, they compare their CNN model with pre-trained VGG-16, ResNet-50, and Inception-v3 models. Despite having a limited dataset, their model surpasses VGG-16 (96%), ResNet-50 (89%), and Inception-v3 (75%) with an impressive accuracy of 100%. Moreover, this model exhibits greater accuracy and efficiency compared to currently used pre-trained models due to its minimal complexity and processing requirements. [19]

Ahmet Çinar developed a technique for diagnosing brain tumors utilizing CNN models and MRI data. They employed the ResNet50 architecture as the foundational model, augmenting it with eight additional layers while removing the final five layers. This customized model achieved an impressive accuracy of 97.2%. Additionally, they assessed the effectiveness of several other models, including GoogLeNet, InceptionV3, AlexNet, ResNet50, DenseNet201, and InceptionV3. The top-performing model accurately classified images of brain tumors. Prior research in the literature has validated the efficacy and potential of this method for application in computer-aided systems for brain tumor detection. [20]

Zhiguan Huang et al. introduced a novel method for classifying brain cancers in magnetic resonance imaging (MRI) known as CNNBCN (Convolutional Neural Network based on Complex Networks). Unlike manually constructed networks, the CNNBCN network structure is generated using randomly generated graph methods and then transformed into a computable neural network via a network generator. The updated CNNBCN model surpasses other models reported in related studies, achieving an impressive classification accuracy of 95.49% for brain tumors. In experimental trials, the model also exhibits lower test loss compared to ResNet, DenseNet, and MobileNet models. Besides yielding excellent results in brain tumor classification, the updated CNNBCN method advances neural network design methodologies. [21]

Yakub Bhanothu and colleagues propose a deep learning system named Faster R-CNN to streamline the laborious and error-prone manual evaluation process of MRI images for tumor diagnosis. Both the region proposal network and the classifier network in the algorithm are built upon the VGG-16 architecture. The Region Proposal Network (RPN) effectively identifies tumors and the regions in which they manifest. The algorithm achieves an average precision rate of 75.18% for gliomas, 89.45% for meningiomas, and 68.18% for pituitary tumors. Its efficacy as a performance metric in tumor detection and classification is demonstrated by a mean average precision of 77.60% across all tumor classes. [22]

In their research, A. P. Rahmathunneesa and colleagues evaluated the performance of four pretrained deep learning networks in classifying brain cancers into four classes. The selected networks—AlexNet, ResNet-50, and GoogLeNet Inception V3—have previously undergone pre-training on the imageNet dataset. The study utilized MRI images of glioma brain tumors and applied preprocessing techniques such as data augmentation and skull stripping. The network designs were compared based on various parameters including accuracy, precision, recall, F1-score, and training/validation time. According to the experimental findings, AlexNet achieved an accuracy of 92.98% in a relatively short duration of 19 minutes, whereas ResNet50 achieved a higher accuracy of 96.05% but required a longer training time of approximately 100 minutes. [23]

Tahia Tazin et al. introduced a unique method that focuses on utilizing X-ray images to diagnose brain tumors and improve patient treatment planning. The method explores the use of convolutional neural networks (CNNs) for brain cancer detection through transfer learning. Pre-trained CNN models such as VGG19, InceptionV3, and MobileNetV2 were employed for deep feature extraction. Performance was evaluated based on classification accuracy, with MobileNetV2 achieving 92% accuracy, InceptionV3 achieving 91%, and VGG19 achieving 88%. The most accurate model, MobileNetV2, enables early tumor diagnosis before the onset of physical limitations such as paralysis. [24]

Divjot Kaur et al. introduce a technique that underscores the importance of early identification and automated approaches for analyzing brain tumors. The study presents three feature extraction models—VGG16, VGG19, and Inception v3—which offer efficient methods for generating precise predictions. Machine learning classifiers are utilized to classify brain tumors as either benign or malignant. The results demonstrate that the highest accuracy of 99.4% is achieved through the combination of VGG16 and a neural network classifier. This study emphasizes the significance of automated techniques in enhancing the precision and effectiveness of brain tumor diagnosis. [25]

Based on the literature review, various methodologies and approaches, including feature extraction, deep learning-based methods, and classifiers employing machine learning and image processing techniques, have been proposed for diagnosing brain tumors. However, there remains potential to enhance the accuracy and effectiveness of these methods. To achieve more comprehensive and precise detection of brain cancers, this study proposes the utilization of multiple filters, CNN models, feature extraction techniques, and classifiers.

Chapter 3

Methods

Image capture, putting the image into the dataset, data augmentation, gaussian smoothing, image normalization, dimensionality reduction, one hot encoding, and classification are some of the processes in the suggested methodology. After the photos are first taken, they are pre-processed utilizing techniques like augmentation, smoothing, normalization, etc. 2072 photos of brain tumors can be accessed online.

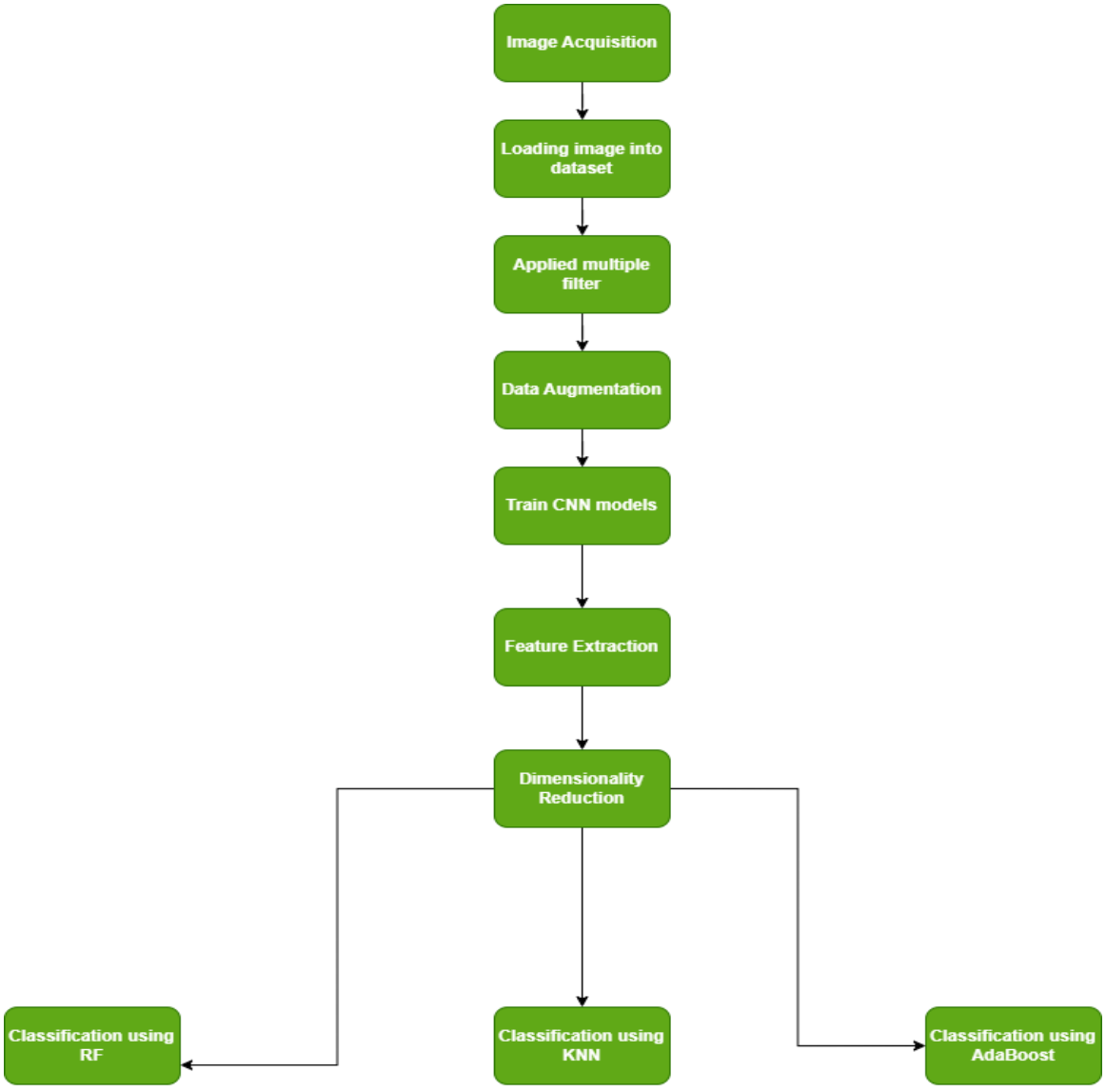

Figure 3.1: Flow chart of proposed methodology

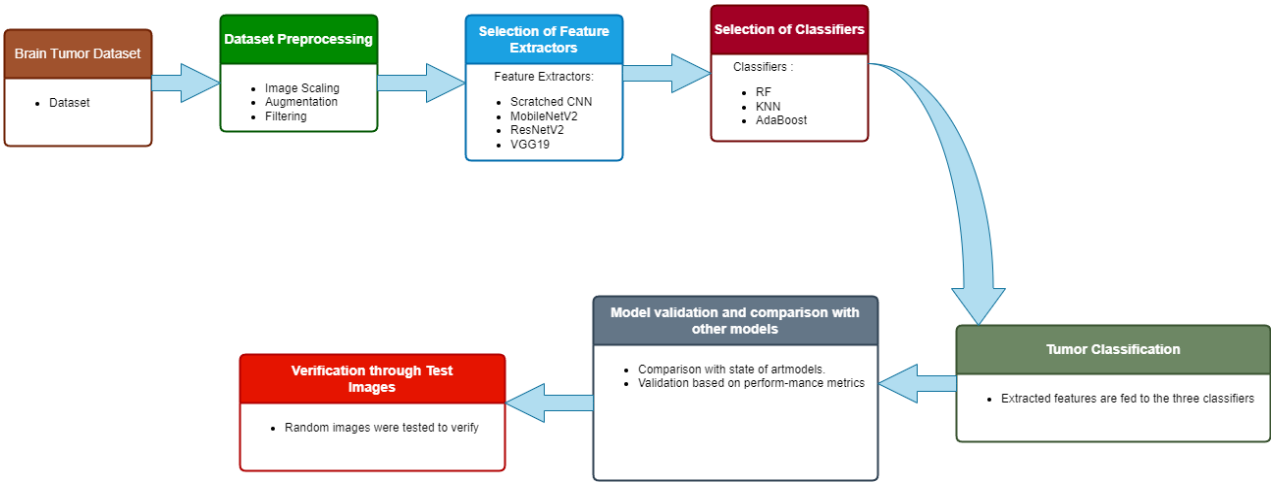

Figure 3.2: Diagram of proposed methodology

3.1 Dataset

A multi-class classification dataset in machine learning assigns each data point to one of several classes, aiming to categorize data points into more than two groups or classes. In this study, a multi-class classification dataset comprising four classes is utilized. These classes include gliomas, meningiomas, pituitary tumors, and instances with no tumors. Each of the Glioma, Meningioma, and Pituitary datasets consists of 1,168 MRI images, while the No tumor dataset contains 1,175 images. Gaussian, Unsharp Mask, and Laplacian filters will be applied to these datasets. Throughout the study, these four types of datasets will be referred to as Dataset 1,2,3,4 respectively.

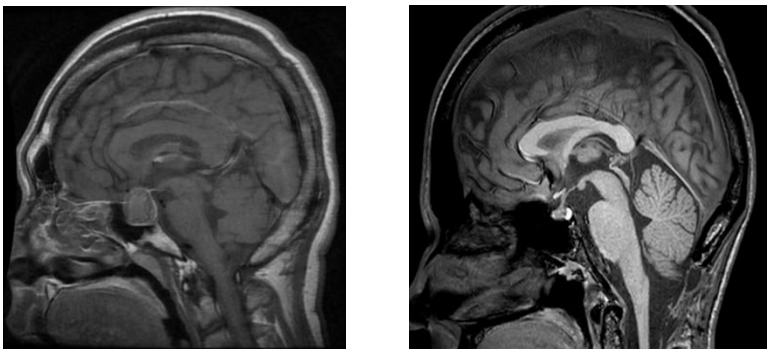

Figure 3.3: Dataset classes

3.2 Data Preprocessing

The initial stage in data analysis is data preprocessing, involving the cleaning, conversion, and organization of raw data to prepare it for further analysis. This process involves addressing missing values, outliers, and ensuring data integrity. Data preprocessing may also encompass tasks such as scaling, normalization, feature engineering, and dimensionality reduction to optimize the data for modeling purposes. Effective data preprocessing enhances the efficiency and accuracy of machine learning algorithms and statistical analyses.

3.2.1 Filtering

3.2.1.1 Unsharp Mask

This dataset comprises MRI images processed with the unsharp mask filter. Unsharp mask filtering is a commonly employed technique in image processing aimed at enhancing image sharpness by emphasizing edges and fine details. This process entails subtracting a blurred version of the original image from the original image itself.

Here's how it works:

1. Blurring the Image

Initially, the original image undergoes blurring through the application of a Gaussian filter or another smoothing filter. This blurred image corresponds to the low-frequency components of the original image.

2. Subtraction

Subsequently, the blurred image is subtracted from the original image. This procedure amplifies high-frequency components, such as edges and fine details, by highlighting the disparities between adjacent pixel values.

3. Combining with Original Image

The outcome of the subtraction is then merged with the original image to generate the final sharpened image.

Mathematically, the unsharp mask filtering process can be represented as follows:

Let I be the original image.

Let g be the blurred image obtained by applying a Gaussian filter.

Let S be the sharpened image obtained by subtracting the blurred image from the original image.

The formula for unsharp mask filtering is:

S = I-(K*g)

where k is a scaling factor that controls the strength of the sharpening effect. Typically k is chosen to be between 0.5 and 1.5. Adjusting the value of k allows for fine-tuning the degree of sharpening applied to the image.

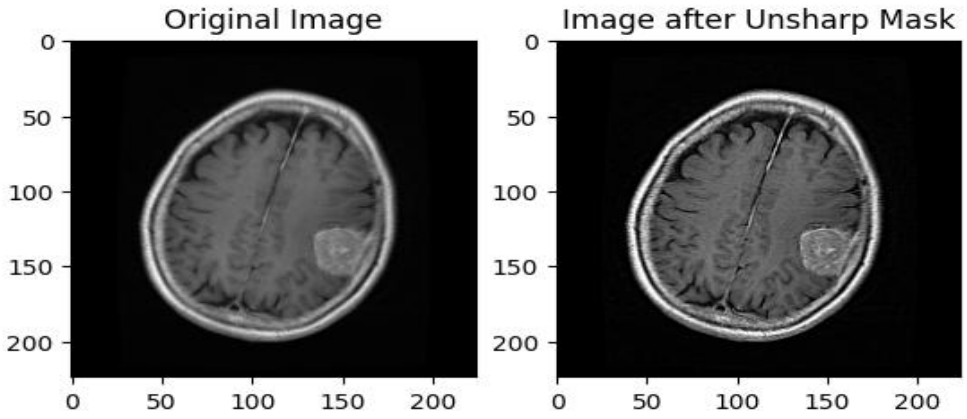

Figure 3.4: unsharp mask filter image

3.2.1.2 Gaussian

This dataset contains MRI images treated using the unsharp mask. Gaussian filtering is an image processing technique that uses a Gaussian function to blur or smooth a picture. This function calculates a weighted average of neighboring pixel values, with higher weights allocated to central pixels and decreasing weights to surrounding pixels. The outcome is a smoother image with less high-frequency noise while preserving image borders and details. Gaussian filtering is widely used in computer vision, medical imaging, and photography to reduce noise, denoise images, and perform pre-processing tasks.

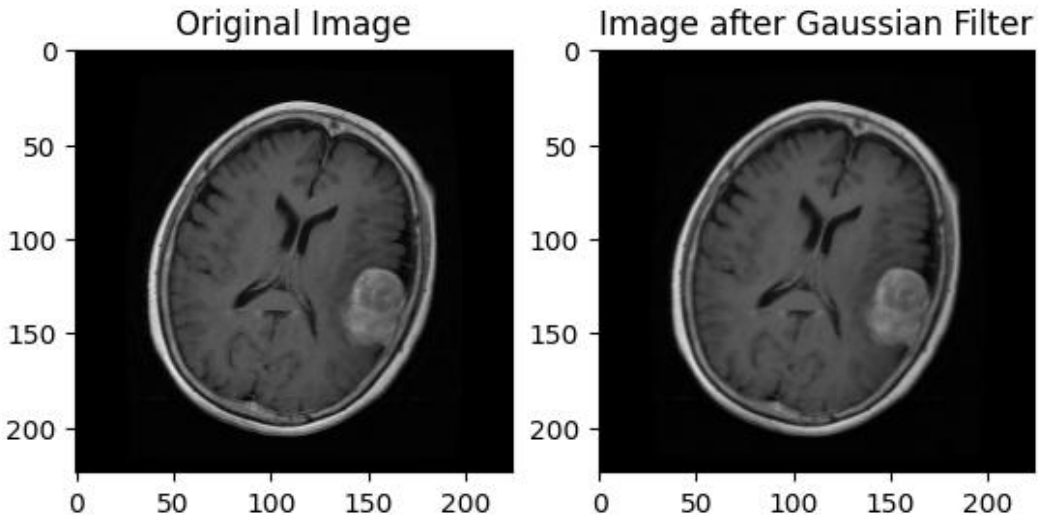

Fig 3.5: Gaussian filter image

3.2.1.3 Laplacian

The Laplacian filter is an image processing technique that utilizes the second derivative of the image to amplify edges and identify features. It highlights regions with the steepest intensity gradient, such as edges, corners, and boundaries. This filter is constructed by convolving the image with a Laplacian kernel, typically a 3x3 or 5x5 matrix. By doing so, it accentuates the high-frequency components of the image while diminishing low-frequency information. The Laplacian filter finds widespread application in edge detection, image sharpening, and feature extraction tasks across various domains, as computer vision, medical imaging, and digital image processing. However, it can also amplify noise, necessitating careful adjustment of parameters or post-processing techniques to mitigate its effects.

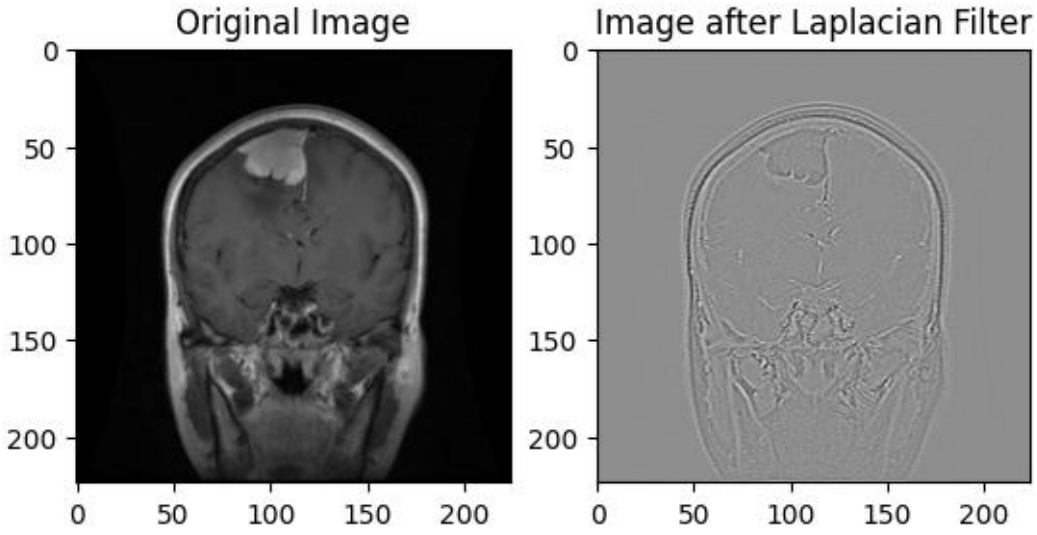

Figure 3.6: Laplacian filter image

3.2.2 Data Augmentation

The datasets undergo minimal preprocessing, which includes image scaling and augmentation. All images are resized to dimensions of 224x224 pixels. Initially, the dataset serves as the input image data. Subsequently, the rescale parameter is applied to normalize the pixel values of the images by dividing them by 255. Following this step, the dataset is divided into validation and training sets, with the validation_split parameter set to 20%.

Validation Split = Validation size / Total size

Furthermore, a zoom range of 0.99 was configured, allowing the images to randomly zoom in and out. Subsequently, the image data is passed to the preprocessing train function, which takes a directory path containing the training photos as input and yields an image generator.

zoom_range = (min_zoom , max_zoom)

Afterward, the image data object is employed to generate batches of training photos, with each batch containing eight images. The target size for each image is defined as 224x224 pixels. In order to maintain reproducibility, the seed value is set to 123. Lastly, the subset parameter is specified as "training", indicating that this constitutes the training set.

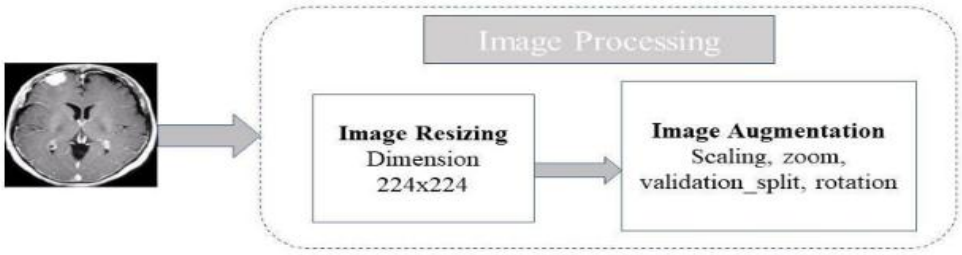

Figure 3.7: Data Processing Stages

3.3 Feature Extraction

Feature extraction refers to the method of reducing the dimensionality of data while retaining pertinent information. It involves transforming raw input data into a feature space, where each dimension represents a significant component or attribute of the data. This approach finds extensive application in machine learning and pattern recognition tasks, aiming to enhance computational efficiency and model effectiveness.

Three pre-trained models ResNet50, VGG16, MobileNetV2, and a Scratch CNN model, were explored for feature extraction. All the aforementioned pre-trained models were applied to the dataset. Feature extraction was performed on the trained data, extracting 1x1024 features from each image.

3.3.1 Modified Scratched CNN

Convolutional Neural Networks (CNNs) are a specific type of neural network designed to process data organized in a grid-like structure, such as images or audio spectrograms. CNNs effectively learn and extract features from input images through a combination of convolutional filters, activation functions, pooling operations, and fully connected layers. In this research, the CNN architecture was fine-tuned and refined through adjustments aimed at enhancing its performance. This adapted CNN architecture comprises multiple layers, including convolutional layers, maximum pooling layers, a flatten layer, and dense layers.

$$Z_i = b_i + \sum_{j=1}^{F} \sum_{k=1}^{F} W_{j,k} \underset{X_{i+j,i+k-1}}{} \quad (1)$$

The initial layer serves as an input layer, capturing images with a shape of 224x224 pixels and

three RGB color channels. This input layer comprises three convolutional layers employing 32 and 64 filters, utilizing a 3x3 kernel size and the ReLU activation function. Subsequently, max pooling layers are applied to reduce the spatial dimensions of the features. The Flatten layer is then employed to flatten the output of the max pooling layer into a one-dimensional array. Following this, the flattened layer is passed through dense layers consisting of 2048 and 1024 units, respectively. In order to enhance the training process and mitigate overfitting, a dropout layer is introduced after each dense layer, with a dropout rate set at 0.3. Finally, the output layer is composed of two units and utilizes as sigmoid activation function.

$$f(x) = 1/(1+\exp(-x)) \quad (2)$$

This updated CNN architecture incorporates dropout and batch normalization techniques, which enhances model accuracy and stability.

3.3.2 Modified MobilenetV2

MobileNetV2 is a variant of the MobileNet architecture, specifically designed for mobile and embedded devices with limited processing capabilities, introduced by Google in 2018. The input shape for MobileNetV2 is defined as 224x224x3, and the pre-trained layers are kept non-trainable. To mitigate overfitting, a dropout layer with a dropout rate of 0.3 is included, followed by a dense layer comprising three units and employing a sigmoid activation function. The model is constructed using the Adam optimizer, with sparse categorical cross-entropy utilized as the loss function, and the accuracy metric employed for evaluation purposes.

$$Loss = -\sum n_{i=1}\, yi * \log \dot{y}_i \quad (3)$$

3.3.3 Modified Resnet50

ResNet50 is variant of the RN architecture, developed to mitigate the issue of vanishing gradients encountered in deep neural networks. It was introduced in 2015 by Kaiming He and his colleagues at Microsoft Research. ResNet50 is a deep neural network comprising 50 layers, predominantly consisting of convolutional layers. The original model is trained on a large-scale image classification dataset containing over 1000 categories. This adapted ResNet50 model possesses fewer trainable parameters compared to the original ResNet50 model, resulting in quicker training times and reduced computational demands.

$$H_p(q) = -\frac{1}{n} \sum_{i=1}^{N} z_i \log(p(z_i)) + (1\text{-}z_i)\log(1\text{-}p(z_i)) \quad 4$$

Furthermore, this model employs binary cross-entropy as its loss function and utilizes the Adam optimizer with a learning rate set to 0.001. In contrast, the original model might utilize various loss functions and optimizers depending on the specific training task at hand.

3.3.4 Modified VGG19

VGG19 is deep CNN architecture developed by researchers at the University of Oxford in 2014. It consists of a total of 19 layers, comprising 16 convolutional layers and three fully connected layers. The VGG19 architecture has undergone certain modifications, leveraging transfer learning techniques. In this approach, a pre-trained VGG19 model with weights from the ImageNet data set serves as the base model. The final convolutional block and dense layers from the original VGG19 model are removed, and two new dense layers, with 1024 and 2 units respectively, are added. The output of the pre trained VGG19 model is flattened and then passed through the newly added dense layers. The Adam optimizer's learning rate is set 0.001, and the model employs categorical cross-entropy as its loss function.

$$H(p,q) = -\sum x \; p(x) \log q\,(x) \quad 5$$

3.4 Classifier

3.4.1 Random Forest

Random Forest(RF) is a versatile method used for both classification and regression tasks. It operates by combining multiple decision trees to enhance prediction accuracy. The algorithm randomly selects a subset of features from the dataset and constructs numerous decision trees based on these selected features. The final prediction is generated by aggregating the predictions all of trees in the forest, typically through a majority vote for classification tasks or an average value for regression tasks. In the random forest classifier, the parameter n_estimators is set to 100, resulting in the creation of 100 decision trees. Additionally, a random seed value of 4020 is specified to ensure reproducibility of results if the code is rerun with the same seed. In the random forest algorithm, the most frequently predicted class is determined

through a voting mechanism. Equation for margin function.

$$f(x) = \arg \max \sum_{j=1}^{j} 1(y=h_j(x)) \quad (6)$$

$$mg(X,Y) = av_k I(h_k(X) = Y) - \max av_k I (h_k(X) = j)$$

3.4.2 K-Nearest Neighbors (KNN)

KNN operates by identifying the k nearest data points in the training set to a given test point and then leveraging the class or average value of these neighbors to predict the test point's classification or value. A smaller value of k results in a more complex model with decreased bias and increased variance. Conversely, a larger k value leads to a simpler model with higher bias and reduced variance. The optimal value of k is typically determined using model selection techniques such as cross-validation. In this scenario, the number of neighbors is set to 2, indicating that each prediction considers the two nearest neighbors.

The Minkowski metric specifies the distance metric to utilize for determining the distance between points in the dataset. The Minkowski distance metric (12) will be identical to the Euclidean distance metric. The equation of distant function (13) and h(x) (14) for KNN is:

$$dist(x,z) = \left(\sum_{r=1}^{d} |x_r - z_r|^p\right)^{1/p} \quad (7)$$

$$dist(x,x') \geq \max dist(x,x'')$$

$$h(x) = \text{mode}(\{y'':(x'',y'') \in S_x\})$$

3.4.3 AdaBoost

The AdaBoost classifier functions by combining multiple weak classifiers into a robust classifier, with each weak classifier specialized in a specific aspect of the data. AdaBoost employs an iterative process to sequentially train a series of weak classifiers on the training data. Each subsequent classifier focuses more on the instances that were misclassified by the preceding classifiers. AdaBoost assigns weights to training samples based on their previous misclassifications to prioritize difficult instances during subsequent iterations. The final result is then generated by aggregating the predictions of all weak classifiers, with each prediction weighted according to its classifier's accuracy.

In this implementation, the parameter n_estimators is set to 10, indicating the utilization of 10 weak classifiers to enhance the model's performance. Additionally, random seed value is set 2020 to ensure reproducibility of results if the code is executed again with the same random seed.

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$
$$\varepsilon_m = \sum y_i \neq k_m(x_i)\, w_i^{(m)} / \sum_{i=1}^{n} w_i^{(m)}$$
$$\varepsilon_m = \frac{\sum_{n=1}^{N} w_n^m \, l(f_m(x_n).y_n)}{\sum_{n=1}^{N} w_n^m} \quad 8$$

3.5 Tools and Libraries Used

3.5.1 TensorFlow

TensorFlow is a powerful and widely used software library that integrates artificial intelligence and machine learning capabilities. Developed primarily by the Google Brain team, TensorFlow is an open-source platform freely available for download. While TensorFlow is predominantly implemented in C++, it offers a Python interface for easy accessibility and interaction with the underlying C++ framework. Python serves as a user-friendly interface for leveraging TensorFlow's capabilities, allowing for seamless execution of computing, ML, and deep learning tasks.

TensorFlow employs symbolic mathematical computations, data flow, and differentiable programming, with a primary focus on training deep neural networks. While TensorFlow code is primarily written in Python, C++ is often utilized for the development of new techniques or features. TensorFlow finds application in various domains of deep learning, providing pre-built structures for implementing methodologies such as CNN, commonly used in tasks like computer vision and natural language processing.

The platform supports a wide array of deep learning applications, including image processing, sequence labeling, classification, prediction, and more. With comprehensive support ranging from model training to deployment, TensorFlow facilitates tasks such as handwritten digit classification, image recognition, word embedding, recurrent neural networks, machine translation using sequence models, and NLP.

One of TensorFlow's key strengths lies in its ability to seamlessly transition from model

training to large-scale production, utilizing the same models across different stages of development. Its versatility empowers researchers and scientists to tackle complex ML tasks efficiently, with the convenience of a user-friendly Python interface.

3.5.2 Kaggle

Kaggle is widely recognized as a premier platform offering convenient access to a plethora of materials and datasets pertinent to ML and data science endeavors. Particularly renowned for its robust support for deep learning applications like neural networks, Kaggle stands out as a comprehensive hub for data exploration, model building, and collaboration among data scientists and machine learning enthusiasts.

The platform boasts an array of features designed to facilitate dataset discovery, exploration, and model development, all within a user-friendly website interface. Moreover, Kaggle fosters a vibrant community of data scientists and machine learning practitioners, encouraging collaboration and knowledge sharing to tackle diverse challenges within the field.

One of Kaggle's distinctive offerings is its regular competitions, providing users and participants with opportunities to showcase their skills and expertise. These competitions serve as an avenue for individuals, whether novices or seasoned professionals, to hone their skills, engage in real-world problem-solving, and vie for recognition.

One of the primary advantages of Kaggle lies in its rich repository of datasets, providing invaluable resources and materials that contribute to an active and engaged community. Whether embarking on a learning journey or delving into a new project, Kaggle serves as an ideal platform for both beginners seeking to acquire new skills and experienced practitioners seeking to participate in competitive events.

Covering a wide spectrum of disciplines including Python programming, ML, data visualization, SQL, deep learning, natural language processing (NLP), and image processing, Kaggle offers comprehensive explanations and assistance on these topics, making it an indispensable and instructive resource for anyone with an interest in the field of data science and ML.

3.5.3 Scikit Learn

Scikit-learn stands out as a robust and widely adopted ML toolkit within the Python ecosystem. Leveraging the capabilities of renowned Python libraries like NumPy and SciPy, Scikit-learn offers a comprehensive suite of techniques catering to various ML tasks, including classification, regression, clustering, and both supervised and unsupervised learning problems.

Designed with simplicity and efficiency in mind, Scikit-learn seamlessly interfaces with popular Python libraries such as NumPy, SciPy, and Matplotlib, facilitating smooth integration into existing Python workflows. It encompasses a plethora of ML algorithms, ranging from logistic regression and support vector machines (SVM) to random forests, empowering data scientists to tackle diverse tasks including classification, regression, and clustering with ease.

Scikit-learn provides users with a rich array of features for data exploration, feature engineering, model training, and evaluation, offering a user-friendly and consistent design that caters to both novice and experienced data scientists and ML practitioners alike. Its intuitive interface and extensive documentation make it an accessible and indispensable tool for rapid and successful execution of various machine learning tasks.

Renowned for its ease of use and robustness, Scikit-learn remains a popular choice among Python users seeking to leverage ML capabilities for a wide range of applications, thanks to its user-friendly design and comprehensive feature set.

3.5.4 OpenCV

OpenCV stands out as a sophisticated and widely utilized toolkit renowned for its capabilities in real-time computer vision, ML, and image processing applications. It holds a paramount position within the image processing industry, offering a diverse array of tools and functionalities tailored for various image processing and computer vision tasks.

One of the key strengths of OpenCV lies in its versatility, providing a plethora of functions encompassing tasks such as face detection, object tracking, landmark recognition, and more. It boasts support for multiple programming languages, including Python, Java, and C++, making it accessible to a broad range of developers and researchers.

OpenCV serves as a comprehensive resource for academics and students delving into the realms of computer vision and image processing, offering a rich library of content and functionalities to support learning and research endeavors. Its capabilities extend to analyzing and manipulating images, facilitating tasks like rotation at arbitrary angles, downsampling, and smoothing through filters like Gaussian blur, thereby enhancing processing efficiency.

Notably, OpenCV excels in providing efficient and optimized methods for real-time image processing, enabling developers and researchers to deploy advanced computer vision techniques and algorithms seamlessly. It finds widespread application across various domains, including object recognition, augmented reality, robotics, and surveillance systems, owing to its robust functionalities and ease of integration into diverse projects.

3.5.5 Matplotlib

Matplotlib stands out as a widely embraced and powerful Python library renowned for its prowess in data visualization, offering a plethora of high-quality plots and charts that effectively convey insights from data.

With Matplotlib, users can effortlessly construct an array of visualizations, including line plots, scatter plots, bar plots, histograms, pie charts, and more. Its expansive set of customization options empowers users to finely adjust every aspect of their plots, from colors, line styles, and markers to axis labels, titles, and legends, ensuring that visualizations align precisely with their requirements.

One of Matplotlib's notable strengths lies in its seamless integration with other Python libraries like NumPy and Pandas, facilitating the visualization of data arrays and simplifying data analysis tasks. Moreover, its compatibility with Jupyter Notebook enables interactive graphing and exploration, enhancing the user experience and enabling dynamic data visualization workflows.

Matplotlib finds widespread application across various domains, including data science, machine learning, scientific research, and data visualization, enabling users to discern patterns, trends, distributions, and relationships within their data. It fosters data exploration, analysis, and presentation, offering a range of output formats, including interactive charts and static images suitable for publication. Additionally, Matplotlib's support for animated visualizations adds an extra dimension to presentations and web applications, enhancing engagement and comprehension.

In summary, Matplotlib serves as an indispensable tool for crafting visually captivating and informative plots and charts in Python. Its versatility, extensive customization capabilities, and seamless integration make it an essential component of the data visualization and analysis toolkit, empowering users to create compelling visualizations that effectively communicate insights from their data.

3.6 Train Test Split

In the realm of deep learning model training, it's a common practice to partition the dataset into three distinct sections: training, testing, and validation. Typically, when we acquire a primary dataset, it arrives as a unified directory or file. To effectively work with the dataset, it's imperative to preprocess and segment it into appropriate subsets for training and testing.

Partitioning the dataset into training and testing subsets is often necessary, and the scikit-learn module offers a handy method for this task.

This function in scikit-learn is commonly employed to split arrays or datasets into separate subsets for training and testing. It accepts various parameters, including the arrays or datasets to be divided, the desired size of the testing set, and options for randomization. By invoking this method and providing the necessary arguments, the dataset can be randomly divided into distinct training and testing subsets. This splitting process ensures that the model is trained on a portion of the data and then evaluated on previously unseen data to gauge its performance.

The validation component is frequently treated as a separate entity, and it can be derived from the training data or combined with the testing data to fine-tune and validate the model. Employing this technique enables us to properly prepare the dataset for deep learning model training, ensuring that the model is trained on a meaningful fraction of the data and evaluated on independent data to assess its generalization capabilities.

In our specific case, we allocated 80% of the data for training, resulting in 3738 entries in our training subset. The remaining 20% of the data was reserved for testing, yielding approximately 935 items in the test subsets. When the original data is not initially divided into training and testing sections, it's common to train models using the entire dataset, often leading to overfitting. To mitigate this issue, dividing the data into separate subsets for training, testing, and validation is essential.

The training data is utilized to train the model, the testing data is used to evaluate the model's performance on unseen data, and the validation data aids in fine-tuning the model and making decisions regarding hyperparameter tuning or model selection. It's crucial to emphasize that manipulating testing data is highly unethical and can yield biased and misleading results. Testing data should remain separate and should not be used for model development purposes.

3.7 Evaluation Metrics

3.7.1 Precision

The precision metric serves as a valuable measure for evaluating the performance of classification models. It is calculated by dividing the total number of true positive samples by the sum of true positive and false positive samples. Precision essentially gauges the model's ability to correctly identify positive samples among those it has predicted as positive. Beyond its application in classification models, precision can also denote the level of correctness or concordance between multiple measurements. It offers an estimation of the similarity or

concordance among various measurements or values. Precision reflects the level of exactitude or accuracy with which a specific dimension or value is determined or measured. It indicates how closely related or congruent two or more measurements are to each other, thereby illustrating the degree of precision in the measurement process.

$$\text{Pr}\, e\, cision = \frac{Tp}{Tp + Fp} \quad (9)$$

3.7.2 Accuracy

The accuracy metric is commonly employed to evaluate the effectiveness of classification models. Nevertheless, accuracy might not be the most suitable metric in certain scenarios, especially when dealing with datasets characterized by imbalanced classes. In such cases, a classifier that consistently predicts the majority class for every instance can yield a high accuracy score while overlooking the minority class entirely. Given that our dataset encompasses multiple classes, the presence of class imbalances could potentially distort the accuracy results, highlighting the need for alternative evaluation metrics that provide a more comprehensive assessment of model performance.

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \quad (10)$$

3.7.3 Recall

Recall, also known as sensitivity or true positive rate, is a crucial evaluation metric for effectively identifying positive events. It measures a classifier's ability to correctly identify all positive instances within a dataset. True positives (TP) represent instances accurately identified as positive by the classifier. Conversely, false negatives occur when events that are actually positive are incorrectly classified as negative by the classifier.

$$Recall = \frac{(Number\ of\ true\ positives)}{(Number\ of\ true\ positives + Number\ of\ false\ negatives)} \quad (11)$$

3.7.4 F1 Score

The F1 score is the metric that offers a balanced assessment of a classifier's performance by

considering both precision and recall. It proves valuable in scenarios involving imbalanced datasets or when there's an uneven cost linked to false positives and false negatives. By combining precision and recall, the F1 score provides a single metric that balances the trade-off between the two. It ranges from 0 to 1, where 1 represents the best possible F1 score and 0 indicates the worst.

$$F1\ Score = \frac{2 \times (precision \times Recall)}{(\ \mathrm{Pr}\, e\ cision\ +\ Recall)} \qquad 12$$

Chapter 4

Results or findings

4.1 Analysis of CNN models for original dataset

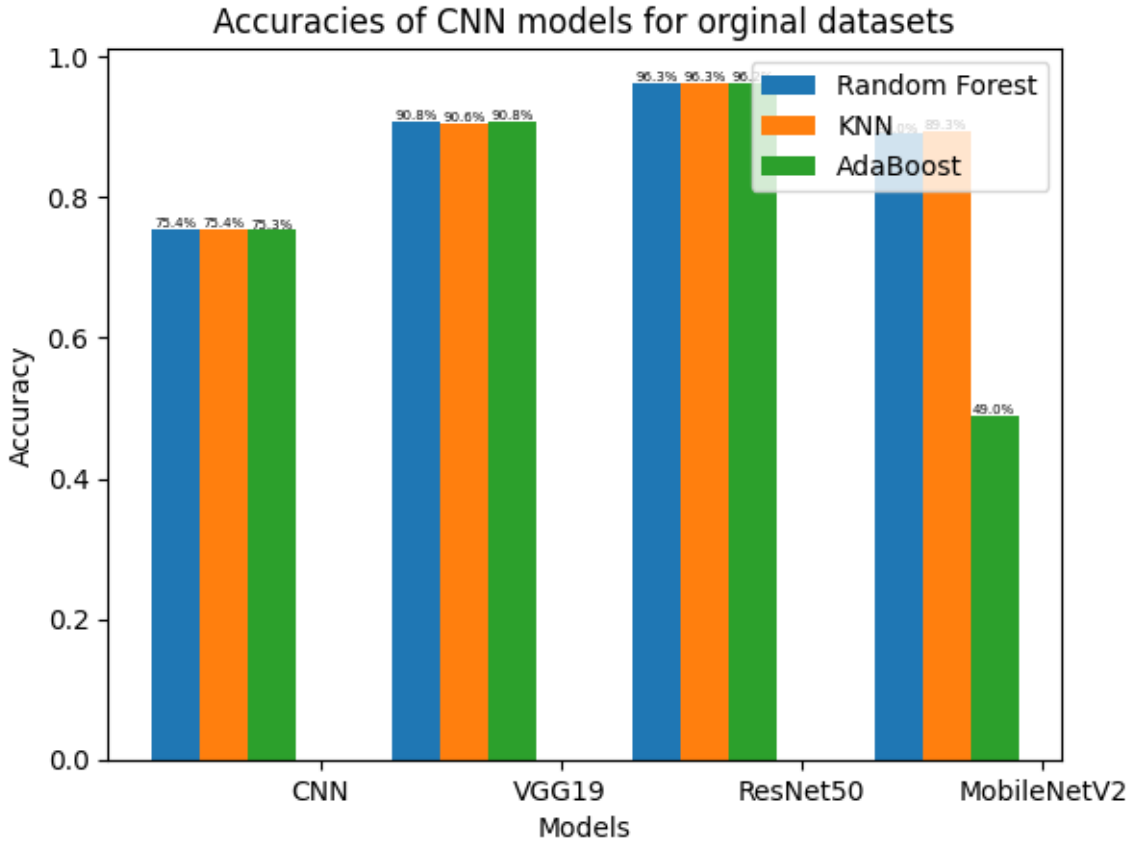

Figure 4.1: Histogram of the accuracy of CNN models on original dataset

For the Scratched CNN model, the Random Forest and KNN classifiers achieved an accuracy of 75.4%, while AdaBoost closely followed with an accuracy of 75.3%. Moving on to the MobileNetV2 model, we observed significantly higher accuracies across all classifiers, with Random Forest achieving 90.8%, KNN at 90.6%, and AdaBoost at 90.8%. The ResNet50 model displayed remarkable accuracy, with all three classifiers achieving an impressive 96.3%. Lastly, for the VGG19 model, Random Forest and KNN both reached accuracies of 89%, while AdaBoost lagged behind with an accuracy of 49%.

4.1.1 Performance Result

Table 4.1: Accuracy results for original datasets.

Model	Accuracy	Loss	Validation Accuracy	Validation Loss
Scratched CNN	1.0000	5.5317e-06	0.7539	2.8737
MobileNetV2	0.9951	0.0175	0.8919	0.5731
ResNet50	1.0000	1.7019e-04	0.9631	0.1485
VGG19	0.9933	0.0250	0.9080	0.7827

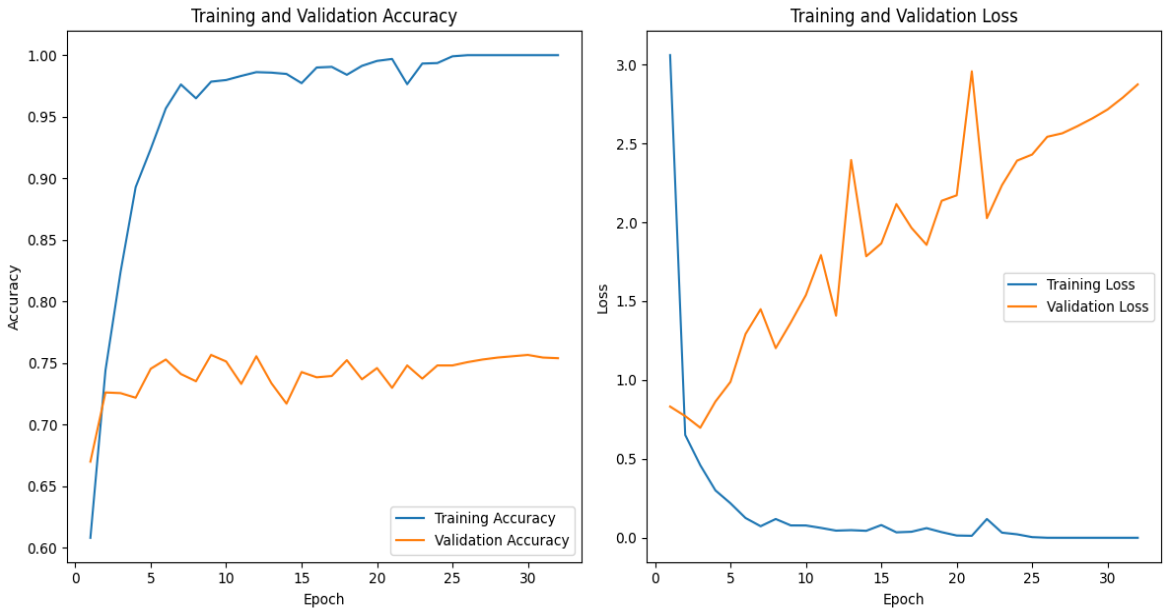

Figure 4.2: Performance Result of Scratched CNN

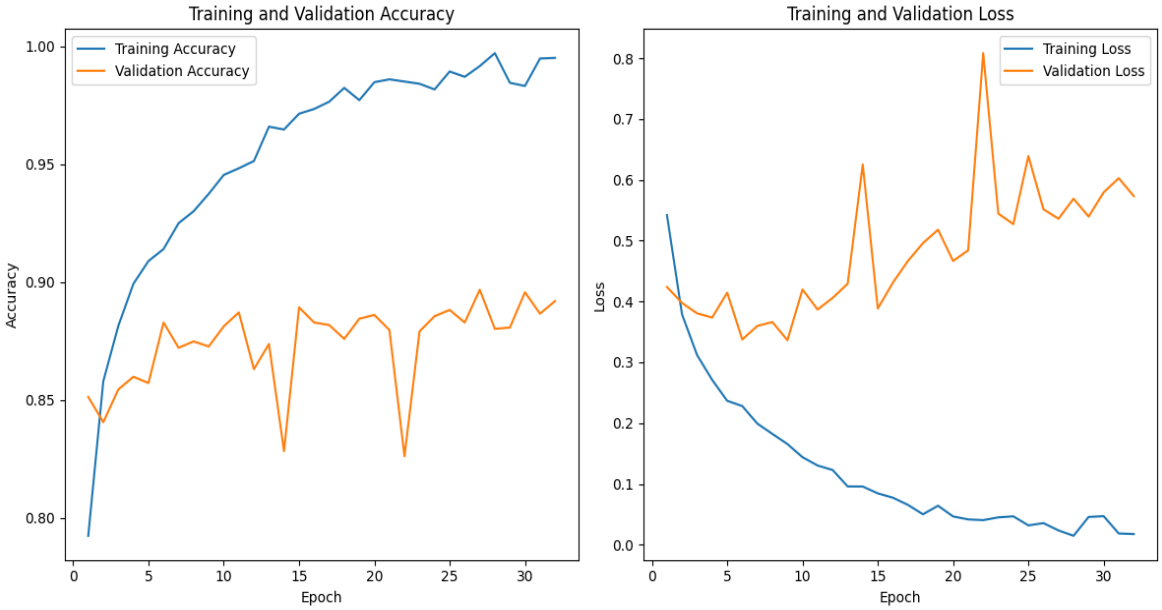

Figure 4.3: Performance Result of mobilenetV2

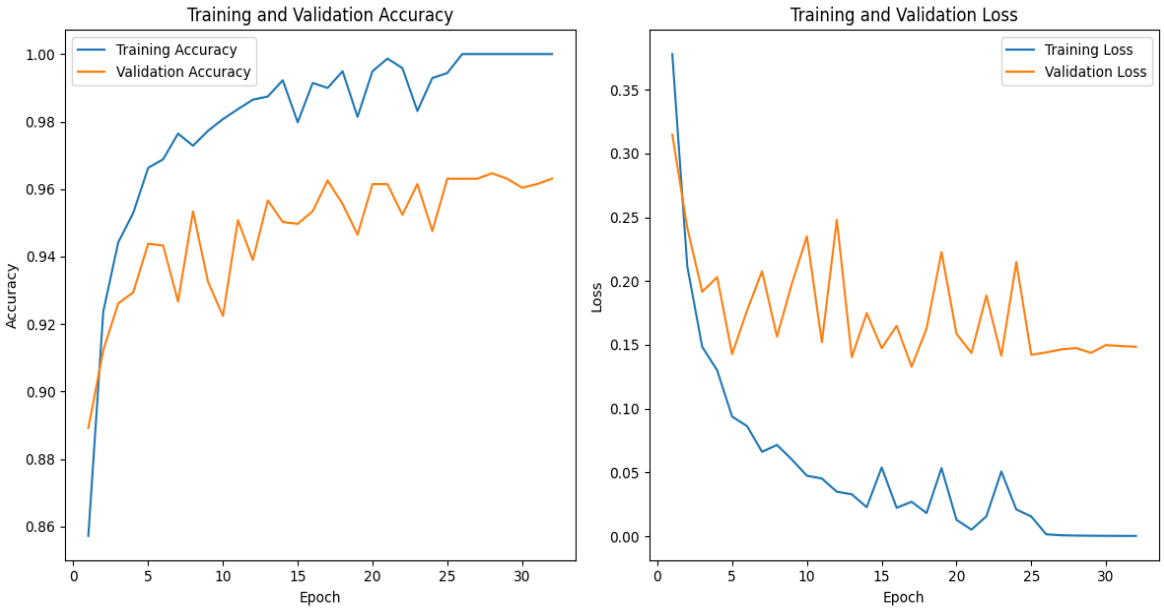

Figure 4.4: Performance Result of Resnet50

Fig 4.5: Performance Result of VGG19

4.1.2 Performance Metrics

Table 4.2: Precision, Recall, F1 Score for Random Forest

Models	Precision	Recall	F1-Score
Scratched CNN	0.7551	0.7539	0.7544
MobileNetV2	0.8905	0.8903	0.8903
ResNet50	0.9632	0.9631	0.9631
VGG19	0.9087	0.9085	0.9084

Table 4.3: Precision, Recall, F1 Score for KNN

Models	Precision	Recall	F1-Score
Scratched CNN	0.7553	0.7539	0.7545
MobileNetV2	0.8937	0.8935	0.8935
ResNet50	0.9632	0.9631	0.9631
VGG19	0.9055	0.9058	0.9055

Table 4.4: Precision, Recall, F1 Score for AdaBoost

Models	Precision	Recall	F1-Score
Scratched CNN	0.7548	0.7533	0.7539
MobileNetV2	0.4108	0.4896	0.3655
ResNet50	0.9621	0.9620	0.9620
VGG19	0.9085	0.9085	0.9081

4.2 Analysis of CNN models for gaussian filter

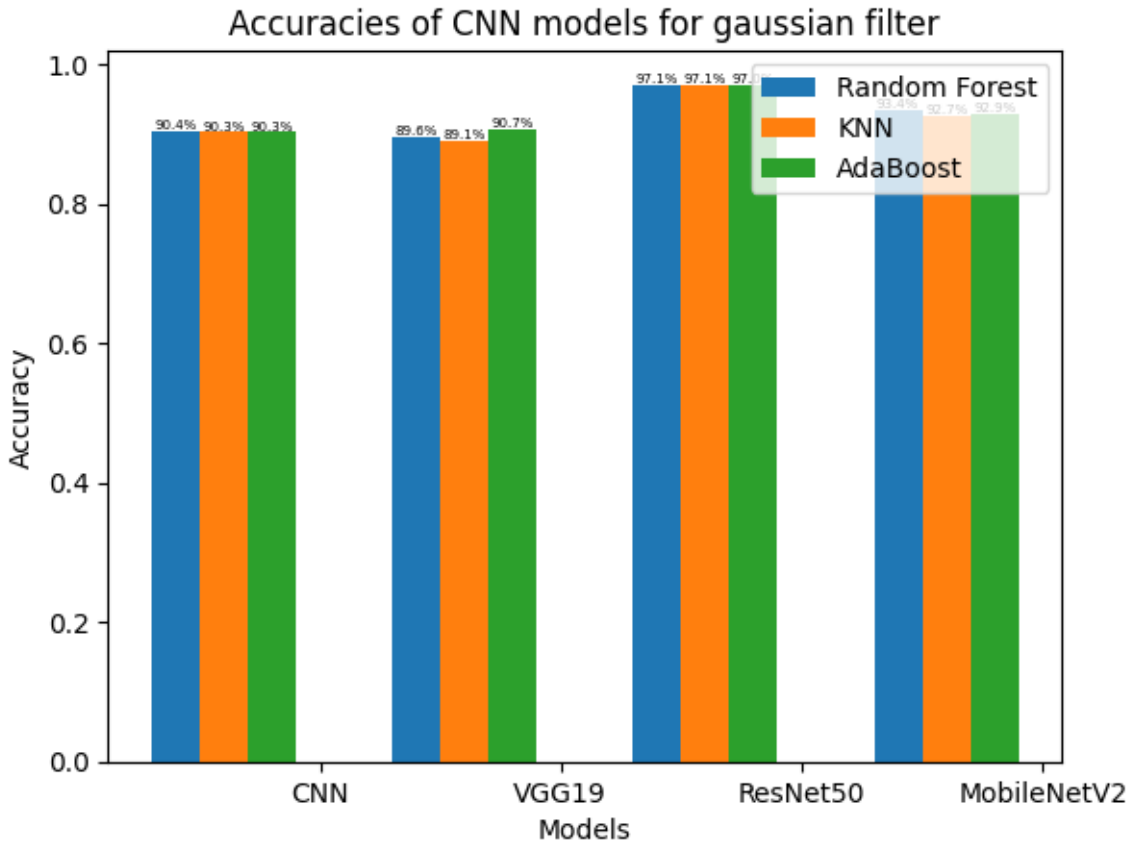

Figure 4.6: Histogram of the accuracy of CNN models on gaussian dataset

For the Scratched CNN model, Random Forest, KNN, and AdaBoost exhibited strong performances, achieving accuracies of 90.4%, 90.3%, and 90.3%, respectively. MobileNetV2 displayed consistent results across classifiers, with Random Forest at 89.6%, KNN at 89.1%, and AdaBoost at 90.7%. ResNet50 emerged as the standout model with remarkable accuracy, where all three classifiers achieved an impressive 97.1%. VGG19, while showcasing commendable accuracy, presented slight variations among classifiers, with Random Forest at

93.4%, KNN at 92.7%, and AdaBoost at 92.9%

4.2.1 Performance Result

Table 4.5: Accuracy results for gaussian filter datasets.

Model	Accuracy	Loss	Validation Accuracy	Validation Loss
Scratched CNN	1.0000	3.0788e-06	0.9027	0.9513
MobileNetV2	0.9990	0.0057	0.9308	0.2995
ResNet50	1.0000	3.3650e-04	0.9709	0.1122
VGG19	0.9862	0.0687	0.9057	0.6382

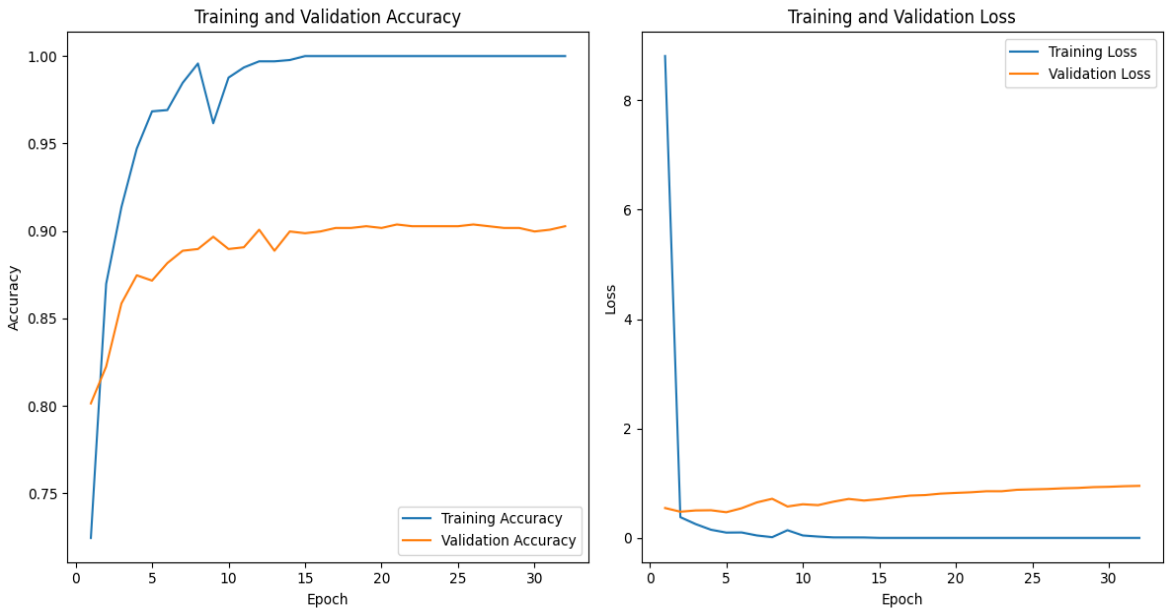

Figure 4.7: Performance Result of Scratched CNN

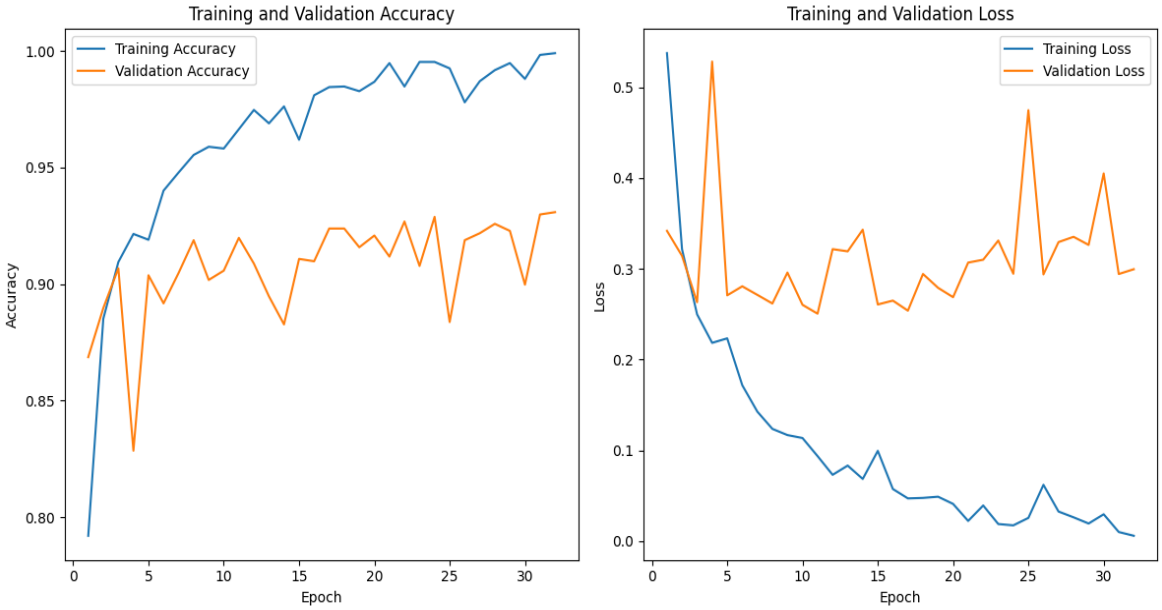

Fig 4.8: Performance Result of mobilenetV2

Fig 4.9: Performance Result of Resnet50

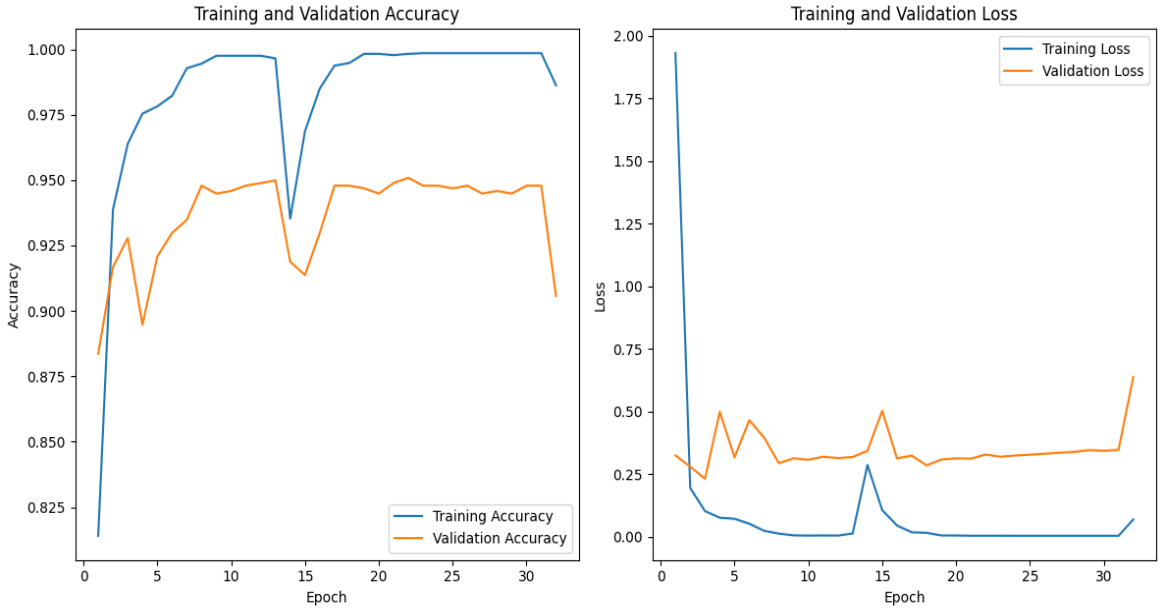

Fig 4.10: Performance Result of VGG19

4.2.2 Performance Metrics

Table 4.6: Precision, Recall, F1 Score for Random Forest

Models	Precision	Recall	F1-Score
Scratched CNN	0.9038	0.9037	0.9037
MobileNetV2	0.9345	0.9338	0.9338
ResNet50	0.9709	0.9709	0.9709
VGG19	0.8951	0.8957	0.8952

Table 4.7: Precision, Recall, F1 Score for KNN

Models	Precision	Recall	F1-Score
Scratched CNN	0.9029	0.9027	0.9027
MobileNetV2	0.9265	0.9268	0.9263
ResNet50	0.9710	0.9709	0.9709
VGG19	0.8901	0.8907	0.8901

Table 4.8: Precision, Recall, F1 Score for AdaBoost

Models	Precision	Recall	F1-Score
Scratched CNN	0.9028	0.9027	0.9027
MobileNetV2	0.9267	0.9288	0.9288
ResNet50	0.9699	0.9699	0.9699
VGG19	0.9062	0.9067	0.9062

4.3 Analysis of CNN models for Laplacian filter

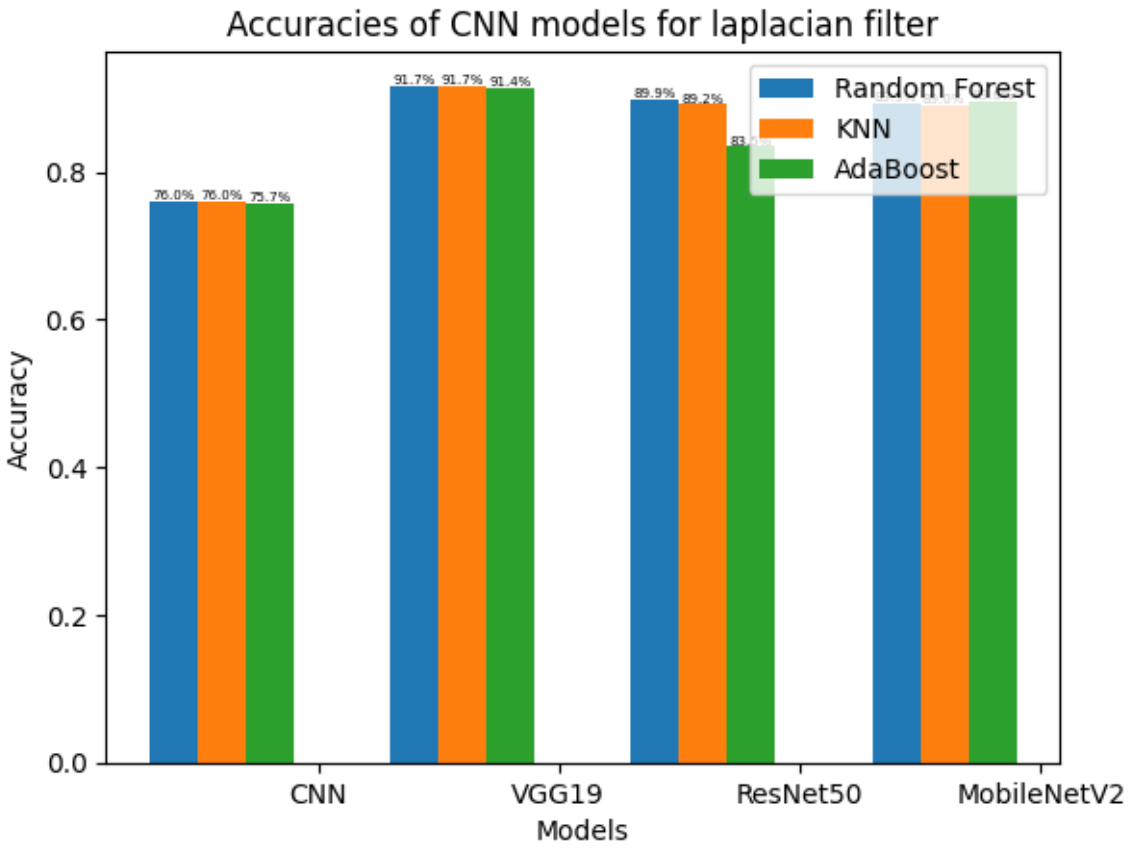

Figure 4.11: Histogram of the accuracy of CNN models on Laplacian dataset

For the Scratched CNN model, all three classifiers demonstrated competitive accuracy rates, with Random Forest and KNN both reaching 76%, and AdaBoost closely trailing at 75.7%. In contrast, MobileNetV2 showcased superior accuracy across the board, with Random Forest, KNN, and AdaBoost achieving impressive accuracies of 91.7%, 91.7%, and 91.4%,

respectively. ResNet50, while exhibiting robust accuracy with Random Forest and KNN at 89.9% and 89.2%, faced a slight dip with AdaBoost, landing at 83.4%. Lastly, the VGG19 model demonstrated consistent performance among classifiers, with Random Forest, KNN, and AdaBoost attaining accuracies of 89.3%, 89%, and 89.6%, respectively.

4.3.1 Performance Result

Table 4.9: Accuracy results for Laplacian filter datasets.

Model	Accuracy	Loss	Validation Accuracy	Validation Loss
Scratched CNN	0.9588	0.1217	0.7623	3.2778
MobileNetV2	0.9719	0.0841	0.8917	0.4385
ResNet50	0.9573	0.1218	0.8806	0.3916
VGG19	0.9859	0.0365	0.9107	0.7806

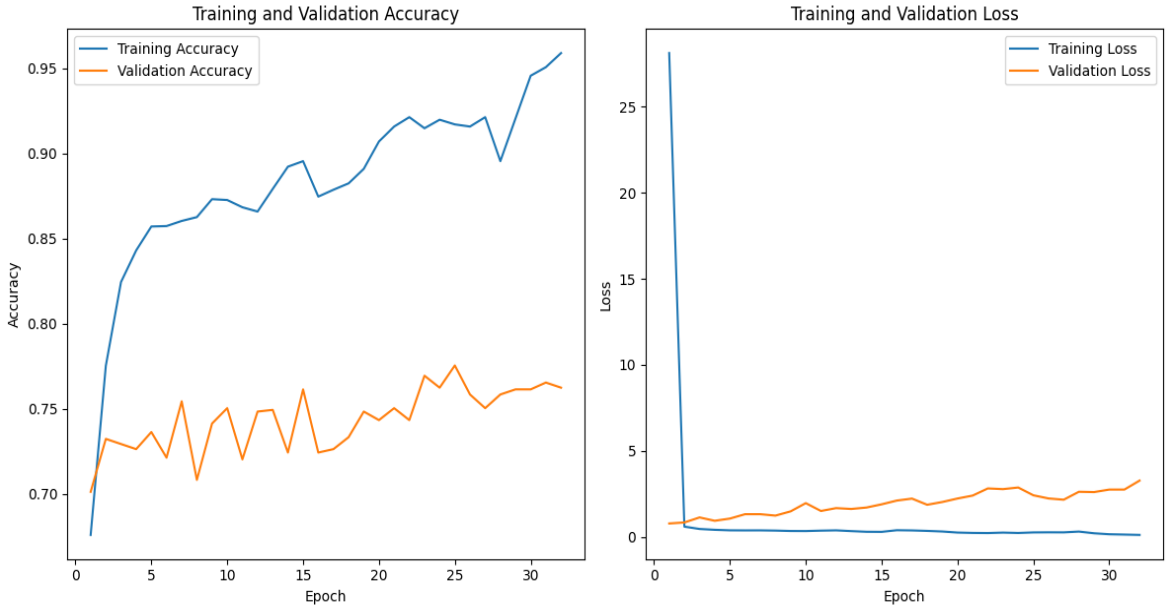

Figure 4.12: Performance Result of Scratched CNN

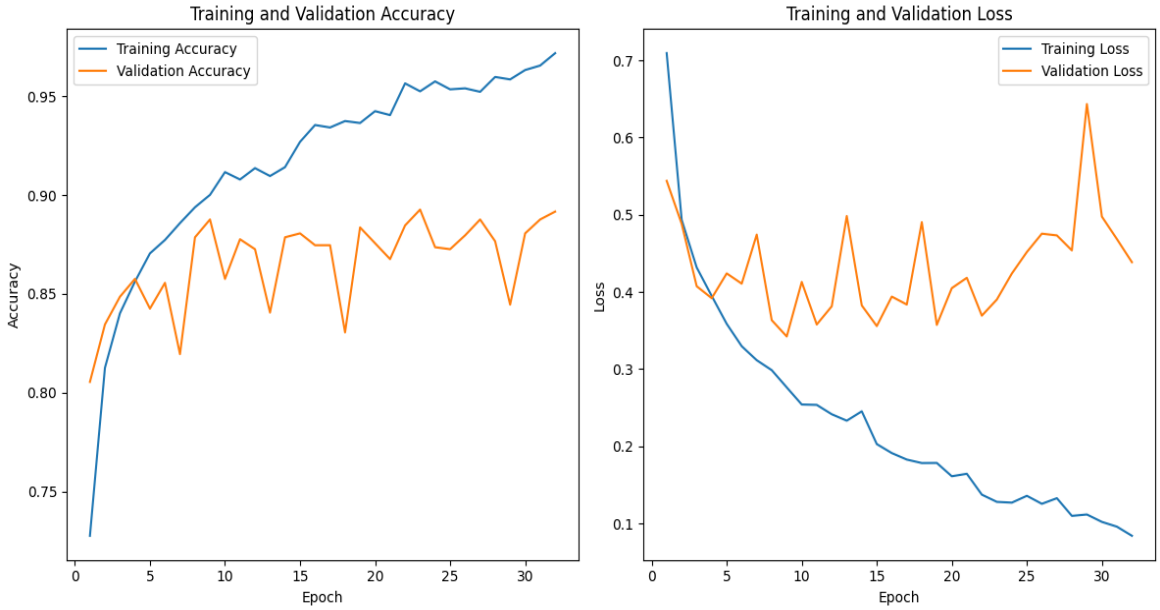

Figure 4.13: Performance Result of mobilenetV2

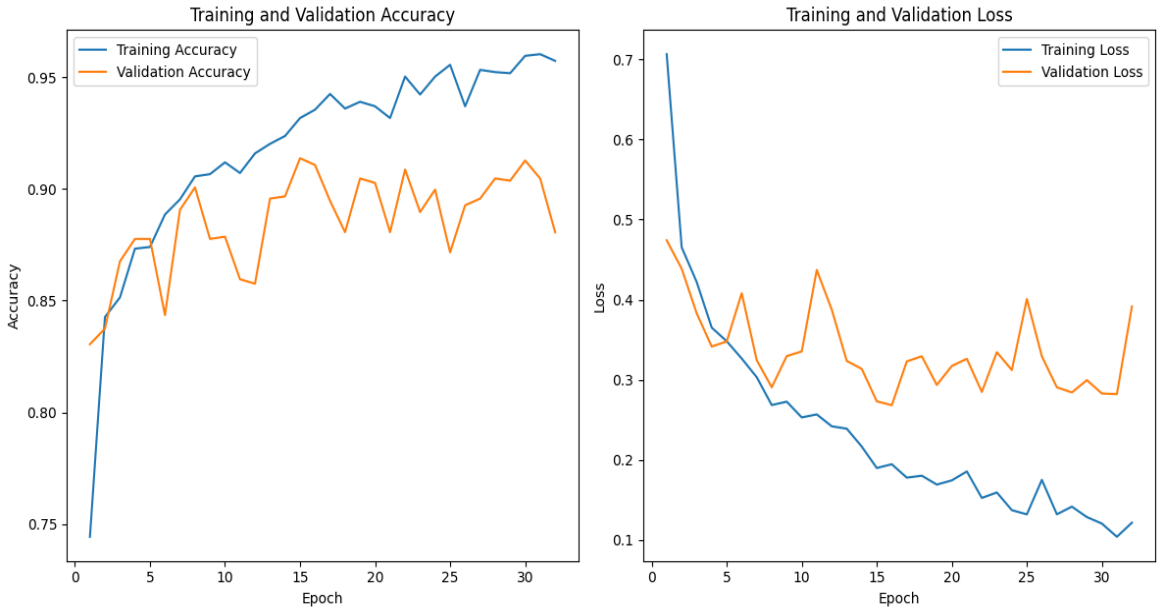

Figure 4.14: Performance Result of Resnet50

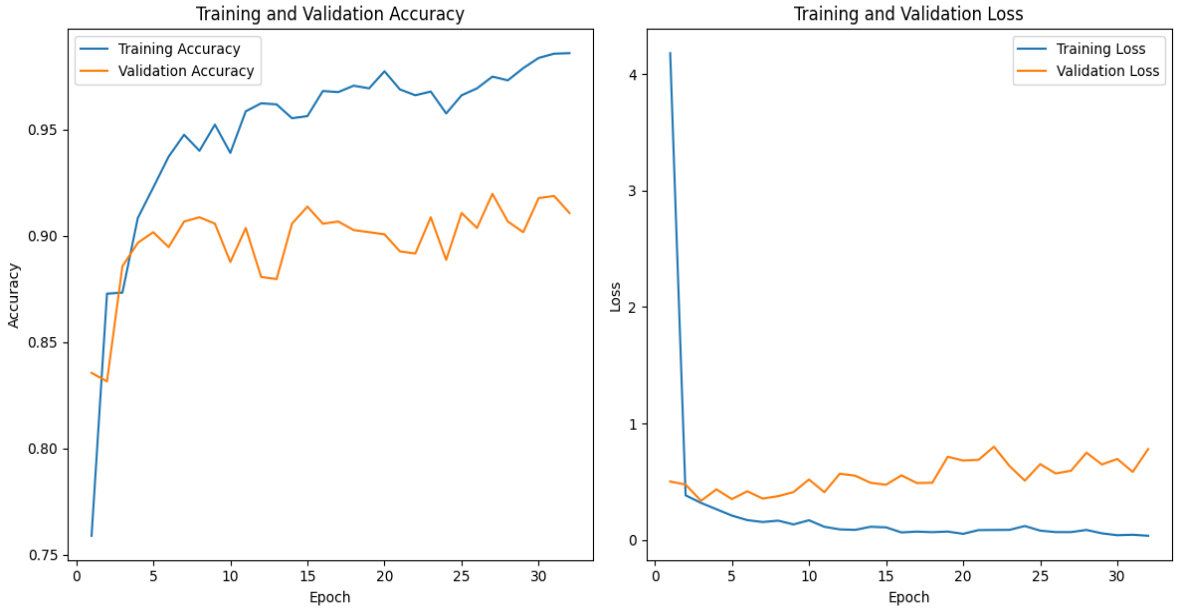

Figure 4.15: Performance Result of VGG19

4.3.2 Performance Metrics

Table 4.10: Precision, Recall, F1 Score for Random Forest

Models	Precision	Recall	F1-Score
Scratched CNN	0.7628	0.7603	0.7610
MobileNetV2	0.8925	0.8927	0.8924
ResNet50	0.8994	0.8987	0.8989
VGG19	0.9170	0.9168	0.9166

Table 4.11: Precision, Recall, F1 Score for KNN

Models	Precision	Recall	F1-Score
Scratched CNN	0.7618	0.7603	0.7605
MobileNetV2	0.8892	0.8897	0.8892
ResNet50	0.8927	0.8917	0.8920
VGG19	0.9169	0.9168	0.9165

Table 4.12: Precision, Recall, F1 Score for AdaBoost

Models	Precision	Recall	F1-Score
Scratched CNN	0.7582	0.7573	0.7572
MobileNetV2	0.8952	0.8957	0.8945
ResNet50	0.8591	0.8345	0.8373
VGG19	0.9146	0.9137	0.9129

4.4 Analysis of CNN models for unsharp mask filter

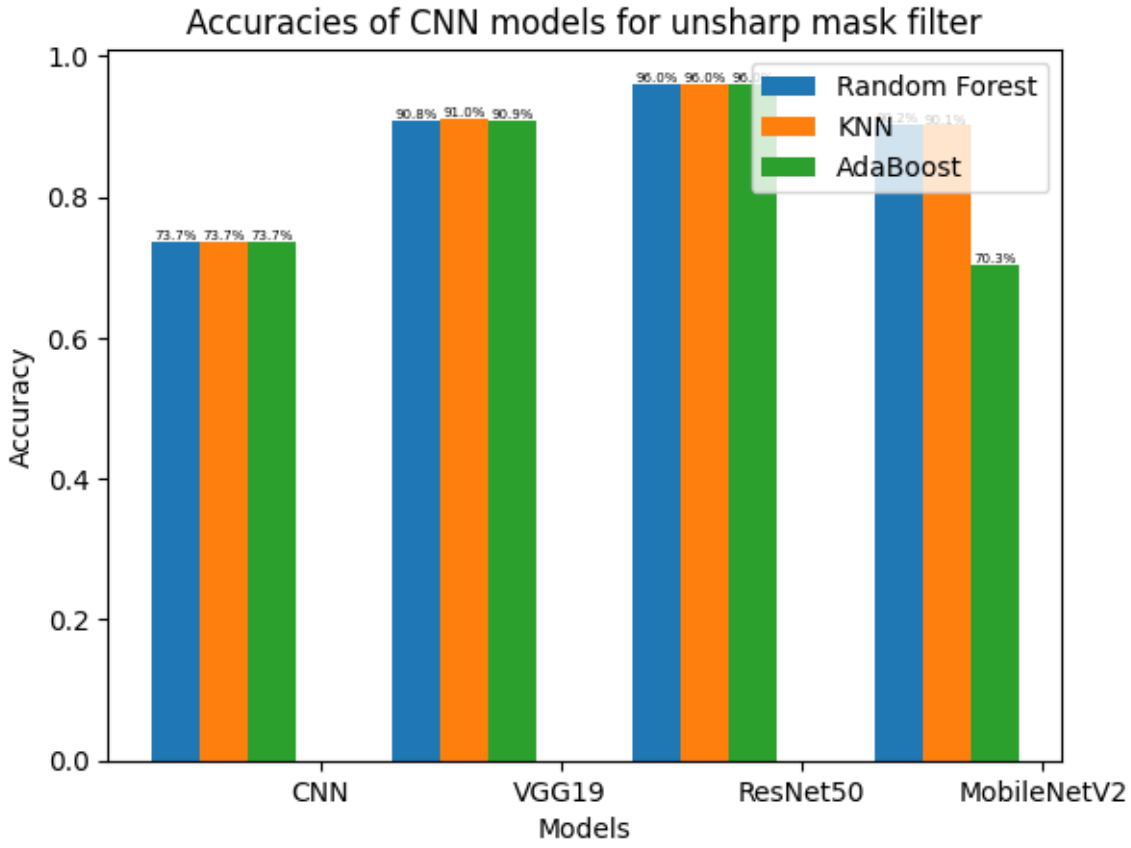

Figure 4.16: Histogram of the accuracy of CNN models on unsharp mask dataset

For the Scratched CNN model, all three classifiers exhibited consistent accuracies, with Random Forest, KNN, and AdaBoost achieving a solid 73.7%. MobileNetV2 showcased strong capabilities in handling the Unsharp Mask Filter dataset, with Random Forest, KNN, and AdaBoost achieving accuracies of 90.8%, 91%, and 90.9%, respectively. ResNet50 demonstrated exceptional performance across all classifiers, boasting an impressive accuracy rate of 96%. In contrast, VGG19 faced some challenges, particularly with AdaBoost, resulting in a lower accuracy of 70.3%, while Random Forest and KNN held steady at 90.2% and

90.1%, respectively.

4.4.1 Performance Result

Table 4.13: Accuracy results for unsharp mask filter datasets.

Model	Accuracy	Loss	Validation Accuracy	Validation Loss
Scratched CNN	1.0000	1.8084e-06	0.7373	3.0253
MobileNetV2	0.9951	0.0163	0.9006	0.4511
ResNet50	1.0000	2.0886e-04	0.9605	0.1830
VGG19	0.9984	0.0049	0.9113	0.8677

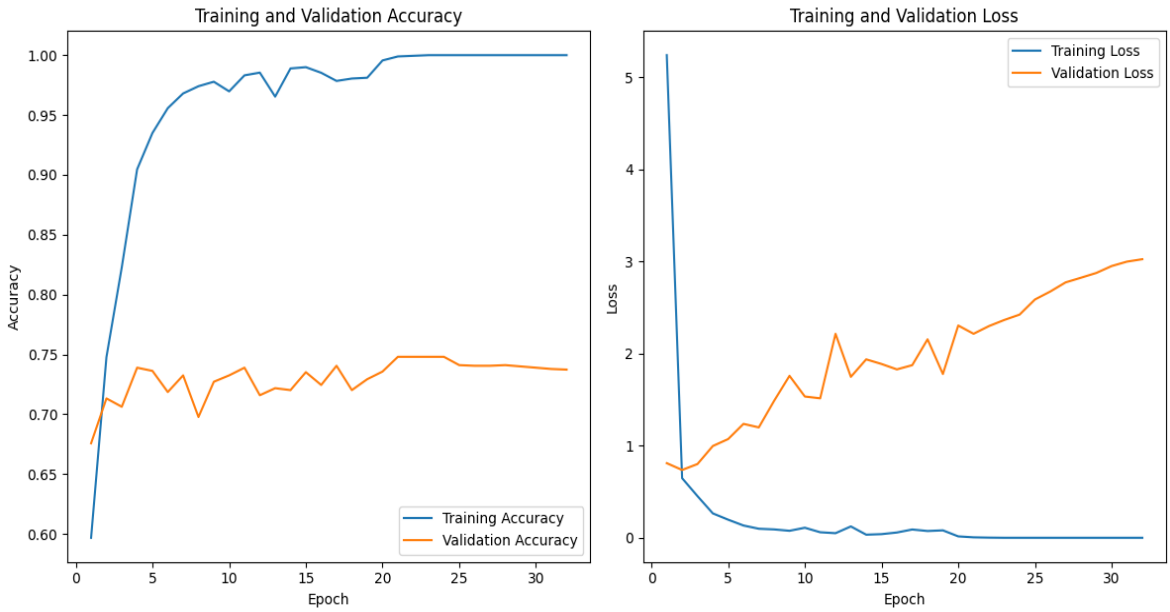

Figure 4.17: Performance Result of Scratched CNN

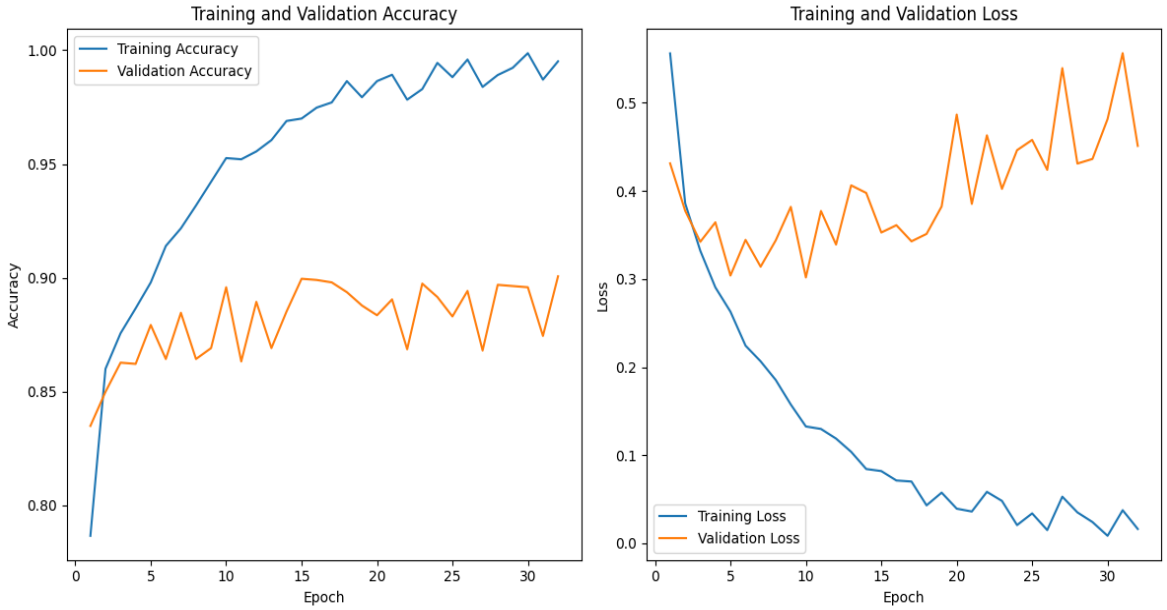

Figure 4.18: Performance Result of mobilenetV2

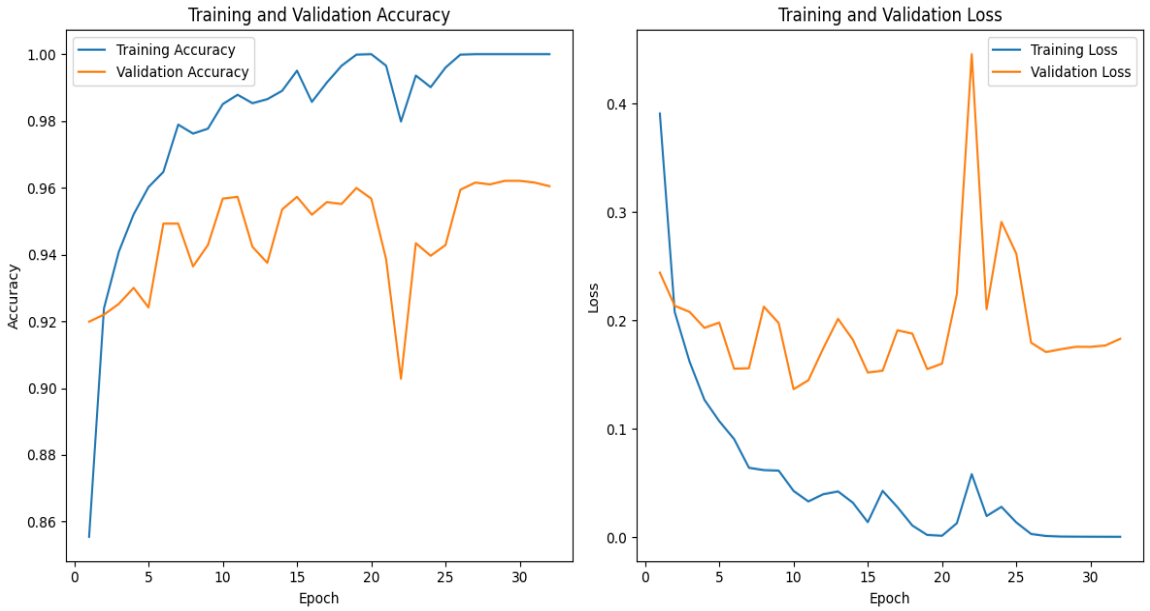

Figure 4.19: Performance Result of Resnet50

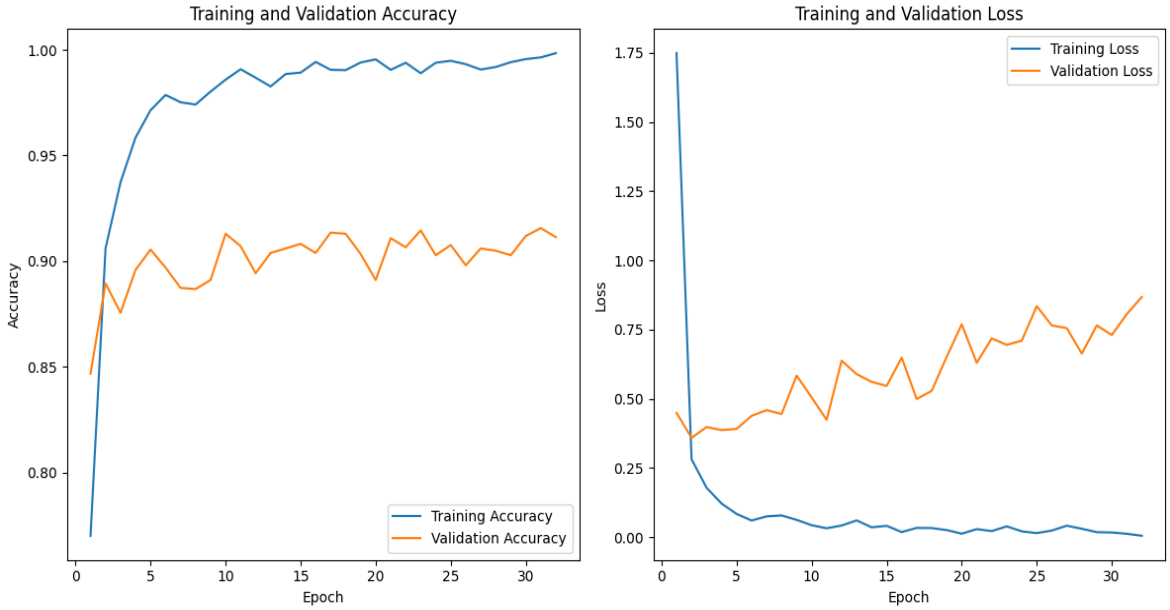

Figure 4.20: Performance Result of VGG19

4.4.2 Performance Metrics

Table 4.14: Precision, Recall, F1 Score for Random Forest

Models	Precision	Recall	F1-Score
Scratched CNN	0.7395	0.7368	0.7378
MobileNetV2	0.9015	0.9022	0.9018
ResNet50	0.9605	0.9605	0.9604
VGG19	0.9078	0.9076	0.9076

Table 4.15: Precision, Recall, F1 Score for KNN

Models	Precision	Recall	F1-Score
Scratched CNN	0.7403	0.7373	0.7384
MobileNetV2	0.9006	0.9012	0.9008
ResNet50	0.9605	0.9605	0.9604
VGG19	0.9105	0.9103	0.9102

Table 4.16: Precision, Recall, F1 Score for AdaBoost

Models	Precision	Recall	F1-Score
Scratched CNN	0.7395	0.7368	0.7378
MobileNetV2	0.5823	0.7035	0.6246
ResNet50	0.9599	0.9599	0.9599
VGG19	0.9087	0.9086	0.9085

4.5 Result Analysis

Table 4.17: Best model and classifier for every dataset

Dataset	Best Model	Best Classifier	Best Accuracy
Original Dataset	ResNet50	Random Forest	96.3%
Gaussian Filter Dataset	ResNet50	Random Forest/KNN	97.1%
Laplacian Filter Dataset	MobileNetV2	Random Forest	91.7%
Unsharp mask Filter Dataset	ResNet50	Random Forest/KNN	96%

In the Original Dataset, ResNet50 stands out as the best model, achieving impressive accuracies of 96.3% with Random Forest, KNN, and 96.2% with AdaBoost. The Gaussian Filter Dataset showcases ResNet50 again as the top model, achieving remarkable accuracies of 97.1% with Random Forest, KNN, and AdaBoost at 97%. In the Laplacian Filter Dataset, MobileNetV2 emerges as the preferred model with accuracies of 91.7% with Random Forest, KNN, and 91.4% with AdaBoost. Lastly, the Unsharp Mask Filter Dataset demonstrates ResNet50's dominance, reaching accuracies of 96% with Random Forest, KNN, and AdaBoost at 96%. Across classifiers, Random Forest consistently exhibits robust performance, making it a favorable choice for various datasets.

Based on the highest overall accuracy, the Gaussian Filter Dataset with the ResNet50 model and the Random Forest classifier stands out as the most favorable choice. This combination consistently achieves top-notch accuracy across various classifiers, making it a strong candidate for applications prioritizing high predictive performance.

4.6 Visual Representation of prediction

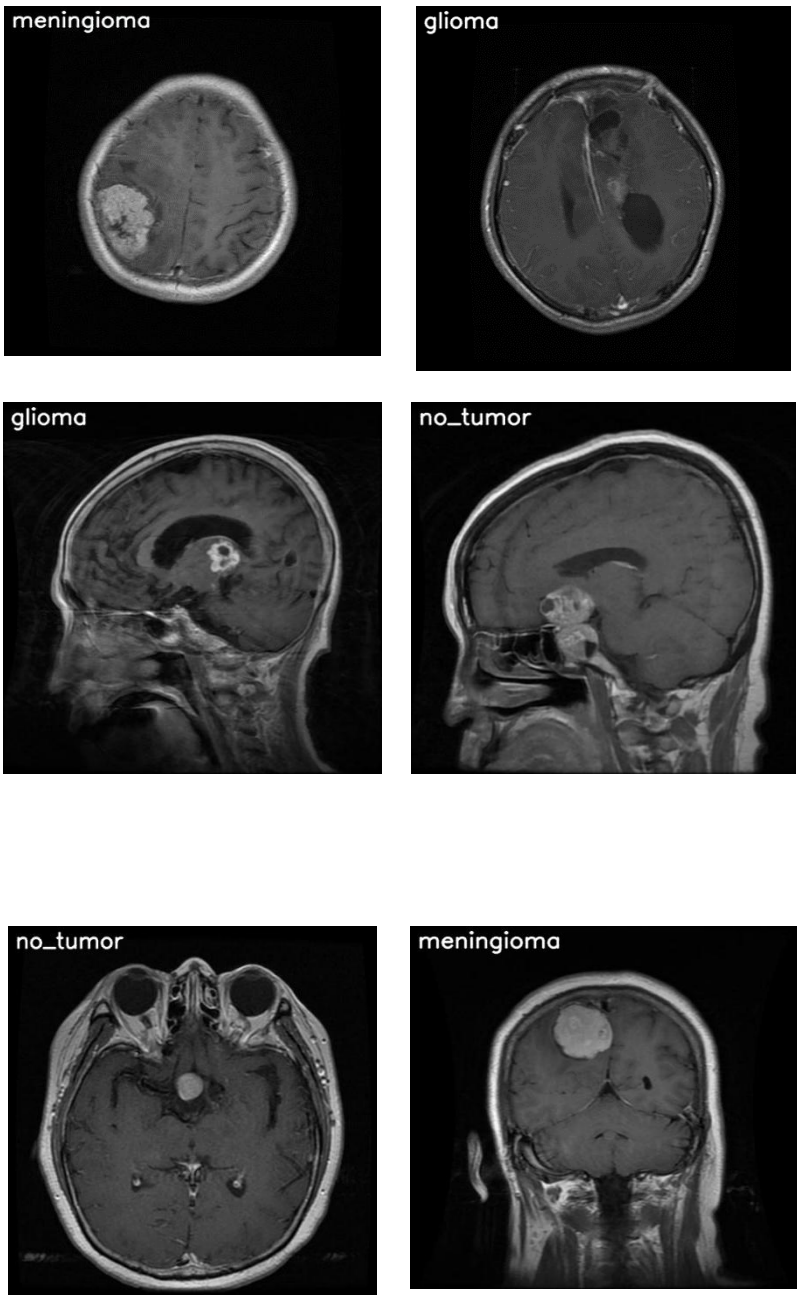

Figure 4.21: visual representation of prediction

Chapter 5

Discussion

This work suggests a novel method for identifying brain cancers from MRI data by combining image processing approaches. By filtering out the noise from the image, the tumor-affected area of the brain can be seen clearly. Classifiers are used in this thesis to classify and label the photos. Random Forest, KNN, and AdaBoost classifiers were employed. Laplacian, Unsharp mask, and Gaussian filters were employed for filtering. For feature extraction, the ResNet50, MobileNetV2, and Scratch CNN models were employed. In all four datasets, Random Forest and Scratched CNN outperform other models and classifiers. The Scratch CNN model with Random Forest classifier on the Laplacian dataset in this study had the maximum accuracy. There is 98.91% accuracy. Unsharp Mask is the second-best dataset, with 98.81% accuracy achieved using Random Forest and Scratch CNN. The Fl score, recall, and precision of several CNN models are also examined in this study.

5.1 Limitations

Taking into account the advancements and promising results reported in this paper, many limitations must be noted:

5.1.1 Limited Dataset Size

The findings' generalizability may be affected by the study's use of a very small dataset. Gaining access to more extensive and diverse datasets may facilitate the testing of the proposed methodology and provide a more comprehensive analysis.

5.1.2 Scope of Tumor Types

It's possible that the study focuses on a certain subset of brain tumor forms, which would limit how broadly it may be applied. Expanding the scope to include other tumor types would increase the effectiveness of the proposed technique.

5.1.3 Limited Clinical Validation

Despite having a high degree of accuracy in identifying brain tumors, the proposed technique may not have gotten enough clinical validation. Comprehensive clinical trials and validation studies with medical professionals and genuine patient data would be necessary to assess how well it functions in real-world scenarios.

5.1.4 Limited Comparison With Existing Methods

While the proposed method demonstrates commendable accuracy, conducting a direct comparison with other established state-of-the-art methods may pose challenges. However, undertaking comparative studies with existing methodologies could offer a benchmark for assessing the effectiveness and efficiency of the system under investigation.

5.1.5 Hardware and computational Requirements

It's possible that the study does not address the hardware and computing needs needed to put the suggested strategy into practice. Resource or computational limitations may make it more difficult to deploy and implement the system in real time.

Chapter 6

Conclusion

Modern medical science relies heavily on image processing to analyze a wide range of disorders. In this study, MRI pictures are used to detect and diagnose brain cancers. Image processing and machine learning classifiers are used to identify and categorize brain cancers. When it comes to brain tumor identification, the Random Forest classifier and the scratch Convolutional Neural Network (CNN) model perform better than the other methods. In our research, this combination produced the best outcomes.

6.1 Future Work

Our future goal is to improve brain tumor classification accuracy through sophisticated approaches and procedures. We will focus on specific tumor datasets such as meningioma, pituitary adenoma, and craniopharyngioma. We intend to work on other medical imaging, such as CT scans and X-rays. This approach aims to help clinicians make informed judgements about their patients' ultimate treatment. This effort will contribute to implementing a predicted accuracy model into Magnetic. Resonance Imaging (MRI) systems can give patients with information about the model's accuracy along with their results. This study provides insights into the most effective filters for brain tumor MRI pictures. Future study aims to improve the accuracy, effectiveness, and clinical application of the brain tumor detection system, resulting in better patient outcomes and supporting medical professionals with diagnostic decision-making.

Bibliography

[1] Padmavathi, K. and Megala, C., 2015. Detection of brain tumour with filtering techniques. International Journal of Advanced Research in Computer and Communication Engineering, 4(7).
[2] Ravichandran, S. and Rajendran, P., Brain Tumor Research Publication During The Period 2012-2021: A Scientometric Study.
[3] Wu, N., Phang, J., Park, J., Shen, Y., Huang, Z., Zorin, M., Jastrzębski, S., Févry, T., Katsnelson, J., Kim, E. and Wolfson, S., 2019. Deep neural networks improve radiologists' performance in breast cancer screening. IEEE transactions on medical imaging, 39(4), pp.1184-1194.
[4] Lee, E.Q., Selig, W., Meehan, C., Bacha, J., Barone, A., Bloomquist, E., Chang, S.M., De Groot, J.F., Galanis, E., Hassan, I. and Kalidas, C., 2021. Report of National Brain Tumor Society roundtable workshop on innovating brain tumor clinical trials: building on lessons learned from COVID-19 experience. Neuro-oncology, 23(8), pp.1252-1260.
[5]Gamage, Praveen. (2017). Identification of Brain Tumor using Image Processing Techniques - Image Pre-Processing and Segmentation.
6[]Liu, D., Liu, Y. and Dong, L., 2019. G-ResNet: Improved ResNet for brain tumor classification. In Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part I 26 (pp. 535-545). Springer International Publishing.
[7]Belaid, O.N. and Loudini, M., 2020. Classification of brain tumor by combination of pre-trained vgg16 cnn. Journal of Information Technology Management, 12(2), pp.13-25.
[8]Mohan, R., Ganapathy, K. and Rama, A., 2021. Brain tumour classification of magnetic resonance images using a novel CNN based medical image analysis and detection network in comparison with VGG16. Journal of population therapeutics and clinical pharmacology, 28(2).
[9]Rasheed, Z., Ma, Y.K., Ullah, I., Al Shloul, T., Tufail, A.B., Ghadi, Y.Y., Khan, M.Z. and Mohamed, H.G., 2023. Automated Classification of Brain Tumors from Magnetic Resonance Imaging Using Deep Learning. Brain Sciences, 13(4), p.602.
[10]Sharma, A.K., Nandal, A., Dhaka, A., Polat, K., Alwadie, R., Alenezi, F. and Alhudhaif, A., 2023. HOG transformation based feature extraction framework in modified Resnet50 model for brain tumor detection. Biomedical Signal Processing and Control, 84, p.104737.
[11]Shah, M.F.M., 2019. Brain Tumor Detection using Convolutional Neural Network. Ahsanullah University of Science andTechnology.
[12]Methil, A.S., 2021, March. Brain tumor detection using deep learning and image processing. In 2021 international conference on artificial intelligence and smart systems (ICAIS) (pp. 100-108). IEEE.
[13]Ari, A. and Hanbay, D., 2018. Deep learning based brain tumor classification and detection system. Turkish Journal of Electrical Engineering and Computer Sciences, 26(5), pp.2275-2286.
[14]Paul, J.S., Plassard, A.J., Landman, B.A. and Fabbri, D., 2017, March. Deep learning for brain tumor classification. In Medical Imaging 2017: Biomedical Applications in Molecular, Structural, and Functional Imaging (Vol. 10137, pp. 253-268). SPIE.
[15]Rehman, A., Khan, M.A., Saba, T., Mehmood, Z., Tariq, U. and Ayesha, N., 2021. Microscopic brain tumor detection and classification using 3D CNN and feature selection architecture. Microscopy Research and Technique, 84(1), pp.133-149.
[16]Zhao, L. and Jia, K., 2016. Multiscale CNNs for brain tumor segmentation and diagnosis. Computational and mathematical methods in medicine, 2016.
[17]Deshpande, A., Estrela, V.V. and Patavardhan, P., 2021. The DCT-CNN-ResNet50 architecture to classify brain tumors with super-resolution, convolutional neural network, and the ResNet50.Neuroscience Informatics, 1(4), p.100013.
[18]Ahmad, I.S., Zhang, S., Saminu, S., Wang, L., Isselmou, A.E.K., Cai, Z., Javaid, I., Kamhi, S. and

Kulsum, U., 2021. Deep learning based on CNN for emotion recognition using EEG signal.
[19]Khan, H.A., Jue, W., Mushtaq, M. and Mushtaq, M.U., 2021. Brain tumor classification in MRI image using convolutional neural network. Mathematical Biosciences and Engineering.
[20]Çinar, A. and Yildirim, M., 2020. Detection of tumors on brain MRI images using the hybrid convolutional neural network architecture. Medical hypotheses, 139, p.109684.
[21]Huang, Z., Du, X., Chen, L., Li, Y., Liu, M., Chou, Y. and Jin, L., 2020. Convolutional neural network based on complex networks for brain tumor image classification with a modified activation function. IEEE Access, 8, pp.89281-89290.
[22]Bhanothu, Y., Kamalakannan, A. and Rajamanickam, G., 2020, March. Detection and classification of brain tumor in MRI images using deep convolutional network. In 2020 6th international conference on advanced computing and communication systems (ICACCS) (pp. 248-252). IEEE.
[23]Rahmathunneesa, A.P. and Muneer, K.A., 2019, November. Performance analysis of pre-trained deep learning networks for brain tumor categorization. In 2019 9th International Conference on Advances in Computing and Communication (ICACC) (pp. 253-257). IEEE.
[24]Tazin, T., Sarker, S., Gupta, P., Ayaz, F.I., Islam, S., Monirujjaman Khan, M., Bourouis, S., Idris, S.A. and Alshazly, H., 2021. A robust and novel approach for brain tumor classification using convolutional neural network. Computational Intelligence and Neuroscience, 2021.
[25]Kaur, D., Goel, S., Nijhawan, R. and Gupta, S., 2022, February. Analysis of brain tumor using pre-trained CNN models and machine learning techniques. In 2022 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS) (pp. 1-6). IEEE.
[26]Kumari, R. and Srivastava, S.K., 2017. Machine learning: A review on binary classification. International Journal of Computer Applications, 160(7).
[27]Amin, J., Sharif, M., Haldorai, A., Yasmin, M. and Nayak, R.S., 2022. Brain tumor detection and classification using machine learning: a comprehensive survey. Complex & intelligent systems, 8(4), pp.3161-3183.
[28]Saeed, M.U., Ali, G., Bin, W., Almotiri, S.H., AlGhamdi, M.A., Nagra, A.A., Masood, K. and Amin, R.U., 2021. RMU-net: a novel residual mobile U-net model for brain tumor segmentation from MR images. Electronics, 10(16), p.1962.
[29]Rajinikanth, V., Joseph Raj, A.N., Thanaraj, K.P. and Naik, G.R., 2020. A customized VGG19 network with concatenation of deep and handcrafted features for brain tumor detection. Applied Sciences, 10(10), p.3429.
[30]Zhang, L., Zhang, H., Rekik, I., Gao, Y., Wang, Q. and Shen, D., 2018. Malignant brain tumor classification using the random forest method. In Structural, Syntactic, and Statistical Pattern Recognition: Joint IAPR International Workshop, S+ SSPR 2018, Beijing, China, August 17–19, 2018, Proceedings 9 (pp. 14-21). Springer International Publishing.
[31]Faraz, N., Naz, B. and Memon, S., 2022. Data mining approach for detection and classification of brain tumor. Mehran University Research Journal Of Engineering & Technology, 41(1), pp.53-64.
[32]Khagi, B. and Kwon, G.R., 2021. Convolutional neural network-based natural image and MRI classification using Gaussian activated parametric (GAP) layer. IEEE Access, 9, pp.96930-96947.
[33]Rajinikanth, V., Satapathy, S.C., Fernandes, S.L. and Nachiappan, S., 2017. Entropy based segmentation of tumor from brain MR images–a study with teaching learning based optimization. Pattern Recognition Letters, 94, pp.87-95.
[34]Mohan, R., Ganapathy, K. and Rama, A., 2021. Brain tumour classification of magnetic resonance images using a novel CNN based medical image analysis and detection network in comparison with VGG16. Journal of population therapeutics and clinical pharmacology, 28(2).
[35]Arshed, M.A., Shahzad, A., Arshad, K., Karim, D., Mumtaz, S. and Tanveer, M., 2022. Multiclass Brain Tumor Classification from MRI Images using Pre-Trained CNN Model.
[36]Lathashree, P.S., Puthran, P., Yashaswini, B.N. and Patil, N., 2022. Brain MR Image Segmentation for Tumor Identification Using Hybrid of FCM Clustering and ResNet. In Inventive Systems and

Control: Proceedings of ICISC 2022 (pp. 231-245). Singapore: Springer Nature Singapore.

[37]Saeed, M.U., Ali, G., Bin, W., Almotiri, S.H., AlGhamdi, M.A., Nagra, A.A., Masood, K. and Amin, R.U., 2021. RMU-net: a novel residual mobile U-net model for brain tumor segmentation from MR images. Electronics, 10(16), p.1962.
[38]Arshed, M.A., Shahzad, A., Arshad, K., Karim, D., Mumtaz, S. and Tanveer, M., 2022. Multiclass Brain Tumor Classification from MRI Images using Pre-Trained CNN Model.
[39]Rajinikanth, V., Joseph Raj, A.N., Thanaraj, K.P. and Naik, G.R., 2020. A customized VGG19 network with concatenation of deep and handcrafted features for brain tumor detection. Applied Sciences, 10(10), p.3429.
[40]Gayathri Shrikanth & Sanika Mhadgut. Brain Tumor Detection using FastAl and OpenCV. May 5,2020. Brain Tumor Detection using FastAI and OpenCV
[41]Pravitasari, A.A., Iriawan, N., Almuhayar, M., Azmi, T., Irhamah, I., Fithriasari, K., Purnami, S.W. and Ferriastuti, W., 2020. UNet-VGG16 with transfer learning for MRI-based brain tumor segmentation. TELKOMNIKA (Telecommunication Computing Electronics and Control), 18(3), pp.1310-1318.
[42]Deshpande, A., Estrela, V.V. and Patavardhan, P., 2021. The DCT-CNN-ResNet50 architecture to classify brain tumors with super-resolution, convolutional neural network, and the ResNet50. Neuroscience Informatics, 1(4), p.100013.
[43]Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. {TensorFlow}: a system for {Large-Scale} machine learning. In 12th USENIX symposium on operating systems design and implementation (OSDI 16) (pp. 265-283).
[44]Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2012. Scikit-learn: Machine learning in python. ArXiv. Org.
[45]Culjak, I., Abram, D., Pribanic, T., Dzapo, H. and Cifrek, M., 2012, May. A brief introduction to OpenCV. In 2012 proceedings of the 35th international convention MIPRO (pp. 1725-1730). IEEE.